Edition of Research Texts on
Informatics and Information Technologies

Artificial Intelligence and Cognitive Science IV

PUBLIKÁCIU PODPORILO ZDRUŽENIE

GRATEX IT INŠTITÚT

v rámci fondu GraFIIT www.gratex.com

This publication is supported by grant agency VEGA SR under grants VEGA 1/0553/12 and VEGA 1/0458/13

Vladimír Kvasnička
Jiří Pospíchal
Pavol Návrat
David Chalupa
Ladislav Clementis
(editors)

Artificial Intelligence and Cognitive Science IV

Slovak University of Technology in Bratislava

PUBLIKÁCIU PODPORILO ZDRUŽENIE

GRATEX IT INŠTITÚT

v rámci fondu GraFIIT www.gratex.com

All contributions have been reviewed by the Editorial Board of series *Artificial Intelligence and Cognitive Science.*

Authorized by Dean of Faculty of Informatics and Information Technologies of Slovak University of Technology in Bratislava by a decision no. file 1/01-04-2014 in April 1, 2014.

ISBN 978-80-227-4208-5

CONTRIBUTING AUTHORS

Ing. Ivana Budinská, PhD.
E-mail: budinbska@savba.sk
Institute of Informatics, Slovak Academy of Science, Dúbravská 9, 845 07 Bratislava

Ing. Marek Bundzel, PhD.
E-mail: marek.bundzel@tuke.sk
Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, University of Technology, 04001 Košice

Ing. Peter Grančič, PhD.
Email: peter.grancic@vscht.cz
Laboratory of Chemical Robotics, Prague Institute of Chemical Technology, Technická 5, 16228 Prague

Doc. Ing. František Štepánek, PhD.
Email: frantisek.stepanek@vscht.cz
Laboratory of Chemical Robotics, Prague Institute of Chemical Technology, Technická 5, 16228 Prague

Mgr. Ján Gondoľ, PhD.
E-mail: jan.gondol@fpf.slu.cz
Institute of Informatics, Faculty of Philosophy and Science, Silesian University, Bezručovo nám. 13, 74601 Opava, Czech Republic.

Prof. Ing. Vladimír Kvasnička, DrSc.
Email: kvasnička@fiit.stuba.sk
Faculty of informatics and information technologies, Slovak Technical University in Bratislava, Ilkovičova 2, 842 16 Bratislava

prof. RNDr. Jiří Pospíchal, DrSc.
E-mail: pospichal@fiit.stuba.sk
Faculty of informatics and information technologies, Slovak Technical University in Bratislava, Ilkovičova 2, 842 16 Bratislava

Ing. Fedor Lehocki, PhD.
Email: fedor.lehocki@stuba.sk
Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology, Slovak Technical University in Bratislava, Ilkovičova 3, 812 19 Bratislava

Ing. Lucia Cibulková
Email: cibulkova.lucia@gmail.com
Institute of Computer Science and Mathematics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Ilkovicova 3, 812 19 Bratislava

Doc. Pavel Nahodil, PhD.
Email: nahodil@fel.cvut.cz,
Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Technická 2, 166 27 Praha 6 – Dejvice

Ing. Jaroslav Vitků
E-mail: vitkujar@fel.cvut.cz
Department of Cybernetics, Czech Technical University in Prague, Faculty of Electrical Engineering, Technická 2, 166 27 Praha 6

Martin Peniak
E-mail: martin.peniak@plymouth.ac.uk
School of Computing and Mathematics, Faculty of Science and Technology, University of Plymouth, PL4 8AA, Plymouth, UK

Prof. Angelo Cangelosi
Email: A.Cangelosi@plymouth.ac.uk
School of Computing and Mathematics, Faculty of Science and Technology, University of Plymouth, PL4 8AA, Plymouth, UK

Doc. PhDr. Karel Pstružina, CSc.
Email**:** Pstružin@vse.cz
University of Economics, Department of Philosophy, sqr. W. Churchill 3, 13000 Praha

Mgr. Igor Sedlár, PhD.
Email: sedlar@fphil.uniba.sk
Department of Logic and Methodology of Science, Faculty of Arts, Comenius University, Bratislava

Doc. PhDr. Ján Šefránek, PhD.
Email: sefranek@ii.fmph.uniba.sk
Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University, 84215 Bratislava

Ing. Ján Vaščák, PhD.
Email: jan.vascak@tuke.sk
Center for Intelligent Technologies, Department of Cybernetics and Artificial Intelligence, Technical University of Košice, Letná 9, 042 00 Košice

Ing. Mária Virčíková
Email: maria.vircikova@tuke.sk
Center for Intelligent Technologies, Department of cybernetics and artificial intelligence, Faculty of electrical engineering and informatics, Technical university of Kosice, Letná 9, 040 01 Košice

Prof. Ing. Peter Sinčák, CSc.
Email: peter.sincak@tuke.sk
Center for Intelligent Technologies, Department of cybernetics and artificial intelligence, Faculty of electrical engineering and informatics, Technical university of Kosice, Letná 9, 040 01 Košice

Foreword

Artificial intelligence and cognitive science are already well established sciences with interdisciplinary and transdisciplinary focus. They provide rigorously oriented computer scientists with wings of imagination, allowing overlap of their technical / scientific disciplines with the humanities and behavioral sciences.

In Slovakia, these two study branches have already their study programs: Artificial intelligence at the Faculty of electrical engineering and informatics, Technical university of Košice, and Cognitive science at the Comenius University in Bratislava. For more than ten years, the Czecho-Slovak conference Cognition and Artificial Life brings together experts from different fields of science (computer scientists, mathematicians, philosophers, psychologists, doctors, ...), who are unified by their interest in learning about the cognitive processes ongoing in the human brain and the modern simulation methods of artificial intelligence: neural networks, evolutionary algorithms and multi-agent systems.

The series Artificial Intelligence and Cognitive Science aims to present the results of this community, achieved in artificial intelligence and cognitive science, in order to help undergraduate students in particular, but also more advanced students. The contents of the book Artificial Intelligence and Cognitive Science IV shows that these sciences have found a base both in philosophy, social science, as well as in natural sciences, and computer science oriented researchers in the areas of philosophy of mind, cognitive psychology, and artificial life.

We are thankful to all authors who contributed to this book. We also express our gratitude to Gratex International, Inc., represented by Ing. Ivan Polášek, PhD., for financial support for publication of this book and to our Faculty of Informatics and Information Technologies STU in Bratislava, namely to the Dean Assoc. Prof. Pavol Čičák, for encouraging of the publication of this edition and its inclusion in faculty editorial series "Edition of Research Texts on Informatics and Information Technologies".

In Bratislava, July 2014

Vladimír Kvasnička
Jiří Pospíchal

Contents

Preface

(1) How artificial agent think 1
Ivana Budinská
1 Introduction 1
2 Agent and multi-agent systems 2
3 Formal representation of knowledge 9
4 Conclusion 18
References 18

(2) Brain Theory Applied 21
Marek Bundzel
1 Introduction 21
2 Memory-prediction Theory of Brain Function 22
3 Description of the Proposed System 27
4 Experiments 37
5 Problems, Discussion and Future Work 41
6 Conclusion 42
References 43

(3) The Future of Search: Perspectives from IBM, Apple, Google, and Others 45
Ján Gondoľ
1 Introduction 45
2 IBM Watson 51
3 Apple Siri 52
4 Google Goggles 54
5 The Past and The Future 60
6 Conclusion
Bibliography 62

(4) Modeling of the Collective Behavior of Chemical Swarm Robots 67
Peter Grančič and František Štěpánek
1 Introduction 67
2 Simulated experiments 71
3 Modeling methodology 73
4 Results and discussion 76
5 Conclusions 86

6 Appendix 1 86
7 Appendix 2 87
References 88

(5) Warren McCulloch & Walter Pitts – Foundations of logical calculus, neural networks and automata 93
Vladimír Kvasnička and Jiří Pospíchal
1 Introduction to basic concepts 93
2 Boolean functions 97
3 Formal specification of neural networks 107
4 Finite state machines [automaton] 110
5 A view of artificial intelligence and cognitive science on the ..... problem of relationship between mind and brain 116
6 Discussion and final notes 121
References 122

(6) Clinical Decision Support Systems and Reasoning with Petri Nets 125
Fedor Lehocki and Lucia Cibulková
1 Introduction 125
2 Clinical Decision Support System 127
3 Clinical Workflow Systems 136
4 Clinical Practice Guidelines and Computer Interpretable Guidelines 139
5 Logical and Fuzzy Petri Nets 143
6 Discussion 155
References 156

(7) Ethology-Inspired Design of Autonomous Creatures in Domain of Artificial Life 159
Pavel Nahodil and Jaroslav Vítků
1 Introduction 159
2 Theoretical Foundation for Designed Agent Hybrid Architecture 161
3 Main Concepts Used in our Novel Approach 174
4 Selected Simulations 177
5 Conclusion 184
Literature 185

(8) Multiple Time Scales Recurrent Neural Network for Complex Action Acquisition: Model Enhancement with GPU-CUDA Processing 187
Martin Peniak and Angelo Cangelosi
1 Introduction 187
2 Method 190
3 Experiments and Results 195
4 Conclusions and Future Work 198
References 199

(9) What we mean when we talk about the mind 203
Karel Pstružina
1 Human thinking 203
2 Creation of concepts 207
3 The Mind 212
Literature 216

(10) Logic and Cognitive Science 219
Igor Sedlár and Ján Šefránek
1 Introduction 219
2 Cognition and Truth 220
3 Reasoning: A Psychological Point of View 221
4 Logic Strikes Back 226
5 Logic and Human Reasoning 226
6 Conclusions 233
References 234

(11) Evolutionary and Genetic Fuzzy Systems 237
Ján Vaščák
1 Introduction 237
2 Fuzzy Inference Systems - Ways of Adaptation 239
3 Adaptation of Rule-based Fuzzy Inference Systems Using Genetic Algorithms 242
4 Modifications and Enhancements of Genetic Fuzzy Systems - Present State-of-Art 249
5 Fuzzy Genetic Systems - Some examples 264
6 Outlooks and Conclusions 269
References 270

(12) Sentio, Ergo Sum: From Cognitive Models of Emotions Towards their Engineering Applications 275
Mária Virčíková and Peter Sinčák
1 Introduction - from human emotion to emotion machine 275
2 Implementation of emotional models 278
3 Computational intelligent techniques for modeling emotions 283
4 Survey of computer emotional models 289
5 Engineering applications of emotional technology 293
6 Quo vadis artificial emotions? 295
References 297

Index 303

How artificial agents think

Ivana BUDINSKÁ [1]

Abstract. Autonomy and intelligence are the characteristics that distinguish software agents from other software entities. These software agents' capabilities depend on the knowledge base and on agent´s behavior. Software agents are considered as autonomous entities with knowledge base and reasoning and discovering new knowledge. Basic characteristics of agent systems are deprived from their architectures, formal representation of knowledge and embedded behavioral models. Specific group of agent systems are multi-agent systems (MAS) that possess great capability of communication, cooperation and coordination. Agents within MAS share the knowledge. Another very important agent systems feature is the capability of learning from experience. Agents can modify their knowledge bases in accordance with cognitive, perception and reactive processes very similarly to the way how natural systems do it.

1 Introduction

Artificial agents can be any artificial entity that is characterized by a certain level of intelligence, working and acting autonomy, and learning capability. Recently artificial agents and agents' paradigm are popular in computer science community, where artificial agent is considered as a piece of code and is called software agent. Comparing to other software entities, e.g. objects, although both represent modular programming approach, there are two key areas where object oriented (OO) approach differs most from agent based technology, namely autonomy and interaction. As to autonomy, objects´ methods have to be invoked by some external entity and so they are considered as being passive elements depending on external actions, while agents do it themselves. Agents are considered autonomous and self-initiating. Agents encompass not only code, but also data, rules and goals that make agents active elements. Agents can observe their environment and decide when and what to do in case of an event occurrence. However, the latest development of programming technology and methods enables also objects being more active by applying some of event-

[1] Institute of Informatics, Slovak Academy of Sciences, E-mail: budinska@savba.sk.sk

listener frameworks introduced by UML and JAVA. The second difference between the OO approach and agents technology is in the way how they communicate. Agent communication model is based on asynchronous messaging and parallel processing and these features are supported by agent-based programming languages. OO languages do not support these features. The communication has to be resolved on the top of OO model.

Agent based technology is usually considered to have one or more of the following advantages comparing to other programming approaches:

- intelligence,
- autonomy,
- decentralization,
- parallelism,
- emergence
- analogies from nature.

It can be said briefly that agents having ability of making observation, decisions, and autonomous action, they can be considered as entities with certain capabilities of thinking. The concept of thinking (thought) usually refers to mental activities resulting in a kind of subjective resolutions. It is related to other concepts including reasoning, cognition, consciousness and imagination. Thought also refers to other important concepts: knowledge and language. Artificial agents have to be equipped with a knowledge base and a language in order to provide thinking activities – decision on when, what and how to do. Therefore the important question in building artificial agents is how to include knowledge into the agent and what kind of language could be used to make abstract thinking and to communicate with other agents either artificial or natural (human beings, animals, etc.).

The chapter is organized as follows: Section 2 introduces basic concepts and definitions from agents´ and multi-agents theory, basic agents´ architectures and common communication languages. Section 3 outlines knowledge representation issues, introduces common knowledge description languages and presents ontology approach to the knowledge formalization and representation. Section 4 deals with knowledge formalization issues related to the Semantic Web and Section 5 provides a brief introduction to agents´ reasoning and an overview of some popular reasoners. Conclusion is given in Section 6.

2 Agents and multi-agent systems

Agents' theory in computational science introduces a concept agent, its functionalities and features, and formal representation of agents. One of the

most general agent´s definitions considers agent as anything what has the capability to perceive and respond environment [22]. Another agent theory deals with agents´ languages, platforms and communication protocols that ensure the basic feature of an agent – communication with its environment.

2.1 Agents architectures

According to Nwana [13], a basic architecture of an agent reflects the three basic agent´s functions: perception, cognition and reaction.

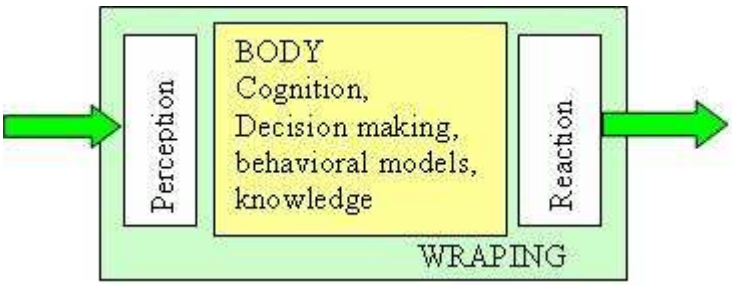


Fig. 1. Basic agent architecture (according to Nwana)

Basic types of agent's architectures are as follows:

1. **Deliberative architecture** - based on a symbolic model of environment. Sometimes it is also known as BDI (Belief-Desire-Intention) architecture that is based on symbolic model of environment. Deliberative agent implicitly encompasses a symbolic model of world and its logical relations. The model is completed by perception from real environment that came from sensors during an agent´s life. However, BDI architecture has not been widely applied in real life applications because of its long reaction time in dynamically changing environment.
2. **Reactive architecture** – developed as a response to the problems connected with deliberative architecture. Compared to deliberative architecture, reactive one does not include any symbolic model of environment and does not apply any complex symbolic reasoning. The most relevant works that contributed to a reactive architecture development include those by R. Brooks, who suggested the so called Subsumption architecture [2]. The basic ideas of this architecture are as follows:
    a. Intelligent behavior can be generated without explicit symbolic representation such as symbolic artificial intelligence.

b. Intelligent behavior is a result of flexible and fault-tolerant rules instead of a result of planning and scheduling.
c. Intelligent behavior is the result of an agent's interaction with environment. Agent´s intelligence is gained and developed during its life. It is not an isolated, native feature of the agent.

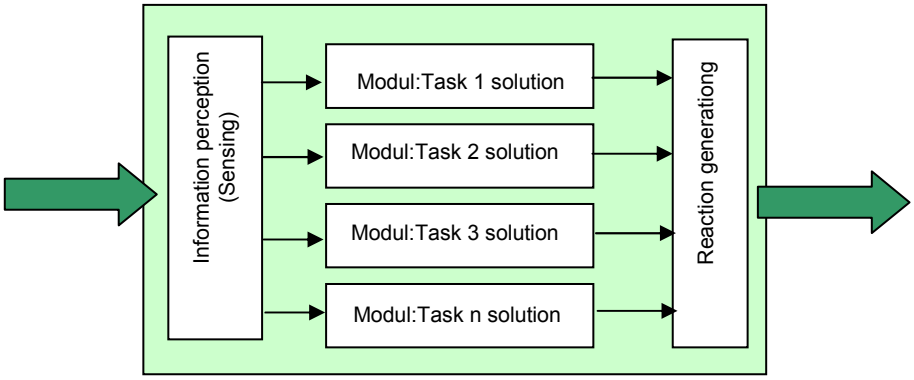


Fig. 2. Subsumption architecture according to Brooks.

Basic idea of reactive architecture lays in agent´s continuous interaction of with its environment. This interaction substitutes the existence of a symbolic model of environment. According to Brooks, intelligence of such systems increases in case of more agents involved in the system. Architecture of reactive agents is based on parallel modules that are responsible for solving different tasks (see **Fig. 2**). The modules are organized in a hierarchical way and they also encompass hierarchical structure of behavioral models that ensure the achievement of the goals. Overall plan is generated with regard to inputs coming from environment as reactions to these inputs. The agent does not need overall built-in plan. Parallel structure of modules leads to increasing fault tolerance, because agent can accomplish the tasks even when one or more modules are corrupted. Hierarchical structure of behavior models comes from their nature characteristics. The lowest level is created by very simple behavioral models that solve the simplest situations (e.g. in mobile robotics – go one step forward, turn back, etc.) More complex behavioral models (e.g. in mobile robotics – follow the leader, avoid a barrier, grasp an object, etc.) create the higher level. In such systems, priority is organized bottom up, that is the simpler behavior models are prioritized to higher level models. The modules are built as simple as possible without applying complex evaluating algorithms or reasonings, in order to increase effectiveness of such systems for use by real time control of systems.

3. **Hybrid architecture** – assumes multiple level hierarchical structures, where the lowest level encompasses reactive part of the agent and the higher level is dedicated to more intelligent, deliberative behavior. These parts have to be separated, because each of them is described in a different way. The reactive part can be modeled by state machine and the deliberative part is described by more complex methods using formalisms of symbolic artificial intelligence [9]. An architecture based on hybrid behavioral model is described in [3]. The agent implies a set of behavioral modules. Classical subsumption architecture enriches by a table – APB (Action Potential Blackboard) that serves as a common area for each of the modules. The resulting behavior is created according decision and planning algorithms that are built-in the APB. This communication through an APB is similar to the common data area used by network communication FAN (Field Area Network). Coordination via message passing is similar to the networked, shared values used by FAN [16].

2.2 Multi Agent Systems

The problems with development complex agent systems end in an idea of building simpler systems that would be able to solve partial tasks and to cooperate in solving a complex problem. A system of cooperating agents is called a multi-agent system (MAS). Building of multi-agent systems seems to be a more convenient and realistic approach in building intelligent systems. While in one-agent systems it is required that the agent encompasses the knowledge about its environment, about goals and actions, agents within a multi-agent systems can be modeled as simpler entities that are responsible for solving of relevant particular tasks. The reason for introducing multi-agent technology was the need to solve complex problems from distributed artificial intelligence.

According to M. Wooldridge [22], MASs represent a group of loosely coupled, relatively independent, intelligent objects – agents that cooperate in order to achieve a common goal. The advantage of multi-agent systems is that their capability to solve complex problems is above the capabilities of individual agents included in the MAS. MAS architecture does not assume common architecture of individual agents. The architecture does not impose requirements on individual agents, which may be rather diverse.

K. P. Sycara[2] defines the following reasons for applying MASs [18]:

- one agent does not have the capacity to solve a complex task,

[2] http://www.aaai.org/AITopics/html/multi.html

- one agent does not have sufficient information to solve a complex task.

Basic features of multi-agent systems are as follows:

- Each agent possesses just limited information resources, or limited capability to solve a complex problem. Therefore each agent approaches to solving of a problem from a different point of view.
- There does not exist any global centralized coordination within MASs. The data structure is also decentralized and agents work asynchronously.

There are several problems addressed in the theory of multi-agent systems:

- formulation and decomposition of a problem and the following synthesis of results within a group of agents,
- communication and interaction among agents, communication languages, protocols and the way and content of time synchronization of communication,
- agents´ activities coherence, blocking undesirable interactions that lead to the system perturbation and/or to defective results,
- coordination and cooperation of agents. Agents should cooperate with representing of knowledge and by reasoning about other agents,
- collisions and conflicts avoidance, recognition and evaluation of conflict situations from the point of view of a global goal, limited resource allocation in case of conflict situations, avoidance of undesirable behavior that can lead to unpredictable behavior.

An agent within a multi-agent system should have some social and communication abilities in order to cooperate with other agents.
In order to communicate, agents must be able to[3]:

- deliver and receive messages,
- parse the messages,
- understand the messages.

Since 2005 there exists the FIPA[4] (The Foundation for Intelligent Physical Agents) organization for agents and multi-agent systems as an IEEE Computer Society standards organization. FIPA works on and releases specifications represented by a collection of standards in order to enable interconnection and interoperation of different artificial agents and agent systems. FIPA proposed a FIPA Agent Management Reference Model (**Fig. 3**).

[3] http://www.obitko.com/tutorials/ontologies-semantic-web/
[4] http://www.fipa.org/

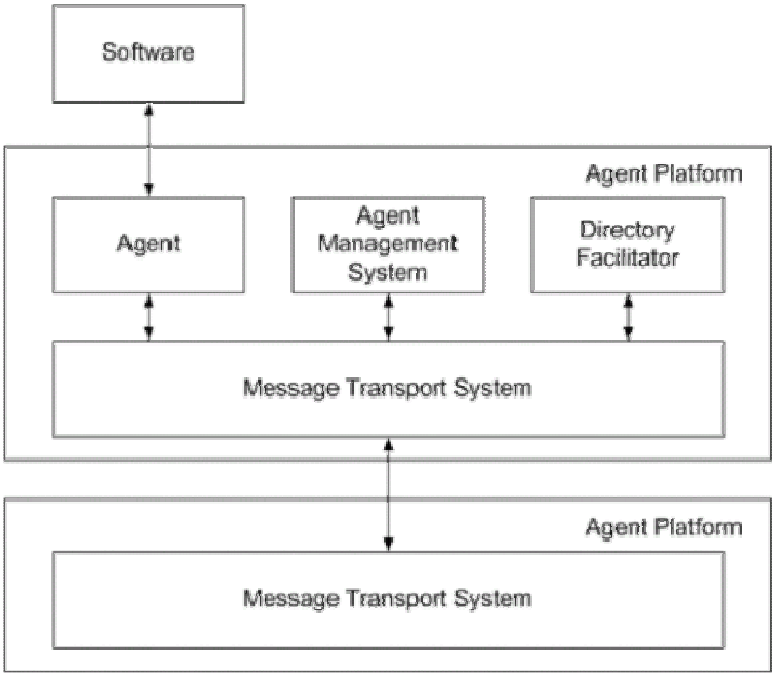


Fig. 3. A FIPA Agent Management Reference Model

2.3 Communication languages

Interaction among agents is crucial for agents systems running. Interaction comprises communication between agents and environment and also with other agents. There exists an object oriented (OO) message broker that matches each single message to exactly one method that invokes only one object. The communication among agents is more complex. There are three basic features that ensure a proper communication among any entities:

- common language,
- common understanding of the knowledge to be exchanged,
- effective way of knowledge interchange.

A communication language should address issues like whom to talk to and how to find communicating partners, when and how to initiate the communication and how to maintain an exchange of messages. Agent Communication Languages (ACL) were designed to respond the requirements of agents systems communication. Generally they also include Interaction and Transport protocols. Although ACLs are formal and unambiguous, they may vary in format and concepts and should be specified for each application. Having

a number of ACLs without a proper standardization may cause difficulties with interaction among agents systems. There are two popular implementations of ACLs that try to support some kind of standards for interconnection among agents: KQML and FIPA ACL. The both languages are based on a speech act theory. They describe communication acts either by natural language or by the terms of a kind of formal representation (e.g. modal logic for FIPA ACL).

KQLM – Knowledge Query and Manipulation Language [6] – consists of three layers: the content layer, the message layer, and the communication layer.

The message layer is the core layer of the KQLM. Protocols for messages transport are identified. Each message is attached by a speech act or performative that identifies the type of performatives such as a query, a command, etc. The message layer may also include additional features that better describe the content of message. These features are related to a message language and ontology. KQLM implementations enable to deliver messages properly even when the message´s content is corrupted and inaccessible.

The content layer describes what kind of message has to be communicated. It bears the content of the message in the related representation language.

The communication layer describes the communication parameters, defines a sender and a receiver.

An example of KQLM message that represents a query from agent *joe* about the price of a IBM stock share:

```
(ask-one
   : sender joe
   : content (PRICE IBM ?price)
   : receiver stock-server
   : reply-with ibm-stock
   : language LPROLOG
   : ontology NYCE-TICKS)
```

In this example the message layer is formed by a performative ask-one, and by performative names : language and : ontology values. The content layer contains a value of the : content keyword - (PRICE IBM ?price). The communication layer identifies the sender and receiver by values of the : sender and : receiver keywords, respectively. A performative ask-all forms a message that requests a set of all answers. Such a request could be conveyed using standard PROLOG as a content language. The above example and some more of them could be found in [6].

Reserved performative names of KQLM can be categorized into seven basic groups:

Basic queries	evaluate, ask-if, ask-about, ask-one, ask-all
Multi-response (query)	stream-about, stream-all, eos
Response	reply, sorry
Generic information	tell, achieve, cancel, untell, unachieve
Generator	stand-by, ready, next, rest, discard, generator
Capability definition	advertise, subscribe, monitor, import, export
Networking	register, unregister, forward, broadcast, route

KQLM implies many information exchange protocols. The simplest is the client-server protocol, where an agent acting as a client sends queries to another agent that represents a server. The agent acting as a server replies to the queries by a single answer or by a set of single answers.
Semantics of KQLM is defined in terms of precondition, postconditions and completion conditions.

FIPA ACL[5] – FIPA Agent Communication Language is an agents communication standard proposed and supported by FIPA [15]. It is based on modal language, and consists of a set of message types and related descriptions. FIPA ACL semantics is based on Semantic Language that represents propositions, objects, and actions.

3 Formal representation of knowledge

Intelligence is often defined as an ability to acquire and employ knowledge. Building a knowledge base is crucial for building any kind of knowledge systems. In some cases, e.g. expert systems, knowledge base is more important for problem solving than the ways how the knowledge is processed and managed (i.e. algorithms).

3.1 General introduction

In order to formalize knowledge, the basic concept of knowledge should be introduced. Knowledge stands on the top of a pyramid, where data create the base and information is in the middle layer. Data are pure facts (numbers, symbols, letters, etc.) that need additional explanation to be understood. Data together with an explanation give information. For instance a number 20 does

[5] http://www.fipa.org/repository/aclspecs.html

not say anything without additional explanation. It could refer to an application domain, or to another value. Saying, that 20 is a temperature might be useful, but without specification that it is temperature given in degrees centigrade, it is still not sufficient. For people not used to SI metrics, there is still something missing to change this information into right knowledge. They need to know how to convert 20 °C into °F. Additional information might come from experience. One has to know that 20 °C is quite convenient temperature for walking in the town. All those compose complex knowledge. It is not easy to formalize that kind of complex knowledge that artificial agents can understand and reason on it. To convert information into knowledge human beings need to do many activities. It is assumed that a common human being possesses certain amount of knowledge. This can be either in-born or acquired. Knowledge acquisition mostly refers to classical learning in classes and then to studying. Much knowledge can be gained by unaware observation of environment. Some knowledge appears in connection with other knowledge. Sometimes one gains a new knowledge – a meaning of concepts (a new word, a name, title, etc.). It is highly possible that from that time he or she will meet the new concept very often. It will appear in newspapers, radio, TV, etc. It does not always mean that the concept did not occur before, or that the frequency of occurrence of that concept is higher. It could be explained in such a way that the one's consciousness became more sensitive to that new concept, before the concept was unknown and imperceptible. Similarly artificial agents can only handle those concepts they know. It is up to the knowledge base builders to make the knowledge base understandable for artificial agents.

There is a correlation between abstract thinking and language. From that point of view the best way for knowledge base creation could be applying a good vocabulary and thesaurus. The problem with dictionaries and thesauri is that they are never good enough. Although the development of machine automatic or semiautomatic machine translation is enormous, the time when a really good and general dictionary will be created is far still. The question is whether we need to create a really "Ultracomplex Maximegaloman Dictionary of All Known Languages" (D. Adams [1]). Many entries within a dictionary have different meaning according to different situations and contexts. The question is how to learn an agent to understand the right meaning of a concept in a specific context. Natural language processing (NLP) methods can help artificial agents cope with this problem. Another approach is to create an artificial language suitable for high intelligent artificial (but maybe also for natural) agents. A similar idea presents Doxiadis´ Professor from Logikomix [4]:

„The ordinary language is not suited to science! So in order to understand reality we must first create a language that is completely logical!"

While building a knowledge base we often refer to the concepts „controlled vocabulary“ and „taxonomy“. Both terms are related to another concept that is very relevant to the knowledge formalization and modeling – ontology. An explanation of relations among vocabulary, taxonomy and ontology is given by Woody Pidcock in his paper[6] with comments by Michael Ushold. A vocabulary, or better a controlled vocabulary, is a collection of the terms, where each of terms is associated with an original and nonredundant definition. Such a vocabulary should be provided by a registered authority. Taxonomy is a group of controlled vocabularies that are hierarchically organized. Each term in a taxonomy is at least in one of the parent-child type relations with another term in the taxonomy. Creating associative relations between terms in taxonomy, a thesaurus is built. Associative relations between terms may be very simple like „a term A is associated with a term B“. There are no rules defined on this level and it is not possible to create knowledge models for an application domain. A meta-model that could be an ontology, could serve for this purpose. Michael Ushold in his comments to the paper[7] adds that the difference between a vocabulary, a taxonomy and an ontology is that ontology associates to the terms in different context. While a vocabulary represents a set of terms with very specific meanings and for the application domain that is very clear, taxonomy adds another meaning to the terms by defining their relations to the terms in other vocabularies. In the frame of artificial intelligence and knowledge modeling, ontology represents a tool with a very rich language that is based on formal logic and intended for specification of the terms.

Recently there exist a great variety of languages and tools that ease ontology building. Building great application domain ontologies depends on domain experts knowledge. Computer aided ontology building is the way how domain experts can create great ontologies without extra knowledge of information technology or artificial intelligence. User interfaces are developed in such a way that experts can concentrate on the right definition of terms and their relations. Computer aided ontology building enables automatic processing of encompassed knowledge. Ontology, in the context of computer and information sciences, represents domain knowledge as a set of representational primitives – classes, and attributes – properties. Classes include information about the semantic meaning of the terms and the constraints regarding the modeled application domain. Comparing to database systems, ontologies are more general, they put higher level of abstraction on top of data models.

[6] http://infogrid.org/wiki/Reference/PidcockArticle
[7] http://infogrid.org/wiki/Reference/PidcockArticle

Therefore ontology languages enable abstraction from data structures and database systems implementation. It can be said that ontologies represent semantic level of knowledge modeling while databases stay on logical and physical data models.

3.2 Ontology – history, basic concepts

The term ontology is known from philosophy and refers to the theory of being and existence. The world „ontology“ comes from the Greek and Pythagoreans started to use it while researching an abstract existence, by looking for order behind chaos occurrence (*he* is in Greek an abstract existence) [9]. Aristotle defined ontology as science of being qua being. He introduced primitive categories such as substance and quality that explain All That Is. They wanted to create all-explaining general ontology. Later, I. Kant refused the possibility of general ontology creating. The ontology was quite popular during the 19th and 20th centuries. Many intellectuals and scientists tended to ontological explanation of the world. Recently, with language developments and explosion of ubiquitous and ambient knowledge, the ontology was getting a new meaning. It appeared that this field of philosophy provides a unique theoretical background for development of intelligent software that is able to understand natural human language.

According to Webster[8] dictionary and thesaurus, ontology is:

“a branch of metaphysics concerned with the nature and relations of being,

a particular theory about the nature of being or the kinds of things that have existence.”

Raul Corazzon[9] defines ontology as a theory of objects and their relations. He distinguishes several types of objects (concrete and abstract, real and unreal, dependent and independent) and several types of relations (relations, dependencies, predictions). He distinguishes three basic ontology types:

Formal ontologies – studying objects from the point of view of the existence. Formal and informal methods of classical ontology are combined with modern mathematical methods of formal logic on this level. Formal ontology is a branch of science that studies forms, states and types of existence.

Descriptive ontologies – gathering information about a group of objects that can be either independent or consequent.

Formalized ontologies – trying to create formal codification of results acquired on the descriptive ontology level.

[8] www.merriam-webster.com

[9] Raul Corazzon: Theory and History of Ontology, www.ontology.co, (prístupné v januári 2011)

3.3 Ontology development methodologies

Creating ontology is not a trivial problem. It requires not only skills in information technologies but also considerable knowledge in the modeled domain. To ease the process of ontology creation a couple of methods have been suggested. The basic principles for building ontology may be derived from the CommonKADS methodology [17], which deals with the common principles of knowledge systems development. CommonKADS methodology was developed within a series of international research and application projects. The process of knowledge system development is structured in a couple of models that have to be created. On the "context" level of abstraction three models are suggested: organizational model, task model and agent's model. The organizational model describes the organization with the aim to discover the problems and opportunities of knowledge management. The task model represents the tasks that are performed within the organization. Task is anything that has to be executed by an agent. The agent model describes all agents – executors of tasks – their roles, competencies, capabilities, and limitations. Above the contextual level lays the conceptual level that covers the communication and knowledge models. The models are derived from the three models in the conceptual level. The knowledge model describes knowledge that is required to perform the tasks. The communication model figures communicative transactions between agents that perform the task. Finally, the design model is an artifact that describes the structure of a knowledge system to be created.

Ontology development methodologies help creating ontologies in various domain oriented applications. Several methodologies have been developed in order to formalize creating ontologies for industrial or other applications. Although ontology development methodologies are not mature enough, they can be helpful in developing ontology based knowledge systems. Overview of some methodologies is given e.g. in [11], [10], or [11].

The Methontology [6] has been developed for Software Life Cycle Processes. It supports project management processes (contains guidelines for planning, project control, quality control, etc.), ontology development processes (contains guidelines for the use of ontology, conceptualization of domain, formalization of ontology, implementation, etc.), and support activities (guidelines for knowledge acquisition, evaluation, ontology integration, documentation, version management, etc.).

[10] www.iet.com/Projects/RKF/SME/methodologies-for-ontologydevelopment.pdf

[11] http://ontoweb.aifb.uni-karlsruhe.de/About/Deliverables/D1.4-v1.0.pdf,

The TOVE methodology [8] was developed at Toronto University in order to help modeling of enterprise processes. The methodology goes from informal definitions to formal competency questions. The ontology must provide vocabulary to answer these questions. First the informal competency questions have to be answered and the basic terms from these answers are extracted. Using vocabulary the informal competency questions are formalized and the ontology has to be evaluated if it is complete.

On-To Knowledge methodology [18] was developed on the basis of KADS methodology. It also uses a method of competency questions [13]. On-To-Knowledge methodology uses a two-loop architecture, which is composed of knowledge processes and knowledge metaprocesses. Knowledge metaprocesses describes building ontology in 5 basic steps (with 13 sub-steps): Feasibility study, Kick-off, Refinement, Evaluation, Application and evaluation.

The methodology by Ushold and King was developed within the Enterprise project and was used in Enterprise Ontology [21] creation. However the methodology is general and may be used in other domains. The skeleton of Usholds and Kings' methodology contains four basic steps: Identification of the purpose of ontology building, building the ontology, evaluation and documentation. Ushold and King's methodology assumes the informal ontology development and then the formalization of the informal ontology by any of formal ontological languages. The procedure of informal ontology development includes collection of concepts by brainstorming, clustering of the collected concepts, and refinement of the concept set by investigating which concepts are basic, which are generic, or specific, what are the relations among them. Concept names have to be specified. Each concept has to be named by an original name that has only one meaning in the ontology. The meaning of the names has to be defined for each concept. The importance of informal ontology that is comprehensible for many people is the crucial idea of this methodology. The methodology by Ushold and King belongs to the most formalized methodologies and can be successfully used in many domain applications.

3.4 Ontology languages and tools

Informal ontology has to be represented by one of the formal ontology languages in order to build computer processed ontology that is only usable in knowledge management systems. Usually ontology development methodology has its own tool to support ontology and instances in formal ontology representation language. In this section brief description of some most used ontology languages and tools is:

Ontolingua

Ontolingua[12] is originally an Interlingua for ontology representation and sharing developed by KSL (Knowledge Systems Lab) at Stanford University. It is designed by adding frame-like representation and translation functionalities to KIF (Knowledge Interchange Format) which is a logic-based Interlingua for knowledge representation. It can translate from and to some description logics languages. Ontolingua itself does not have inference functionality. It has currently developed into a development environment which provides a set of ontology development functions (browse, create, edit, modify and use ontology) and a library of modular and reusable ontologies.

RDF

RDF (Resource Description Framework) is a framework for metadata description developed by W3C[13]. It defines the triplet <object, attribute, value>, in which object is called resource and can be represented by a web page, URL address, etc. A triplet itself can be an object and a value. Value can be a string or resource.

Attributes represent links between objects and values. RDF model is a base for creating a semantic network. RDF has an XML-based syntax (called serialization). But, RDF is different from such a language in that it is a data representation model rather than a language and that the XML's data model is the nesting structure of information and the frame-like model with slots[20].

OWL (DAML+OIL)

An OWL – Web Ontology Language is designed for use by applications that need to process the content of information instead of just presenting information to humans. OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics[14]. The application of the OWL format for ontology for the agent system is relatively new. An advantage of OWL ontology is the availability of tools that can reason about it. Tools provide generic support that is not specific to the particular subject domain. Constructing ontology in OWL enables to benefit from third party tools based on the formal properties of the OWL language.

OntoEdit

OntoEdit [10] is a professional tool that helps to create ontology based on On-To-Knowledge methodology and CommonKADS. OntoEdit contains inference machine based on the F-Logic. It plays crucial role in the evaluation process.

[12] http://www.ksl.stanford.edu/software/ontolingua (Available in December 2005)

[13] http://www.w3.org (Available in December 2005)

[14] http://www.w3.org/TR/owl-features/ (Available in December 2005)

Opposite to the description logic, F-Logic can express arbitrary powerful rules which quantify over the set of classes.

Protégé

Protégé [11] is a powerful tool for building and creating domain ontology. It supports some formal ontology languages as RDF, and OWL, contains customizable user interface, and has powerful plug-in architecture, that enables integration with other applications.

3.5 Building the Semantic Web

Current World Wide Web (WWW) is a huge library of interlinked documents that are transferred by computers and presented to people. It has grown from hypertext systems, but the difference is that anyone can contribute to it. This also means that the quality of information or even the persistence of documents cannot be generally guaranteed. Current WWW contains a lot of information and knowledge, but machines usually serve only to deliver and present the content of documents describing the knowledge. People have to connect all the sources of relevant information and interpret them themselves.

The next level is presented by the Semantic web initiative. It is a collaborative effort led by World Wide Web Consortium (W3C). It aims to enhance current web in order to support people in finding the information they are looking for. The Semantic web intends to advance from a huge distributed hypertext system to a huge distributed knowledge system. The difference between the current web and the semantic web is that the semantic web would support sharing data instead of sharing documents. A common framework that allows data sharing across the internet should be provided. Many efforts have been given to advance personalized web browsing.

The architecture of semantic web provided by W3C is shown in Fig. **4**. The first layer consists of URI identifiers and a character set - Unicode. Unicode is a standard for encoding international character sets. There are many languages that use many specific characters both written and spoken in texts on the www. The unicode standard enables all the languages to be used in standardized forms. URI stands for Uniform Resource Identifier, and it is known as a system of unique addresses of linked documents on the web. There are two additional terms – related to URI - URL and URN. URL (Uniform Resource Locator) is a subset of URI and refers to the address of a network location of documents (e.g. http://www.something.stg). Another subset of URI is URN that allows to identify a resource without implying its location and means of dereferencing it - an example is urn:isbn:0-123-45678-9. Sometimes also the abbreviation IRI can be found in connection with WWW. IRI stands for International Resource Identifier, and it is a more general concept than URI,

because it enables using Unicode characters within URI. Both Unicode and URI were standards for the classical WWW.

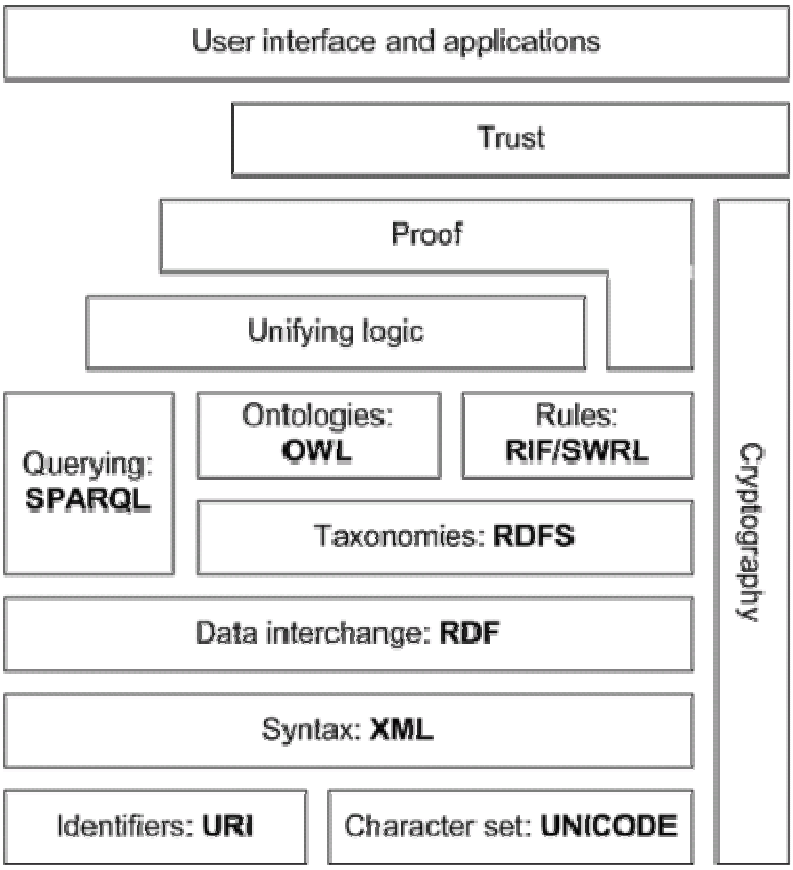


Fig. 4. Semantic web layered architecture

On top of URI and Unicode layer, there is syntax layer represented by XML (Extensible Markup Language). It was defined in order to support structured information in documents. XML enables specifying more different markup vocabularies within one XML document. A particular set of XML documents can be unified under specific XML schema.

A data interchange layer refers to the RDF – Resource Definition Framework. It is a framework that supports representation of resources´ information – resources metadata - in a graphical way. Beside metadata information (such as title, author, modification date, etc.), RDF can be used for storing any other data.

The next layer presents standardization of taxonomies. In order to formalize description of classes and properties, a RDF Schema (RDFS) was

designed. RDF Schema comes from its formal semantics within RDF. RDFS description of classes and properties can be used to create a lightweight, simple ontology. More advanced ontologies can be created using OWL (Web Ontology Language).

A Simple Protocol and RDF Query Language (SPARQL) were developed for querying RDF data, RDFS and OWL ontologies as well. SPARQL is a query language based on SQL syntax, but it uses RDF triples and resources for finding and returning results of the query. Beside a query language, SPARQL also provides a protocol for accessing RDF data.

After all semantics and rules are executed, a proof and trust of results have to be attested. Cryptography methods, such as digital signature, are used for verification of the origin of the sources. On the top layer, user interface is situated.

4 Conclusion

Ontologies are now beyond the control of philosophers. Building ontologies employ many experts from different areas. With the help of informatics and information technologies it seems to be a reasonable task. Ontology builders expect variety of features to support huge knowledge sharing on the base of ontology interconnection, joining and completion. There are ontologies for different application domains. Building useful and well suited knowledge systems requires development of new ontologies that enable domain dependent knowledge reasoning. There is still need for new ontology development for newer knowledge systems.

Acknowledgement. This work has been supported by the SRDA scientific grant agency under grant No. 0261-10 BioMRCS and by the VEGA scientific grant agency under grant No. 2/0197/10 .

References

[1] Adams D.: *Life, the Universe and Everything* (in Czech *Život, vesmír a všetko*, transl. Patrik Frank), SLOVART 2005

[2] Brooks A.R.: Intelligence without representation, In Artificial Intelligence, Vol. 47, pp. 139, 159, 1991

[3] Davis D.N.: *Reactive and Motivational Agents: Towards a Collective Minde*, Lecture Notes in Artidicial Intelligence 1193, Intelligent Agents II, Springer 1997

[4] Denny M.: *Ontology Tools Survey*, revised, 2004 http://www.xml.com/pub/a/2004/07/14/onto.html (accessed in January 2012)

[5] Doxiadis A., Papadimitriou Ch. H., art Papadatos A., color Donna A.: *Logicomix*, Bloomsbury 2009

[6] Fernandez-Lopez, M.: Meta-modelling for ontology development and knowledge exchange. In: 15th ECAI Conference Workshop 1 "*Ontologies and semantic interoperability*" Lyon 2002, p. 6-1

[7] Finin T., Labrou Y., Mayfield J.: *KQLM as an agent communication language,* a draft paper for a chapter in Jeff Bradshaw (Ed.), "Software Agents", MIT Press, Cambridge, 1995, http://download.polytechnic.edu.na/pub4/download.sourceforge.net/pub/sourceforge/j/project/ja/jadeutvt/LibrosManuales/KQML%20y%20KIF/KQML%20as%20an%20agent%20communication%20language.pdf (accessed in January 2012)

[8] Fox M. S, Gruninger M.: *Enterprise modeling*, www.eil.utoronto.ca/enterprise-modelling/papers/fox-aimag98.pdf, (Available in December 2005)

[9] Kostelník P.: *Practical introduction into symbolic artificial intelligence* (in Slovak *Praktický úvod do symbolickej umelej inteligencie*), in book *Umelá inteligencia a kognitívna veda I*., Ed. Kvasnička V., Pospíchal J., Kozák Š., Návrat P., Paroulek P., Edícia učebných textov FIIT STU, 2009.

[10] Lendvai F.L.: *History of thinking (*in Slovak *Dejiny myslenia*, transl. M. Zágoršeková), nakladateľstvo Pravda, 1985

[11] Mizoguchi R.: Part 2: *Ontology development, tools and languages*. Available at www.ei.sanken.osaka-u.ac.jp/pub/miz/Part2V3.pdf, (accessed December 2005)

[12] Motik B.: *KAON – Reasoning in Description Logic using Resolution and Deductive Databases*

[13] Noy, N. F.: *Guidelines to ontology development* www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noymcguinness.pdf (Available in December 2005)

[14] Nwana H.S., Ndumu D.T.: A Perspective on Software Agents Research, In: *The Knowledge Engineering Review*, Vol 14, No 2, pp 1-18, 1999. http://citeseer.nj.nec.com/nwana99perspective.html

[15] O´Brien P.D., Nicol R.C.: FIPA – Towards a Standard for Software Agents, *BT Technology journal*, Volume 16, Number 3, 51-59, DOI: 10.1023/A:1009621729979

[16] Palensky P.: *Distributed Reactive Energy Management*, PhD Thesis, Technical University Wien, Austria, 2001

[17] Schreiber G., Akkermans H., Anjewierden A., de Hoog R., Sgadbolt N., Van de Velde W., Wielinga B., *Knowledge Engineering and Management, The CommonKADS Methodology*, The MIT Press, 2002

[18] Staab, S. H., Schunurr, R. S., Sure Y.: Knowledge processes and ontologies, IEEE Inteligent Systems, Special Issues on *Knowledge Management* 16(1), 2001, pp. 26-34

[19] Sycara K.P.: *Multi Agent systems*, http://www.aaai.org/Resources/Papers/AIMag19-02-007.pdf (accessed in December 2005)

[20] Tauber J.: *What is RDF*, 2006, www.xml.com/pub/a/2001/01/24/rdf.htnl

[21] Ushold M., King. M., Moralee S., Zorgios Y., *The Enterprise Ontology*, 1995, available at http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html, (Available in December 2005)

[22] Wooldridge M.: *An Introduction to Multi Agent Systems*, John Wiley & Sons, 2002

Brain Theory Applied

Marek BUNDZEL[1]

Abstract. Jeff Hawkins has explained a memory-prediction theory of brain function in 2004. Several of the concepts described in the theory are applied here in a computer vision system for a mobile robot application. The aim was to produce a system enabling a mobile robot to explore its environment and recognize different types of objects without human supervision. The system presented here works with time ordered sequences of images – dynamic images – instead of static images. The structure of the proposed system and the algorithms involved are explained. Brief survey of the existing algorithms applicable in the system is provided and future applications are outlined. Problems considering changes movements of the sensing system are listed and a solution is proposed. The proposed system was tested on a sequence of images recorded by two parallel cameras moving in a real world environment.

1 Introduction

Intelligent robots as commonly depicted in science fiction are able to interact with their environment, with people and to perform various human like actions. Often it seems that we are only one step away from building such machines. However, despite of a long research in the field it is not the case. We may be able to build highly sophisticated robotic bodies or, theoretically, a complex intelligent system controlling the actions of a robot but so far we have failed to make the robot perform some tasks we people consider very basic. For example, our robots cannot see as we do. We did not build a robust, multipurpose system giving a robot the ability to visually recognize objects in its environment - yet. Humans (indeed many animals) do it easily thanks to their highly developed brains. Learning what the environment consists of is the first step in the development of an intelligent behavior.

Jeff Hawkins has explained a memory-prediction theory of brain function in 2004 [1]. This theory was among the first that provided unified a unified basis for thinking about the adaptive control of complex behavior and it is in the

[1] TU Košice, FEI, KKUI, Letná 9, Košice, 04001, Slovakia, E-mail: marek.bundzel@tuke.sk

focal point of this chapter. Several of the concepts described in the theory are applied here in a computer vision system for a mobile robot application. The aim was to produce a system enabling a mobile robot to explore its environment and recognize different types of objects without human supervision.

The training process of the system is somewhat similar to the way a child learns about the world. The child sees things and learns about the existence of various objects alone. No adult trains a child to see. It can recognize the objects again even though it does not know their names or function. Later the adult can tell the child names of some objects of interest so they can be referred to in communication. The child does not learn about all objects and all their alternative appearances and views at once but rather gradually increases its knowledge by coming into contact with different environments. First, the system collects visual data recorded at a steady framerate while moving around various objects. Unsupervised learning is applied to identify entities comprising the environment. Human operator can assign names to the entities found. The possible advantage of such human - machine interaction is that it may be less demanding than creating extensive training sets describing all objects of interest. It can be also considered more natural with respect to the human operator. On the other hand there are no direct means as how to attract the attention of the system to particular objects and therefore the identified objects can differ from what the operator would like to obtain. The criterion of the training is the frequency of occurrence of spatial-temporal patterns. Therefore anything what appears in the sensory input frequently can be isolated as an object. The unsupervised learning mechanism has some features of the memory-prediction theory of brain function.

The system presented here works with time ordered sequences of images – dynamic images – instead of static images. It utilizes a tree structure of connected computational nodes similar to Hierarchical Temporal Memory [2] in many aspects. Each computational node performs the same operations and works in two modes: training and recognition. First, spatial structure of the data is discovered by means of clustering algorithm applied to smaller portions of the input data. Then temporal structure is discovered by application of a temporal data mining algorithm identifying frequent sequences of spatial features.

2 Memory-prediction Theory of Brain Function

The memory-prediction theory of brain function was created by Jeff Hawkins and described in the book On Intelligence: How a New Understanding of the Brain will Lead to the Creation of Truly Intelligent Machines [1]. The theory

gives possible explanations to the role of the mammalian neocortex and its associations with the hippocampus and the thalamus in matching sensory inputs to stored memory patterns and how this process leads to predictions of the future sensory inputs. The very basic idea is that the brain is a mechanism predicting the future. Hierarchical regions of the brain predict their future input sequences. These predictions do not need to concern distant future. For example, we predict continuation of a familiar song we are listening to, position of a ball we intend to catch or a sensation of our foot touching the next step while walking down stairs. But brain is also capable to make more complex and longer predictions if it is of some use to the organism.

The theory is motivated by the observed fact that the neocortex is remarkably uniform in appearance and in structure. Principally the same structures are used for a wide range of behaviors available to mammals. If necessary the regions of the neocortex normally used for one function can learn to perform different task. Adults who are born deaf process visual information in regions that normally become auditory regions. Blind adults use what is normally a visual cortex to read braille although since braille involves touch one could expect it to primarily activate touch regions. Several assumptions can be made:

- patterns from different senses are equivalent inside the brain
- the same biological structures are used to process the sensory inputs
- a single principle (a feedback/recall loop) underlies processing of the patterns

The basic processing principle is hypothesized to be a feedback/recall loop which involves both cortical and extra-cortical participation. Although certain brain structures are identified as participants in the core "algorithm" of prediction-from-memory, these details are less important than the set of principles that are proposed as basis for all high-level cognitive processing.

Time plays important role in functioning of a brain. Patterns coming from different senses are structured in both space and time. For example, most of the tactile information we get makes no sense to us until it is structured in time. We are usually not able to recognize surface or an object from a single touch. We slide our hands over it and perceive a sequence of patterns. Vision also relies on temporal patterns although it is less obvious. The patterns entering our eyes are constantly changing over time. About three times every second the eye makes a saccadic movement. In real life we constantly move our head and body, the environment itself is in motion.

As such, the brain is a feed forward hierarchical state machine with special properties that enable it to learn. The state machine actually controls the

behavior of the organism. Since it is a feed forward state machine, the organism responds to future events predicted from past data.

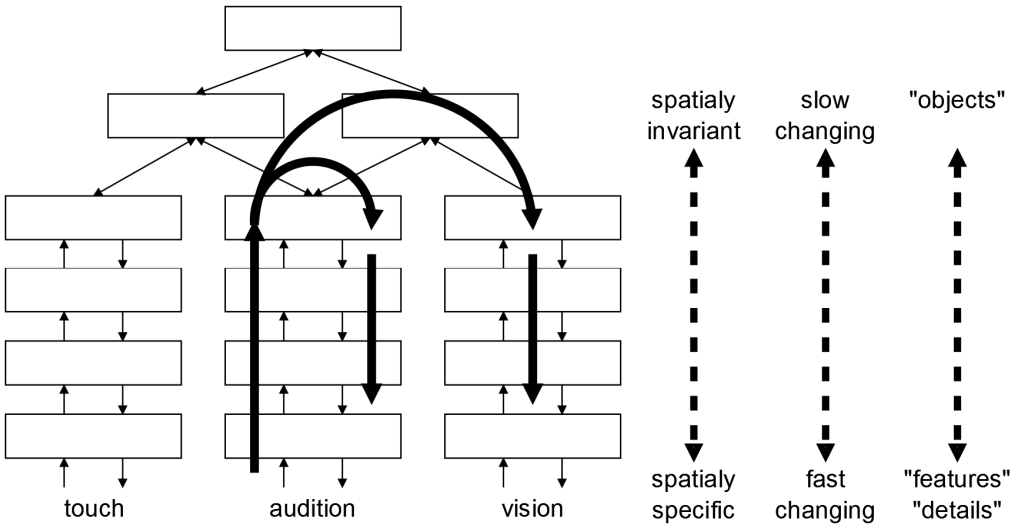


Fig. 1. Information flows up and down sensory hierarchies to form predictions and create unified sensory experience. Invariant representations are formed.

The cortex is built as a hierarchy of six layers. What Hawkins considers one of the most important concepts in his book is that the "cortex's hierarchical structure stores a model of the hierarchical structure of the world". Every object in the world is composed of a collection of smaller objects. Most objects are part of larger objects thus forming a nested structure. An object is characterized by a set of subobjects (features) which appear constantly together. Figure 1. shows a schematic view of the cortical hierarchy.

Considering the process of vision, bottom-up information starts as low-level retinal signals (indicating the presence of simple visual elements and contrasts). At higher levels of the hierarchy, increasingly meaningful information is extracted, regarding the presence of lines, regions, motions, etc. Even further up the hierarchy, activity corresponds to the presence of specific objects - and then to behaviors of these objects. Top-down information fills in details about the recognized objects, and also about their expected behavior as time progresses.

The hierarchy is capable of memorizing frequently observed sequences of patterns and developing invariant representations. The lowest level of the hierarchy recognizes sequences of the raw sensory input and ascending in the hierarchy each level recognizes sequences of the sequences recognized on the level below it. The information on the lower levels of the hierarchy is fast

changing and slowly changing on the upper levels. Representations on the lower levels are spatially specific and become spatially invariant on the upper levels. Higher levels of the cortical hierarchy predict the future on a longer time scale, or over a wider range of sensory input. Based on the memory-prediction theory Hawkins made several predictions regarding existence of cells with specific functions:

- "Anticipatory cells" are presumably distributed in all areas of cortex and fire in anticipation of a sensory event. In primary sensory cortex the anticipatory cells should be found in or near the lowest layer e.g. in the case of vision at a precise location in the visual field. Hawkins predicts that when the features of a visual scene are known in a memory, anticipatory cells should fire before the actual objects are seen in the scene.

- "Name cells" presumably exist in all regions of cortex. Learned sequences of firings comprise a representation of temporally constant invariants. Hawkins calls the cells which fire in this sequence "name cells". Hawkins suggests that these name cells are in layer 2, physically adjacent to layer 1. "Name cells" should remain on during a learned sequence. By definition, a temporally constant invariant will be active during a learned sequence. Hawkins posits that these cells will remain active for the duration of the learned sequence.

- "Exception cells" should remain off during a learned sequence. Hawkins' novel prediction is that certain cells are inhibited during a learned sequence. A class of cells in layers 2 and 3 should not fire during a learned sequence, the axons of these "exception cells" should fire only if a local prediction is failing. This prevents flooding the brain with the usual sensations, leaving only exceptions for post-processing. "Exception cells" should propagate unanticipated events. If an unusual event occurs (the learned sequence fails), the "exception cells" should fire, propagating up the cortical hierarchy to the hippocampus, the repository of new memories.
- "Aha! cells" should trigger predictive activity. Hawkins predicts a cascade of predictions, when recognition occurs, propagating down the cortical column (with each saccade of the eye over a learned scene, for example).

The memory-prediction theory claims a common algorithm is employed by all regions in the neocortex. The theory has given rise to a number of software

models aiming to simulate this common algorithm using a hierarchical memory structure. These include early model [3] that uses Bayesian networks and which made foundation for later models like Hierarchical Temporal Memory (HTM) [2] or open source project Neocortex by Saulius Garalevicius [4].

2.1 Hierarchical Temporal Memory

HTM is a machine learning model developed by Jeff Hawkins and Dileep George of Numenta, Inc. that models some of the structural and algorithmic properties of the neocortex using an approach somewhat similar to Bayesian networks.

An HTM network is a tree-shaped hierarchy of levels which are composed of computational nodes. More nodes are in the lower levels and fewer nodes are in the higher levels of the hierarchy. Each node performs the same functions as the entire HTM does which is "discovering and inferring causes" [2]. Technically, a computational node works in two modes which will be called a *training mode* and *recognition mode*. In the training mode the computational node first performs grouping of spatial and then grouping of temporal patterns. Grouping of spatial patterns is done by a clustering algorithm. Spatial patterns are assigned to groups that are fewer in number than the possible patterns, so resolution in space is lost in each node. The assignment of the patterns to groups is based on their spatial similarity. A mechanism must be provided to determine the probability of a novel input belonging to each of the groups. Grouping of temporal patterns works on discretized (spatially quantized) inputs i.e. indices of the groups to which the inputs most likely belong to are passed to the temporal grouping algorithm. The temporal grouping algorithm receives discretized inputs, one following another. Despite emphasizing the importance of finding and using frequent sequences in [1] and [2], it appears that HTM, as initially implemented and published on the Numenta's website, stores only the information on spatial patterns that appear frequently together and discards the sequential information. This data structure is usually referred to as a frequent itemset, e.g. [10]. Later HTM-based system using a sequence memory is described in [5]. In [5], a frequent sequence means a subsequence frequently occurring in a longer sequence. This is also known as a frequent episode, e.g. [13]. The length of the stored frequent temporal patterns can be fixed or variable, depending on the algorithms used and the user settings. Frequently observed patterns (frequent itemsets) are stored into separate groups based on how likely they are to follow each other in the training sequence(s). Each group of temporal patterns represents a single *cause* in [2] terminology or *name* in [1]. In recognition mode is the node confronted with inputs one following another. Membership of each input to the spatial groups is determined. Discretized input

is written on a top of the sequence of previous discretized inputs. This sequence of certain length is compared to the stored groups of temporal patterns. The set of probabilities assigned to the temporal groups is a node's *belief* about the input pattern. This also represents the output of the node, which is passed up the hierarchy.

Sensory data comes into the bottom level nodes. Each node sees a portion of the sensory data. The bottom level nodes output the generated pattern and enable the next level of the hierarchy to be trained and so on. The top level has a single node that stores the most general names which determine, or are determined by, smaller concepts in the lower levels which are more restricted in time and space. A node in each level interprets information coming in from its child nodes in the lower level as probabilities of the names it has in memory. Several names are stored in each level. In a more general scheme, the group's probability value can be sent to any node(s) in any level(s), but the connections between the nodes are still fixed. Each node outputs probability values for all known groups to the input of other nodes. The higher level node combines this output with the output from other child nodes thus forming its own input pattern. Since resolution in space and time is lost in each node as described above, beliefs formed by higher-level nodes represent an even larger range of space and time. This is meant to reflect the organization of the physical world as it is perceived by human brain. Larger concepts (e.g. causes, actions, and objects) are viewed by humans to change more slowly and consist of smaller concepts that change more quickly. Hawkins postulates that brains evolved this type of hierarchy to match, predict, and affect the organization of the external world.

In the Numenta's implementation of HTM, the output of the HTM's top node is matched with a name defined in a training set using supervised learning, for example, a Support Vector Machine. The initial HTM did not use feedback and predictions

3 Description of the Proposed System

3.1 Functions and Structure

Similarly to HTM, the proposed system is a hierarchy of computational nodes, grouped into layers. A layer is a two dimensional rectangular grid of nodes. A node N is identified by indices l, x, y (l is the index of the node's layer, x, y are the node's coordinates within the layer). Sensory data (either raw or preprocessed) forms the bottom layer's input matrix. The sensory data is image data from a single or two parallel color cameras, though only grayscale images

were used here. The preprocessing can include any filtering or image processing algorithm which will be considered beneficial for the application.

The receptive field of a node is a rectangular portion of its layer's input matrix, defined by width and height. The receptive fields of the nodes within a layer do not overlap and together they cover the input matrix. The receptive field of a node in the bottom layer in the stereoscopic setup is formed as shown in Figure 2.

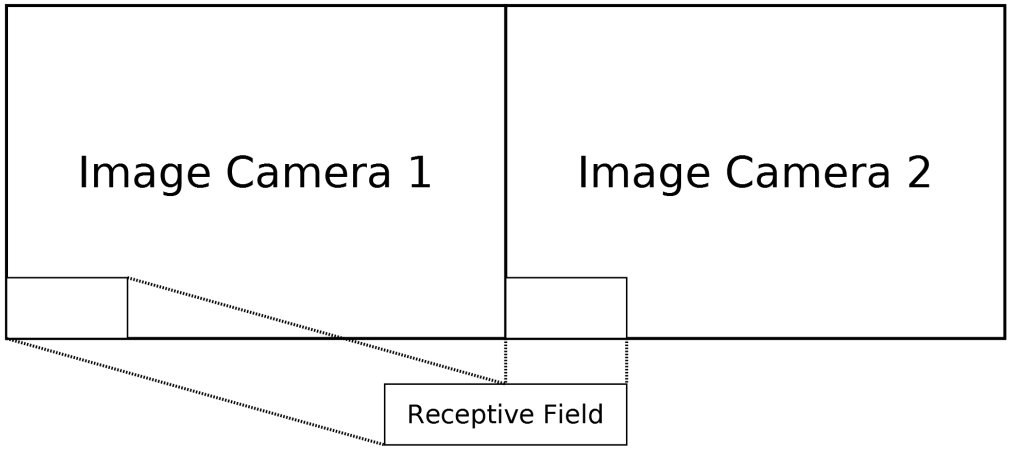


Fig. 2. Forming a receptive field of a node in the bottom layer in the stereoscopic setup - example

The stimuli in the receptive field of a node at time t form a vector $\overrightarrow{RF}_{l,x,y,t}$. Ordering of the elements of the portion of the layer's input matrix corresponding to the receptive field of a node into a vector is arbitrary but must remain constant. Output of the nodes of a layer forms the input matrix for the layer above. The top layer contains a single node. Output of the top node represents the output of the system. An example of the process is given in Figure 3.

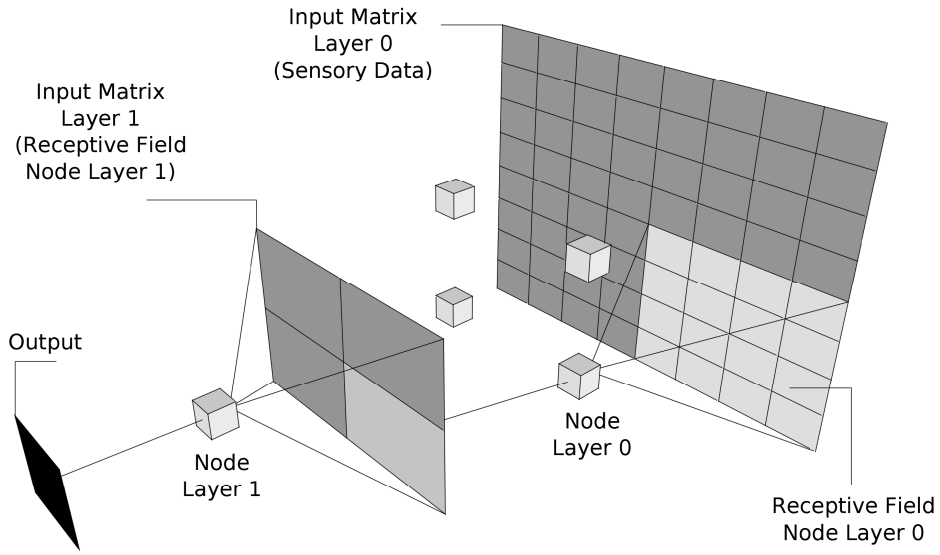


Fig. 3. Example, two layer hierarchy of nodes in one time step. Layer 1 contains a single node therefore the input matrix of the Layer 1 and the receptive field of the node in Layer 1 are identical.

A node operates in training and recognition modes. Training of a node is performed in two stages. The node performs spatial grouping of the training input patterns appearing in its receptive field by means of an algorithm for cluster analysis (clustering). The number of the identified spatial groups (clusters) reflects the structural complexity of the input data. The parameters of the spatial grouping algorithm of the nodes in separate layers are likely to require different settings. The spatial grouping algorithm must provide a mechanism for categorization of a novel input.

K-means clustering [6] was used in this work. The similarity measure was Euclidean distance. The training patterns for the cluster analysis in a node ***N*** are represented by a set $\left\{\overrightarrow{RF}_{t_0}, \overrightarrow{RF}_{t_0+1}, \ldots, \overrightarrow{RF}_{t_{end}}\right\}$. For example, if ***N*** is in the bottom layer, the training set will contain data representing the patterns which were appearing over time in the portion of the image data covered by the receptive field of ***N.*** The algorithm produces a set of k centroids $\boldsymbol{C} = \left\{\vec{C}_0, \vec{C}_1, \ldots, \vec{C}_{k-1}\right\}$, where k is set by the user. The centroids are vectors with the same number of elements as the receptive field vector of the node.

After the spatial groups are identified, the node processes the training patterns $\left\{\overrightarrow{RF}_{t_0}, \overrightarrow{RF}_{t_0+1}, \ldots, \overrightarrow{RF}_{t_{end}}\right\}$ ordered in time, starting with the oldest. Each training pattern is assigned to exactly one spatial group. In this work, each training pattern $\overrightarrow{RF}_t$ is assigned to the spatial group that has the closest centroid

in terms of Euclidean distance. The index of the winning spatial group $w_t, w \in \{0, \ldots, k\}$ is appended to a time ordered list $\boldsymbol{S}$ if $w_t \neq w_{t-1}$. The node ignores repeating states both in training and recognition modes for the reasons explained in Section 3.3. The time ordered list of indices (a sequence of indices) $\boldsymbol{S}$ represents the training data for a Temporal Data Mining algorithm searching for frequent episodes within $\boldsymbol{S}$. w_t represents the state of the receptive field of the node in time *t* and $\boldsymbol{S}$ represents the recording of the transitions between the states. The Temporal Data Mining algorithm used in this work is described in [7]. It is based on the frequent episode discovery framework [13]. It searches for frequent episodes with variable length. The frequent episodes identified by $\boldsymbol{N}$ are stored in a list $\boldsymbol{E}$ of lists $\{\boldsymbol{E_0}, \ldots, \boldsymbol{E_{Ne-1}}\}$, where *Ne* is the number of the identified frequent episodes. The user determines the minimal length of the frequent episode to be stored. It is ensured that the shorter episodes are not contained in the longer episodes because it would create undesired ambiguity.

Operation of $\boldsymbol{N}$ in recognition mode is divided into two consecutive stages. First, a novel input $\overrightarrow{RF_t}$ is categorized into one of the spatial groups identified in the training process $\overrightarrow{RF_t} \rightarrow w_t$. If $w_t \neq w_{t-1}$, w_t is appended to the list $\boldsymbol{BS}$ (the *buffer stack*) and the oldest item of $\boldsymbol{BS}$ is deleted. Constant length of $\boldsymbol{BS}$ is thus maintained. $\boldsymbol{BS}$ can be seen as a short term memory because it records the recent changes of states of the receptive field of the node. The length of $\boldsymbol{BS}$ is defined by the user. The elements of $\boldsymbol{BS}$ are initialized to -1 at the start of the algorithm. -1 does not appear in the stored frequent episodes therefore $\boldsymbol{BS}$ cannot be found in any of them before it is filled with valid values after start or after reset.

Second, in the given time step, the node tries to find which of the frequent episodes stored in $\boldsymbol{E}$ contains $\boldsymbol{BS}$ (in direct and reverse order). The purpose is to recognize whether the sequence of the recent changes in the receptive field has been frequently observed before. The output of $\boldsymbol{N}$ in time *t* is a binary vector $\vec{O}_t$. The elements of $\vec{O}_t$ correspond to the stored frequent episodes. If $\boldsymbol{E}_i$ contains $\boldsymbol{BS}$ in the given time step, the *i*-th element of $\vec{O}_t$ is set to 1 otherwise it is set to 0. If $\boldsymbol{E}_i$ is shorter than $\boldsymbol{BS}$ the corresponding number of older items in $\boldsymbol{BS}$ is ignored and the matching is performed with the shortened buffer stack.

There are several conditions modifying the behavior of a node in recognition mode. The node can be active (flag $A = 1$) or inactive (flag $A = 0$), with nodes initially starting with $A = 1$. The conditions are checked in each time step. If $w_t = w_{t-1}$ the counter T_{idle} is incremented by 1. $\vec{O}_t$ will be equal to $\vec{O}_{t-1}$. If T_{idle} exceeds a user defined timeout constant T_{out}, A is set to 0, and the elements of $\vec{O}_t$ are set to 0. The node remains inactive until there is a significant change in its input ($w_t \neq w_{t-1}$). If that happens, the node is reset: A is

set to 1, T_{idle} is set to 0 and the elements of ***BS*** are set to -1. This is to avoid unrelated events lying further apart in time being considered one event by a node. For example, if only a portion of the robot's vision field is changing the nodes processing the unchanging portion will turn inactive. This also reduces the computational load. Table 1 summarizes the algorithms used.

Calculation of the output $\vec{O}_t$ of a node in recognition mode in one time step can be seen in pseudocode as follows:

{ w_{t-1}, ***BS***, T_{out}, A have assigned values}
$w_{t-1} \leftarrow$ Categorize($\overrightarrow{RF}_t$,***C***) {Categorize current input using the centroids}
if $w_t \neq w_{t-1}$ **then**
    if $A = 0$ **then** $A \leftarrow 1$
    end if
    $T_{idle} \leftarrow 1$
    Push(***BS***, w_t) {Append w_t to ***BS***, delete oldest element of ***BS***}
    $\vec{O}_t \leftarrow$ FindInEpisodes(***BS***, ***E***) {Find which episodes contain ***BS***}
    $w_{t-1} \leftarrow w_t$
    $\vec{O}_{t-1} \leftarrow \vec{O}_t$
else
    if $A = 0$ **then** $\vec{O}_t \leftarrow \vec{O}_{t-1}$
    else $T_{idle} \leftarrow T_{idle} + 1$
        if $T_{idle} > T_{out}$ **then**
            $A \leftarrow 0$
            SetAllElements(***BS***, -1) {Set all elements of ***BS*** to -1}
            SetAllElements $\vec{O}_t$,0) {Set all elements of $\vec{O}_t$ to 0}
            $\vec{O}_{t-1} \leftarrow \vec{O}_t$
        else
            $\vec{O}_t \leftarrow \vec{O}_{t-1}$
        end if
    end if
end if

When ***BS*** is found to be part of a stored frequent episode, prediction of the future inputs already resides in the remaining part of the frequent episode. The prediction can be for example used to reduce ambiguity by categorization of the incoming input if it is noisy. This feature was not used here, however.

The user defines the structure of the hierarchy (i.e. number of layers, dimensions of receptive fields for nodes in each layer) and the setting of training algorithms for each layer. The layers can be trained simultaneously, but

Table 1. Algorithms used during training (T) and recognition (R)

Algorithm	Mode	Comments
Data preprocessing	T,R	e.g. Gabor filtering, normalization etc.
Clustering	T	Parameters set for each layer separately
Categorization	T,R	$\overrightarrow{RF}_t \rightarrow w_t$
Temporal data mining	T	Parameters set for each layer separately
Sequence matching	T,R	finding ***BS*** in ***E***
Name assignment	T	Naming the objects found

it is more suitable to train the layers consecutively, starting with the bottom layer. In this way, it is ensured that a layer about to be trained is getting meaningful input. In order to simplify the learning process and to increase generality, a modified training approach can be used. Instead of training nodes of a layer separately a *master node* $\boldsymbol{N}_{master}$ is trained using the data from the receptive fields of all nodes in the layer. The training set of the master node is:

$$\begin{aligned}&\{\overrightarrow{RF}_{L,0,0,t_0}, \ldots, \overrightarrow{RF}_{L,0,0,t_{end}}, \ldots \\ &\ldots, \overrightarrow{RF}_{L,m-1,n-1,t_0}, \ldots, \overrightarrow{RF}_{L,m-1,n-1,t_{end}}\}\end{aligned} \tag{1}$$

where L is the index of the layer of m by n nodes being trained. This implies that the receptive fields of all nodes must have the same dimensions. When $\boldsymbol{N}_{master}$ is trained, a copy of it replaces nodes at all positions within the layer:

$$\begin{aligned}&\forall i = 0, 1, \ldots, m-1;\ j = 0, 1, \ldots, n-1: \\ &\boldsymbol{N}_{L,i,j} \leftarrow \boldsymbol{N}_{master}\end{aligned} \tag{2}$$

Based on the assumption that objects can potentially appear in any part of the image although they were not recorded that way in the training images, the advantage is that each node will be able to recognize all objects identified in the input data. In this work, the modified training approach was applied on each layer of the hierarchy.

After the unsupervised learning of the system is completed, the operator assigns names to the objects the system has identified. The output of the top layer (node) of the hierarchy is a binary vector. All images from the training set for which a particular element of the binary vector is non-zero are grouped and presented to the operator. The operator decides whether the group contains a majority of pictures of an object of interest. This is done for all elements of the output vector. When the system is tested on novel visual data, ideally, the

elements of the output vector should respond to the same type of objects as in the training set.

3.2 Brief Survey of Applicable Algorithms

This section is aimed to provide insight into a scale of existing algorithms which can be applied in a system described above. The algorithms in question are related to the following operations: clustering, temporal data mining and sequence matching.

Spatial structure discovery algorithms - clustering algorithms - represent a deeply researched domain. Cluster analysis, primitive exploration with little or no prior knowledge, consists of research developed across a wide variety of communities. There are many existing well documented algorithms. A survey of clustering algorithms [8] provides information on categorization of the algorithms and illustrates their applications on some datasets.

Temporal data mining on the other hand has raised much less awareness. Since temporal data mining brings together techniques from different fields such as statistics, machine learning and databases, the literature is scattered among many different sources. Surveys on temporal data mining techniques can be found in [9] and [10]. The classical time series analysis has quite a long history of more than fifty years. Temporal data mining is of a more recent origin with somewhat different constraints and objectives. One main difference lies in the size and nature of data sets and the manner in which the data is collected. Often temporal data mining methods must be capable of analyzing data sets that are prohibitively large for conventional time series modeling techniques to handle efficiently. Moreover, the sequences may be nominal-valued or symbolic rather than being real or complex-valued. The typical applications include mining customer transaction logs for estimating customer buying patterns.

The second major difference between temporal data mining and classical time series analysis lies in the kind of information which is searched for. The scope of temporal data mining extends beyond the standard forecast or control applications. Unearthing of useful (and often unexpected) trends or patterns in the data may be of greater relevance. The techniques particularly relevant to the problems described here are the framework of sequential pattern discovery [12] and the frequent episode discovery framework [13] for mining of frequent sequential patterns.

The framework of sequential pattern discovery [12] is essentially an extension (by incorporation of temporal ordering information into the patterns being discovered) of the original association rule mining framework proposed for a database of unordered transaction records [11] which is known as the

Apriori algorithm. The Apriori algorithm exploits very simple but very powerful principle: if ***i*** and ***j*** are itemsets such that ***j*** is a subset of ***i***, then the support of ***j*** is greater than or equal to the support of ***i***. Thus, for an itemset to be frequent all its subsets must in turn be frequent as well.

A second class of approaches to discovering temporal patterns in sequential data is the frequent episode discovery framework [13]. In the sequential patterns framework, we are given a collection of sequences and the task is to discover (ordered) sequences of items (i.e. sequential patterns) that occur in sufficiently many of those sequences. In the frequent episodes framework, the data are given in a single long sequence and the task is to unearth temporal patterns (called episodes) that occur sufficiently often along that sequence. [13] applied frequent episode discovery for analyzing alarm streams in a telecommunication network. The algorithm used in the system described above [7] uses the frequent episode discovery framework [13].

Searching for sequences in large databases is another important task related to temporal data mining performed by the proposed system. Database of the frequent episodes may potentially grow very large. The problem is concerned with efficiently locating subsequences (***BS*** in this case) often referred to as queries in large archives of sequences (***E*** in this case). Query-based searches have been extensively studied in language and automata theory. While the problem of efficiently locating exact matches of substrings is well solved, the situation is quite different when looking for approximate matches [14]. Because in the real world sequences of events are rarely identical, it is approximate matching that we are more interested in.

As was mentioned, because the sequences rarely occur the same way twice, it is possible that the proposed system will identify similar but not identical frequent episodes. Application of sequence clustering algorithm can be considered. There are a variety of methods for clustering sequences [10]. At one end of the spectrum there are model-based sequence clustering methods. The other broad class in sequence clustering uses pattern alignment-based scoring or similarity measures to compare sequences.

3.3 Domain Related Problems and Comparison

The problems of the application of the described system in a mobile robot are largely related to the balance which must be achieved between the robot's velocity, the scanning frequency, the dimensions of the receptive fields of the nodes in the bottom layer and the measure of the discretization of the input space into spatial groups (the number of spatial groups). In this work, the parameters were set based on logical assumption and/or trial and error.

Let us assume the robot is in forward movement. The objects in its vision field appear larger as the robot approaches and leave the vision field sideways as the robot passes. Let us assume there is an object whose features are consecutively assigned to three different spatial groups A, B and C as it moves in the vision field. Assuming a constant framerate and image processing, the observed sequence may be ...→ A → A → A → B → B → B → C → C → C → ..., or ...→ A → B → C → ... or ...→ A → C → ..., depending on the velocity of the robot. However, it is desirable that the object be characterized by a constant temporal pattern within the range of the robot's velocity.

To minimize the influence of the changing velocity, the nodes ignore repeating states. The disadvantage is that objects distinguished by variable number of repeating features will be considered a single object type. A lower velocity, higher scanning frequency and rougher discretization of the input space increases the frequency of the repeating states in the observed sequence and vice versa. To ensure optimal performance these values must be in balance.

The number of the spatial groups to be identified by a node is relative to the structural complexity of the spatial data. It is the value to start the tuning with. The scanning frequency (including processing of the images) is largely limited by the computational capacity of the control computer. During training, the robot first collects the image data without processing them so the scanning frequency can be higher and the robot can move faster. It should be taken into consideration that the robot's velocity will likely have to be reduced when the system enters recognition mode due to the reduction of the scanning frequency. The velocity of the robot can be easily changed but cannot be too low for meaningful operation.

Ideally, there will be frequently repeating states in the sequence observed by a node regardless of the robot's velocity. The robustness of the system will be higher assuming that no important features are being missed. This problem is most critical with the nodes in the bottom layer, because the input patterns are changing slower in the higher levels of the hierarchy. Setting the dimensions of the receptive fields of the nodes in the bottom layer influences how long an object will be sensed by a node. If the dimensions are too small given the velocity, the scanning frequency and the discretization, it is more likely that two unrelated objects will appear in the receptive field of a node in two consecutive time steps. This means that the identified frequent episodes (if any) would include features of different objects instead of including different features or positions of a single object. The resolution of the recognition may deteriorate below an acceptable level. On the other hand, if the receptive fields of the nodes in the bottom layer are too large, it is more likely that multiple objects will be sensed simultaneously by a node. In every time step, the stimulus in the receptive field is assigned to a single spatial group. One of the sensed objects

will thus become dominant. However, in the following time steps, other objects in the receptive field may become dominant and the observed sequence will lose meaning.

If there are multiple objects in the vision field of a robot during operation, the system may separate a single object, a group of objects as a single object, may not recognize the objects (all elements of the output vector are 0) or may misinterpret the situation (an element of the output vector will become active which is usually active in the presence of a different object). Note that any frequent visual spatial-temporal pattern may be identified as a separate object during training, and not necessarily as a human would do it. No mechanism for covert attention was implemented to the system at this point; the system evaluates the vision field as a whole.

The proposed system is closely related to other models based on the memory-prediction framework ([2], [3], [4] and [5]). It has the same internal structure as HTM. The sequence memory system published by Numenta in 2009 [5] uses a mechanism to store and recall sequences in an HTM setting. The Temporal Data Mining technique used in [5] aims to map closely their proposed biological sequence memory mechanism. In contrast to the proposed system, it enables simultaneous learning and recall. The system proposed here could not utilize this feature now because when a node identifies a new sequence, the dimension of its output vector increases and retraining of the nodes in the layers above is necessary. Neither the proposed system nor [5] provide means of storing duration of sequence elements.

The proposed system can be considered an HTM with a sequence memory. The products published by Numenta and the proposed system use different algorithms for cluster analysis and Temporal Data Mining, but the main difference is that the proposed system is specialized for a real time computer vision application on a mobile robot. This required implementation of a mechanism for minimizing the influence of the robot's changing velocity on storing and recalling the frequent episodes and usage of relatively fast methods. In contrast to HTM, the system does not supervise learning on the top level. The human-machine interface for labeling the categories of the identified objects is used instead. In other words, the proposed system is allowed to isolate objects on its own. The supervision has a form of communication (although primitive at the moment) instead of typical supervised learning, when observations must be assigned to predefined categories.

4 Experiments

4.1 Experimental Setup

The experiments were performed on image data recorded with two identical cameras installed forward facing side by side on a mount. Optical axes of the lenses were parallel to each other (50mm apart) and to the floor (100mm above). Each camera covered 61-degree horizontal and 48-degree vertical field of view. The vertical field of view touched the floor 230mm in front of the mount. The horizontal fields of view of the cameras started to overlap 43mm in front of the mount. The mount was designed for a small experimental robot encountering relatively small objects placed on the floor. The cameras would capture a full height of a human app. 4m in front of the mount.

The image data were recorded when moving the mount in a 1.7m x 2.1m arena. The arena was not intended to provide a visually empty environment but a semi realistic office environment. It was a part of office space limited by furniture (closet, drawer boxes etc.) and walls. There were three small objects placed in the arena: toy dog, toy turtle and toy car. The average speed of the mount forward movement was app. 50mm/s. The images were taken at 2fps and 320x240 pixels resolution. These values were chosen so that the controlling computer can process the incoming images online if Gabor filtering was used (Gabor filtering is relatively time consuming). The recorded data set contained together 1140 images from each camera. 60\% of the dataset was used for training and 40\% for validation.

There are two experiments presented. The first one was performed on single camera data and the later one on stereoscopic data. In the single camera experiment the system consisted of two layers of computational nodes (150 nodes on layer 0, 1 node on layer 1). Image data were converted to grayscale. Using of Gabor filtering did not improve the results of the system as expected and was not used. This is probably due to the fact that the image data were taken in visually stable environment and that the set of objects was the same in training and validation (only views of the objects changed, not the objects themselves). Dimensions of the receptive fields of nodes at layer 0 were 32x16 pixels. This relatively large receptive field was used so that objects would remain in the field of view of the node for several consecutive time steps and meaningful frequent sequences could be identified. Using 2fps framerate at given velocity caused rapid relative movement of the objects in the consecutive images. As most of the movement is along the horizontal axis the receptive field's width is larger than its height. K-means clustering based on Euclidian distance was used, categorizing into 80 clusters on layer 0. This corresponds to the relatively high variation of information captured by the relatively large crop.

Clustering on layer 1 required 20 clusters. The minimum support coefficient of the algorithm defined as the ratio of occurrences to the total number of events in the data was set to 0.003 on layer 0 and 0.01 on layer 1. The minimal length of sequences at layer 0 was set to 3, **BS** length to 3 and T_{out} = 10 (what corresponds to 5s at the given framerate, i.e. the node turns inactive if there is no significant change on its input for more than 5s). The minimal length of sequences at layer 1 was set to 1, **BS** length to 2 and T_{out} = 10.

In the stereoscopic experiment the system consisted of two layers of computational nodes (150 nodes on layer 0, 1 node on layer 1). Image data were converted to grayscale. Dimensions of the receptive fields at layer 0 were 32x16+32x16 pixels (crops from camera 1 and 2 merged). K-means clustering based on Euclidean distance was used, categorizing into 100 clusters on layer 0. This corresponds to the higher variation of stereoscopic information compared to the single camera information. Another problem arises here. The cameras must be precisely aligned so that there is information on the same object received from cameras 1 and 2 in the receptive field. Clustering on layer 1 required 20 clusters. The minimum support coefficient of the algorithm was set to 0.003 on layer 0 and 0.01 on layer 1. The minimal length of sequences at layer 0 was set to 3, **BS** length to 3 and T_{out} = 10. The minimal length of sequences at layer 1 was set to 1, **BS** length to 2 and T_{out} = 10.

4.2 Experimental results

It is difficult to evaluate the results of unsupervised learning. In this case the emphasis is on consistency of results on the training and on the testing set. This means that when operator assigns a name to certain outputs these outputs will become active for the same objects in the training and in the testing set.

In the single camera experiment together 40 frequent episodes were found on layer 0 (11 of length 2 - not used, 7 of length 3, 12 of length 4 and 10 of length 5). The sequences may be visualized as e.g. the sequence of appropriately post-processed cluster centroids. In the ideal case the centroids would represent different elementary shapes. Without Gabor filtering the centroids in our case were relatively uniform patches of various luminance meaning the changes in luminance are more important rather than changes of texture. That also means that changes in exposure would influence the behavior of the system negatively. Appropriate data preprocessing is needed to achieve robustness. Together 24 frequent episodes were found on layer 1 (20 of length 1 (cluster index), 3 of length 2 and 1 of length 3). Based on this the output vector had 24 elements. Variation of information on the output of layer 1 was lower which confirmed the expectation based on the memory-prediction theory. The sequence written by layer 1 on which the temporal data mining was performed

had 86 elements. This means that on 640 images of the training set layer 0 changed state 86 times. In the typical behavior certain state of layer 0 persisted for several timesteps and then it has changed. This is consistent with the fact that objects remain in the field of view for some time (unless there are sudden changes in robot movements).

The operator was confronted with groups of images for which the particular elements of the output vector were non-zero. In 95% of the cases there was a sole non-zero element in the output vector. In 100% of the cases there was at least one non-zero element in the output vector. This is because the frequent episodes of length 1 contained all categories (states) of layer 0. Based on the prevailing appearance of certain object in a group the operator assigned a name to the particular element of the output vector. 8 names were assigned together.

Table 2. Experimental results (* number of outputs responding to the object)

Monoscopic Experiment					
Object Name	**Count Train**	**Count Test**	**Corr. Train**	**Corr. Test**	**O.***
empty carpet	112	59	85%	81%	8
drawer box	38	15	79%	80%	1
car	71	42	89%	90%	4
white table	45	28	88%	89%	2
turtle	28	18	82%	83%	1
dark table	88	57	90%	79%	1
el. socket	49	21	71%	81%	1
sliding door	55	34	73%	79%	1
unidentified	198	182	-	-	5
Stereoscopic Experiment					
Object Name	**Count Train**	**Count Test**	**Corr. Train**	**Corr. Test**	**O.***
empty carpet	105	52	74%	77%	7
drawer box	33	17	76%	76%	1
car	50	35	86%	80%	3
white table	45	28	87%	82%	3
turtle	22	15	68%	80%	2
dark table	88	57	83%	75%	1
el. socket	47	19	74%	63%	1
sliding door	55	28	78%	71%	2
unidentified	239	205	-	-	3

Groups containing uncharacteristic mixture of objects were designated as "unidentified". "Unidentified" included 29% of the images of the training set and 40% of the images of the testing set.

Table 2. summarizes the experimental results. Table 2. states the object name, how many images of the training and testing sets were identified as given object, number of correct identifications and number of inputs responding to the presence of given object (at least one the outputs became non-zero in the presence of given object). Correctness of identification had to be performed by manual counting of the images of the group containing given object. E.g. for the "empty carpet" the correct image would not contain any toys or other identified objects but for "car" the image must contain at least a certain portion of the car and can contain any other object. Automatic evaluation was not possible as it was unclear which objects will the system identify beforehand. Figure 4. shows the typical examples of the objects found.

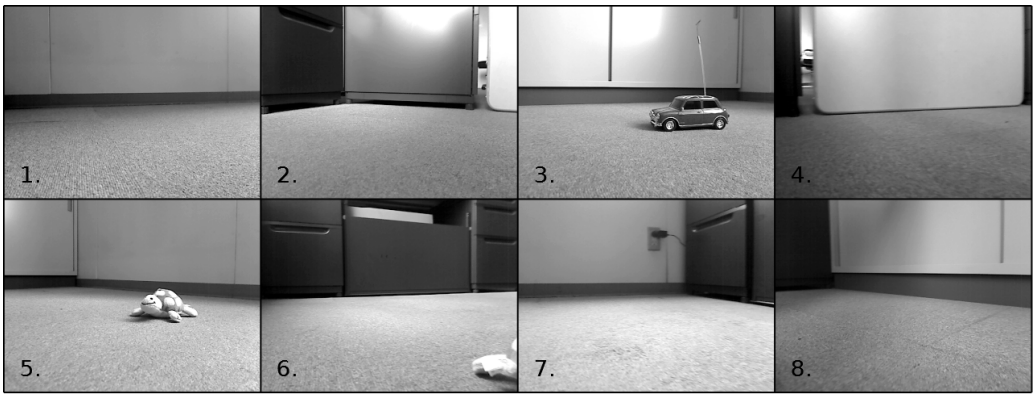


Fig. 4. Typical examples of the objects identified in the single camera experiment. 1. “empty carpet”, 2. “drawer box”, 3. “car”, 4. “white table”, 5. “turtle”, 6. “dark table”, 7. “corner with electric socket”, 8. “closet with sliding door”

In the stereoscopic camera experiment together 45 frequent episodes were found on layer 0 (19 of length 2 - not used, 12 of length 3, 8 of length 4 and 6 of length 5). Appearance of the centroids was very similar as in the single camera experiment. This is probably due to the fact that the information in the crops from cameras 1 and 2 do not significantly differ in most cases. Together 23 frequent episodes were found on layer 1 (20 of length 1 (cluster index) and 3 of length 2). Based on this the output vector had 23 elements. Overall the behavior was very similar to the single camera experiment. Therefore the operator searched for the same objects as in the single camera experiment so that comparison would be possible. "Unidentified" included 35% of the images of the training set and 45% of the images of the testing set. The experimental

results indicate that adding the information from the second camera increased confusion rather than resolution. It is difficult to assess whether this is to be attributed to the slightly different setting of the algorithms or to problems related to cameras alignment.

The experimental results indicate strong consistency between the results on the training and testing sets. If there was an object identified during training it is very likely that the same elements of the output vector will respond to the object also on the testing set. However, still a large portion of the sets in not identified despite of the presence of identified objects in the images.

5 Problems, Discussion and Future Work

One of the main problems of the system is a large number of user set parameters significantly influencing the functioning of the system. A human operator is needed to name the objects therefore trial and error approach is time consuming. Algorithms for automatic optimization of these parameters are to be developed.

Like HTM, the system can be modified for supervised learning. A supervised learning setup can provide more data to evaluate the behavior of the system and to develop methods for optimization of the user set parameters.

The algorithm matching **BS** with the frequent episodes **E** searches for absolute match. Discretization of the input space provides the only mean for generalization now. Algorithm searching for approximate matches should be employed. Also, predictions and feedback are to be implemented.

The brain presumably includes information on the organism's own movements when making predictions about future inputs. We propose Tout to be inversely proportional to the robots' velocity in the future, to avoid the system turning inactive if the velocity decreases or the robot stops. Incremental learning is an important feature of the system to be developed. Adding new frequent episodes in layer L increases dimensionality of the input matrix of the consecutive layer. Therefore the layers above L must be retrained, which requires storing the corresponding training set. The problem considers mainly higher levels of the hierarchy recognizing more complex objects. If the lower levels of the hierarchy are sufficiently trained the basic object comprising the world has been correctly identified. More complex objects are usually comprised from the same basic objects.

In addition to learning new sequences also forgetting should be considered (also discussed in [4]). An autonomous robot will presumably explore different environments and collect a large amount of knowledge. Matching a current sequence to all of the stored frequent episodes would become more and more demanding over time. We propose that frequently used sequences would be

kept active and less frequently used sequences would be transferred into storage or completely forgotten. In the case that a sequence is not recognized the episodes in storage should be checked first for a possible match and retrieved if necessary. In this way, the robot could keep active only those learned sequences which are relevant for the current environment.

There is no mechanism that enables the system to identify the position of a recognized object in the image at this point. The system usually fails if there are multiple objects in the scene. We plan to develop a mechanism for covert attention, presumably using feedback, to identify the location of an object, masking it and searching for other objects in the scene.

The performance of the system deteriorated slightly in the stereoscopic experiment. The causes of the problems mentioned in the stereoscopic experiment may be eliminated by using a hierarchy with more layers and integrating the information from separate cameras at higher levels of the hierarchy, not in layer 0 directly.

Strain on the human operator is high as are the time demands. More sophisticated means of human-system interaction should be developed, possibly using a mechanism for the grouping of similar images and presenting only model examples to the operator and/or highlighting the objects in the scene that have triggered the response.

6 Conclusion

A system for unsupervised identification of objects in image data recorded by moving cameras in real time using a novel combination of algorithms was presented here. The system has some features described in memory-prediction theory of brain function. A solution was proposed for the elimination of uncertainty linked with the variable speed of the robot.

The system represents an early attempt to make a machine that learns largely on its own and only needs occasional advice from the human operator. It is possible that this approach may be more suitable for creating intelligent multipurpose systems than trying to heavily supervise the learning process.

Acknowledgement. This work is supported by Japan Society for the Promotion of Science and Waseda University, Tokyo.

References

[1] Hawkins, J., and Blakeslee, S. (2004). *On Intelligence: How a New Understanding of the Brain will Lead to the Creation of Truly Intelligent Machines*, Times Books, ISBN 0-8050-7456-2

[2] Hawkins, J., and George, D. (2006). *Hierarchical Temporal Memory - Concepts, Theory and Terminology*, White Paper, Numenta Inc.

[3] George, D., and Hawkins, J., (2005). A Hierarchical Bayesian Model of Invariant Pattern Recognition in the Visual Cortex, *Proceedings of 2005 IEEE International Joint Conference on Neural Networks*, Vol. 3, 1812-1817.

[4] Garalevicius, S. J., (2007). *Memory-Prediction Framework for Pattern Recognition: Performance and Suitability of the Bayesian Model of Visual Cortex*, FLAIRS Conference, 92-97.

[5] Hawkins, J., George, D., and Niemasik, J., (2009). Sequence Memory for Prediction, Inference and Behaviour, *Phil. Trans. R. Soc. B*, Vol. 364, No. 1521, 1203-1209.

[6] MacQueen, J. B., (1967). Some Methods for classification and Analysis of Multivariate Observations, *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, 1:281-297.

[7] Patnaik, D., Sastry, P. S., and Unnikrishnan, K. P., (2008). Inferring Neuronal Network Connectivity from Spike Data: A Temporal Data Mining Approach, *Scientific Programming*, vol. 16, 49-77, ISSN: 1058-9244.

[8] Xu, R., and Wunsch II, D. (2005). Survey of Clustering Algorithms, *IEEE Transactions on Neural Networks*, Vol. 16, No. 3, 645-678.

[9] Han, J., Cheng, H., Xin, D., and Yan, X., (2007). Frequent Pattern Mining: Current Status and Future Directions, *Data Mining and Knowledge Discovery*, Vol. 14, No. 1.

[10] Laxman, S., and Sastry, P. S., (2006). A Survey of Temporal Data Mining, Sadhana, *Academy Proceedings in Engineering Sciences*, Vol. 31, Part 2, 173-198.

[11] Agrawal, R., Imielinski, T., and Swami, A., (1993). A Mining Association Rules Between Sets of Items in Large Databases, *Proceedings of ACM SIGMOD Conference on Management of Data*, 207-216.

[12] Agrawal, R., and Srikant, R., (1995). Mining Sequential Patterns, In *Proceedings of 11th Int. Conf. on Data Engineering*.

[13] Mannila, H., Toivonen, H., and Verkamo, A. I., (1997). Discovery of Frequent Episodes in Event Sequences, *Data Mining Knowledge Discovery* 1, 259-289.

[14] Wu, S., and Manber, U., (1992). Fast Text Searching Allowing Errors, *Commun. ACM* 35(10), 83-91

The Future of Search: Perspectives from IBM, Apple, Google and Others

Ján GONDOĽ[1]

Abstract. Information retrieval has been evolving rapidly: increasing computing power as well as software improvement leads to new processing-intensive applications difficult to implement previously. IBM's Watson supercomputer is one example of the possibilities. Natural language processing combined with powerful algorithms to extract possible answers to well-formulated questions from structured data (databases), semi-structured data (hypertext) and unstructured documents are very useful beyond the Jeopardy game. Medicine and legal profession are example areas. Apple's Siri application used in the iPhone 4S uses a voice recognition layer to communicate with humans in a more natural way (Watson's input was text-based). Google Goggles product relies on image recognition technology to make search even more ubiquitous. IBM's Watson, Apple's Siri and Google's Goggles can all perform search but none of them have typed keyword input as a starting point. The paper discusses the technologies employed, applications in real life and possible scenarios of development. For a Siri-like program (natural language input) combined with Watson-like back-end (reasoning engine) and Google-like extensive knowledge base the uses could be very interesting.

1 Introduction

In order to understand the current state of the art of search and the possible future development paths we should first have a look at the **technology landscape**. It is helpful to understand the trends in hardware, software, networking and other technologies and see how the new developments affect people's lifestyles and how this changes their interaction with information.

[1] Institute of Computer Science, Faculty of Philosophy and Science, Silesian University, Bezručovo nám. 13, 74601 Opava, Czech Republic. E-mail: gondol@gondol.sk

1.1 Developments in Hardware

First of all, **computing power available for a fixed cost keeps increasing** and it will likely take some time before the physical limits of silicone-based technology are reached. We are able to pack more and more transistors on an integrated circuit and according to many experts Moore's Law is likely to keep holding true for several more years. A transistor in modern chips is about **100 times smaller than a human red blood cell**, and the miniaturization will continue and make microelectronic components even finer.

Modern chips have enough power to work with high-quality audio and video (so processing high-definition multimedia is no longer an issue, even in mobile devices) and we'll be able to use the ever increasing performance for **much more sophisticated software applications** than before.

Plummeting costs of hardware devices can be welcome by consumers. Year 2012 marked the launch of Raspberry Pi, a general-purpose computer costing $25[2], capable of working with external USB-connected devices and enough performance to decode 1080p HDTV video in real time, while consuming about 3 watts of power[3]. The acquisition costs of laptops has been falling dramatically (the same cannot always be said about the support costs and TCO), and we have seen the advent of quite usable **tablets for less than $100** incl. shipping anywhere in the world from sites like DX.com (previously known as DealExtreme) and LightInTheBox.com. Powerful smartphones are now within reach of most consumers in developed countries and their adoption has been growing significantly. According to Nielsen Research (2012), "almost half (49.7%) of U.S. mobile subscribers now own smartphones, as of February 2012. (…) In February 2011, only 36 percent of mobile subscribers owned smartphones."

Miniaturization in microprocessor manufacturing has also led to energy efficiency: **smaller transistors consume less power**. This has important implications – multi-core processors in the mobile devices have become affordable and their traditional battery still works acceptably. Also, the desktop devices can be many times more powerful than the smartphones and the data centers can be both "greener" (i.e., more energy-efficient) and delivering more performance at the same time.

[2] There are two models of Raspberry Pi, one for $25 and the other one (with onboard Ethernet and two USB ports instead of one) for $35.

[3] It is interesting to note that Raspberry Pi was launched in the United Kingdom, exactly three decades after the legendary Sinclair ZX Spectrum computer from the same country.

Internet connection is so important that some countries (with Finland leading the way) declared it a legal right[4]. It makes sense to invest in the internet infrastructure – and, as a result, connectivity has been improving. Today, private individuals in Slovakia can afford **many times faster** internet link than some sizable organizations had less than 15 years ago[5]. Of course, there is still a digital divide between countryside and big cities and it is worrying that even in some cities in Slovakia decent broadband is still simply unavailable[6]. It is not just available bandwidth that has been improving. We should also mention **network latency** which significantly contributes to the user experience: with low latency mobile networks we can have a new class of real-time applications, previously impossible to implement satisfactorily. **Mobile broadband** is slowly becoming more and more available. Today it is possible to connect to a high-speed 3G network even in the mountains, e.g. some peaks in the High Tatras. Unfortunately, connection on a train or in a bus / car (even on some of the highest-traffic routes in Slovakia) is still rather sketchy.

Besides processing power and network connectivity we should also mention the trends in storage. In spite of the 2011 floods in Thailand which destroyed several manufacturing plants, resulting in increased costs of hard drives, the long-term trend is clear: magnetic **storage keeps getting cheaper**. Moreover, RAM as well as flash-based memory (used in solid-state drives) are also becoming cheaper and the SSDs (which are about to attack the $1/GB price point) are finally entering the mainstream. Fast random access to data (provided by the computer's RAM as well as the SSDs) and access to large quantities of data (provided by the hard drives) pave the way for sophisticated search technologies requiring a lot of storage and processing performance.

Consumer devices are now equipped with a number of **sensors**: web cameras are no longer optional components of laptops (as they once were), phones or tablets – they are almost always included and it takes extra effort to find models without them. GPS, temperature sensors, humidity sensors, luminosity sensors, accelerometers, microphones,… – users carry them along continuously, often without even realizing it. The telephones can sense location and proximity to various other digital devices or real-world objects (using Bluetooth, NFC, Wi-Fi, GSM base station / cell ID geolocation or other

[4] Also, the United Nations' Human Rights Council argued that disconnecting individuals from the Internet is a violation of human rights that is against the international law.

[5] Shared leased lines of 64 kbps (based on ISDN and other technologies) were not that uncommon. Today it is possible (in certain locations) for a private individual to buy a 100+ Mbps connection for a very reasonable price.

[6] If there is no DSL, no cable TV and no direct line-of-sight to a Wi-Fi ISP then mobile internet (with all its downsides) may the best option.

methods) and it is trivial to continually collect this environmental information and use it (not only) for search-related tasks[7]. We still haven't realized what we could do with mobile devices that have enough capacity to locally **store the recording of the entire human life**: i.e., all the sounds (or at least all the recognized speech), all the sights (or all the identified objects using machine vision), and all the information the user has ever seen. It is already possible to save several years of human sensory experiences on a mobile device and make it locally searchable, no network access needed[8]. As of year 2012, telephones and tablets with such large storage are becoming a reality: the whole "recording of a person's life" will not fit on a SD card yet, but available storage capacity grows fast enough – when the SD card is full, it is possible to simply replace it with a bigger one (say, twice as big in two years time) and keep recording.

1.2 Evolving Software

There are several areas of software development that seem to be especially important for the evolution of search. The first is related to **parallel processing**. With the proliferation of massive data centers (discussed below), the need to coordinate and exploit the connected resources has been growing (intra-rack, inter-datacenter,…). A similar thing has been happening on the desktop: increasing frequency may not be the best way to increase the overall performance for several reasons, adding more cores is preferred. As the processors contain more and more cores (Tilera Corp., for example, already manufactures 100-core processors), the need for parallel processing and **coordination of resources** in multiple execution threads is now dire. Software typically needs to be re-written to utilize multiple cores[9], but if it is optimized for parallelism[10], it can perform very complex tasks in a speedy way. Certain modern web browsers, for example, use multithreading and parallel execution across multiple processor cores for faster rendering (performance boost can be seen well when many tabs are open). Few people realize that when they perform a Google search, more than a thousand servers work simultaneously to zip through the index of the entire World Wide Web, rank the results, format the page, add the advertisements and return back the result; and this is all

[7] Lifetracking goes way beyond the basic phone sensors. The database at Quantifiedself.com lists over 500 resources for self-tracking and personal informatics.

[8] Of course, carrying such sensitive data around has interesting privacy implications, but that is beyond the scope of this text.

[9] Many consumers do not realize that having a 6-core CPU running old software is simply a waste (some cores may be unused or underused).

[10] When talking about this kind of processing, we could draw a parallel between individual workers and workers organized in factories or individual neurons vs. neural networks: the more sophisticated the network and coordination, the better the possible result.

orchestrated in less than 0.2 seconds[11]. This is another example of working in parallel.

Another important area of development that can positively influence search-related applications is **networking**. New communication protocols (SPDY – pronounced "speedy", WebSockets, etc.) are making the user experience faster and more secure and are now beginning to enter the mainstream (Twitter started using the experimental SPDY protocol in March 2012, even though it has not been standardized yet and has been implemented only in Google Chrome and Firefox). Another networking protocol, OpenFlow, focuses on the datacenters and helps them deal with network scaling and management[12]. New networking developments may help maximize the utility of high-bandwidth, low-latency networks and make the overall web experience faster, more secure and processing cheaper. This, together with other software and hardware developments, **helps create a "perfect storm"** for truly innovative client-server search solutions[13].

Much of the software mentioned previously has been developed in the open: most of the web is run on **open-source** operating systems, served by open-source servers, using open networking protocols and displayed to users on open-source web browsers. The availability of free software helps to cut costs dramatically and is enjoyed (in many ways) by IBM, Apple, Google and numerous startups alike. Moreover, forking and customizing open-source projects provides a quick way to bootstrap development for anyone.

1.3 The Rise of the Datacenter

One area of special importance with regards to the future of search is the **rise of the datacenter** during the last decade. With fast networks with low latency, powerful processing and cheap storage, it makes sense to **offload some work from the mobile clients** and move it to specialized computers located elsewhere. Of course, this is how web search has worked for a long time – but with today's networks and modern datacenters even more can be done: Apple uses "the cloud" for voice recognition in their Siri product (see below), Amazon's Silk browser in the Kindle Fire tablet uses Amazon Web Services to speed up the user's browsing experience (feature similar to Opera Turbo), etc..

[11] This information was published in year 2009. With Google adding complex new features to search, it would not be surprising if the number of servers used grew over time.

[12] OpenFlow does not help optimize for faster response times but it can help optimize the datacenter management, helping to speed up the back-end management processes.

[13] These networking developments may not (yet) have been implemented by IBM, Apple or Google in their discussed products, but they do seem to have the potential to help bring the search forward.

The end-user devices (smartphones, tablets) are doing their share of processing and are utilizing even more resources in the network when needed. When the clients are powerful enough, an interesting breed of search applications can come up (Google Goggles mentioned below is one of them).

Google, Facebook and Amazon have been leaders in bringing the datacenter evolution forward, each in a different way. 1) Google uses a number of proprietary technologies (both hardware and software, many designed in-house for its own needs) and is quite secretive about many of its moves. It also currently has more servers than any other company. By building a network of data centers around the world with physical proximity to the end users and good connectivity across network carriers it is able to offer extremely low latency and offer very good response time in their applications (search, Google Apps, etc.). Geographic concentration and presence near the edge of the network (close to the consumers) is important for snappy application user experience. 2) **Facebook** has decided to "open source" and share their data center designs as part of the OpenCompute project. Significant benefits to the wide community outside of Facebook operations remain to be seen but we think that the openness of the process is commendable. 3) **Amazon** is a large infrastructure provider used by a number of popular products (Netflix, Foursquare, Reddit, Quora, Dropbox) which allows startups to launch technologically complex services and scale elastically. When designing a product which needs a lot processing power and space to grow (which is hard to predict in advance) such as an innovative search engine, service providers like Amazon can make the development, the launch and scaling much cheaper than other potential options, in a number of scenarios.

It is important for any web service provider (PaaS, IaaS, SaaS, traditional web hosting companies, content delivery networks,…) to be reasonably close (in terms of latency) to the **edge of the network**: according to Nielsen (2009), "0.1 second is the response time limit if you want users to feel like their actions are directly causing something to happen on the screen. (…) If it takes longer than 0.1 seconds for the revised state to appear, then the response doesn't feel 'instantaneous' — instead, it feels as if the *computer* is doing something to make the menu open." Google's Senior Vice President Urs Hölzle has made it clear that his company will focus hard on the aspect of speed: "We want you to be able to flick from one page to another as quickly as you can flick a page on a book. So we're really aiming very, very high here… at something like 100 milliseconds."

2 IBM Watson

IBM supercomputer called "Watson" convincingly showed in the game show Jeopardy! that it could **successfully challenge the very best human players** in a quiz show and work with natural language in a fast and sophisticated way. It was not the first time when computers challenged talented humans in a game: IBM's Deep Blue computer beat the then-world-champion Garry Kasparov in 1997. In 1996, Chinook software from the University of Alberta in Canada was the program which won a "human" world championship (in checkers). And there were others. Watson, however, sounded very impressive because **it competed with humans using "their" natural language** – this was not just a simple game which could be reduced to algorithms. As an article printed in the Economist (The Difference Engine) put it, "defeating a grandmaster at chess was child's play compared with challenging a quiz show famous for offering clues laden with ambiguity, irony, wit and double meaning as well as riddles and puns—things that humans find tricky enough to fathom, let alone answer."

The details of the game and technology used were discussed in Gondol (2011). Here we'll mention several ideas which we believe are important to the evolution of search applications.

First of all, Watson was a system built for **question answering** (DeepQA technology) in a very specific class of questions, not intended for general search. Yes, it outsmarted very capable humans, but it was programmed for one specific domain – to win in Jeopardy!, with specific kinds of questions and specific answers. We need to point out, however, that in the future, it may be optimized for other uses: healthcare (helping doctors with diagnoses), law (processing vast collections of legal text), etc. IBM was developing this technology not just as a PR stunt but as a marketable technology.

One of the reasons why Watson stands out is its pragmatic design: it does not fully comprehend the question. It **does not require perfect understanding of language**, unambiguous semantic analysis and perfect proof for its answers. It instead focuses on finding the most likely answer candidates. After generating a large number of potential answers, Watson tries to find proof for each generated hypothesis, using both structured and unstructured data. Crucially, it is able to calculate confidence for responses, so it knows when the answer is very likely to be true and when it is more of a guess.

Question-answering systems can become very capable and reliable and could be **coupled with more traditional search applications** (e.g. – Is the incoming query formulated as a question? If so, use a Watson-like system for parsing the question and generating potential answers and see if they have high enough confidence. If the input is not a question or computed confidence is low, try something else, e.g. run a traditional full-text search.)

Watson uses a lot of open-source technology but as a whole, it is a closed-source and expensive commercial system for now. In a few years, however, it is quite reasonable to expect that a fully **open-source lightweight alternative** (albeit less capable in the beginning) will be available. Also, powerful hardware is likely to keep becoming cheaper, so an overall "Watson clone" may be soon within reach of many organizations, even individuals. As we said, such question answering technology may be an enhancement for search applications.

3 Apple Siri

Sci-fi literature and movies reflect people's long-time fascination with talking computers. Siri, software agent (personal assistant) using voice recognition for input running on Apple iOS, makes such voice-based interface available to the masses[14]. It was introduced in October 2011 and is currently (as of early April 2012) only officially available for iPhone 4S (unofficial ports not supported by Apple can also be found). It uses advanced technology for language processing, and, according to Apple's official web site, it "understands what you say, knows what you mean, and even talks back". The communication is in **natural language and there is no need to remember specific commands** and keywords (which typically used to be required in competing products). If additional information is needed, Siri asks follow-up questions. It can be used to perform a number of tasks, e.g. (examples in this section are from Apple.com):

- **Set reminders**, alarms, timers. One can say when and where the reminders should appear, e.g. "Remind me to make a dentist appointment when I get to work".
- **Send text messages** (sms) and e-mails. ("Tell my wife I'm running late" / "Text Ryan I'm on my way".)
- **Find information** ("Any good burger joints around here?"). Find contacts. Find directions ("Where's Apple?" / "How do I get home?"). Search the web. If applicable, sites like Yelp and Wolfram Alpha will be used for finding relevant information.
- **Schedule meetings** (input of information in the calendar: what, who, where). It is a convenient interface for entering calendar events.
- **Place phone calls** ("Call a taxi"). If needed, Location Services are used to determine the current location.
- **Play music** (e.g. by a specific artist).

[14] Of course, Siri is not the first such application, but it is important because it is quite capable and it is being mass-marketed.

- Voice recognition also works for **dictation**. Text input in third-party applications is supported.
- To find **other options**, one can ask Siri about Siri. ("What can you do?")

As we could see from the example queries, it is possible to talk to Siri as if it were a human being. It will **use the context** (location, time, etc.) and the information learned about the user ("home" address, "work" address, "mom's" telephone number) to carry out the tasks. If more information is needed (e.g., when asked to call "mom" for the first time), Siri **asks for clarification**. Once it learns who "mom" is (which contact is assigned to her) or where "home" is, it will be able to automatically re-use this information later.

There are a number of advanced technologies running in the background. The focus here, however, is on **carrying out tasks**, not question answering (like in IBM Watson) or simple web search, all the while using many of the iPhone built-in applications (and possibly 3rd-party services in the future, if an API is introduced).

Some journalists commended Apple for creating a **"personality" for Siri**[15]. There have also been, on the other hand, criticisms and even lawsuits for "over-promising" (and under-delivering) functionality. Moreover, certain English speakers with distinctive accents (e.g. the Scottish) criticized Apple for poor voice recognition capability and various users were unhappy that certain features (navigation, traffic information, etc.) were available only inside of USA. While Siri may be imperfect in some aspects, we should realize that this technology is still in development and is likely to **get better over time**. With Apple backing the project, millions have now been exposed to the technology and there should be plenty of training data to feed into the machine learning algorithms for further improvement.

Google currently does not have software that would completely replace Siri on Android handsets. **Google Voice Actions has similar functionality**[16] but specific pre-defined commands need to be used ("send text to…" / "navigate to…" / "go to Wikipedia" / "call … home"), the communication is currently not "free-form" or conversational. The expected **upgrade**

[15] Siri will sometimes respond in a humorous way, as noted even by mainstream media (examples from a CNN article: Q: Am I fat? A: I prefer not to say. – Q: What are you wearing? A: You have the wrong personal assistant, Clint. – Q: Siri, are you affiliated with Skynet? A: I can't answer that. – Q: How much wood would a woodchuck chuck if a woodchuck could chuck wood? A: Don't you have anything better to do?). TalkToThePhone.com is a web site that collects funny or quirky responses of Siri.

[16] It can be used to send text messages and e-mails, get driving directions, call contacts and businesses, navigate to web sites, etc..

(codenamed Majel) should make Voice Actions more Siri-like and work with casual language. Google's acquisitions put it in a position to create an application that leverages the user's interest graph and is a capable digital assistant.

We think that conversational interface certainly has its place, such as **in the car** (e.g. for completing simple tasks while waiting in the traffic jam, as it can be dangerous to be distracted while driving – unless one uses a self-driving car, of course). In a number of other situations, conversing with the phone can be (at the present time) **socially awkward**. This awkwardness may go away when such activity becomes commonplace, just like it did with hands-free headsets. Using dictation of text messages in public may be undesirable because it is not discrete. Asking for "good restaurants near work" is a query that one will probably not repeat very often – it is more useful when traveling – but it is questionable how many people will want to publicly broadcast verbal queries hinting at the fact that they are strangers. Playing music is convenient enough using current interfaces, no Siri needed.

We believe, however, that voice-based interface does have its place in smartphones and other devices – possibly robots of the future; asking is sometimes more convenient than typing. Therefore it is not a surprise that Siri is going to be integrated with systems in certain Mercedes Benz vehicles, possibly with upcoming iPhone cameras and maybe with some Bluetooth Low Energy devices ("Siri, lock the front door."). A New York Times blog said that the future AppleTV may feature Siri (the user can ask for a specific TV show). It would not be surprising because some Samsung TVs already do voice recognition (in addition to face recognition, so the TV is actually "watching the watcher" through the built-in HDTV camera, in an interesting twist of irony) and Nuance, the provider of voice recognition to Apple, advertises its own voice-controlled television: the Dragon TV.

4 Google Goggles

Google has been experimenting with **untraditional ways to find information** – without having to use keyword-based search: "search by voice", "search by location" (helps to learn about traffic conditions, get navigation information, find points of interests, such as restaurants, hotels, shops, etc. – e.g. "what's nearby" feature in Google Maps) and "search by sight". When searching visually, Google recognizes the object(s) seen by the user's camera and returns information (e.g. web pages, translation) related to them.

Google Goggles is a downloadable **image recognition[17] application** for mobile operating systems. Just like SoundHound or Shazam can identify music playing on the radio (using each song's unique "signature" or a "digital fingerprint"), Google Goggles can identify objects in the smartphone camera's field of view. All that is necessary is to launch the application and take a photograph[18]. If Goggles recognizes a landmark, for example, it will display related information, such as the corresponding Wikipedia page.

According to the official web page[19], it currently (as of April 2012) supports the recognition of:

- **Text** (turns a photographed image into text). This text can be translated into a foreign language using Google Translate technology. Because the recognition is server-based, a data plan (or a Wi-Fi connection) is required.
- **Landmarks**. The application performs correctly for well-known landmarks. It will take some time, however, for it to learn the visual appearance of lesser known (and rarely photographed) objects all around the world.
- **Books**. When a book cover is scanned, Goggles will attempt to identify the book. In our test, this performed correctly even for many non-English book covers (e.g. children's books in Slovak language) but failed to identify others. Again, with more training data available, the situation is likely to improve.
- **Contact information**. Goggles can now scan a business card and save the information to the cell phone's contacts (quickly, without typing)[20]. We consider this to be a very practical feature. The smartphone replaces yet another device – a portable business card scanner (just like it replaced point-and-shoot cameras, MP3 players, pocket GPS navigation devices and others).
- **Artwork**. Again, this feature performed quite satisfactorily in our test. When a painting is scanned, Google attempts to identify it and provide more information. Watching art exhibitions is therefore no longer dull even for non-enthusiasts, paintings in a way "come alive".

---

[17] Visual search / image-based search / reverse image search (highly advanced mobile version of services like tineye.com).

[18] Pressing the camera button is no longer necessary in the latest version, continuous scanning is used.

[19] The comments are ours.

[20] We also recommend adding the person on social networks, especially LinkedIn. If this person leaves to work at another company, LinkedIn will help to stay in touch with them, even though the contacts on the paper business card may no longer work.

- **Wine**. Scanning of wine labels is supported.
- **Logos**. It is now easy to open the web site of the company to which the logo belongs.

The application supports even more objects, not explicitly mentioned by the home page:

- **Covers of DVDs**, covers of computer games,…
- **Bar codes, QR codes**[21]. There are already applications that perform this kind of recognition[22]. When a bar code is scanned, it is possible to find prices for the item in other stores, both brick-and-mortar (provided they publish the price information online) and in online stores. Such functionality, used en masse, has a potential to disrupt retail, further hurting local store owners (e.g. booksellers) and putting more pressure on the margins which are already thin in some retail segments. Moreover, the user can look for similar products if he/she is not satisfied with the current one.
- **Sudoku**. When a Sudoku is scanned, Goggles proposes a solution. While this may seem amusing, other functionality is probably more useful.

There is currently only limited support for recognition of common objects like plants or animals. In the future, however, it should be easy to recognize plant leaves thanks to their often distinctive geometric shape.

Google Goggles was launched in Google Labs in December 2009. In February 2010 (at the Mobile World Congress in Barcelona), Google's then-CEO Eric Schmidt demonstrated software prototype version with **text translation** feature. A photograph would be taken, image recognized as text, OCR'd (recognized) and translated to a foreign language. It is easy to imagine the usefulness of this: translation of street signs in foreign countries, **menus in restaurants**, etc. (provided that Internet connectivity is available, which is not always the case). Goggles 1.1 launched three months later (in May 2010) with the previously demoed integrated translation and other rather incremental improvements (just like Goggles 1.2). In October 2010, Google Goggles

[21] In a way, Goggles-style software tools can in fact replace some QR codes because they can identify objects directly and redirect the user straight to a URL associated with the recognized object.

[22] SnapTell (for Android or iPhone) displays links to Amazon, YouTube, Wikipedia, IMDB and more, when presented with a barcode or a photograph of a DVD cover. ShopSavvy (for Android or iPhone) is a barcode reader with comparison shopping features.

technology became available outside of Android, in **iPhone's "Google Mobile"** application (which also includes voice queries etc.). Version 1.3 for Android (January 2011) added an almost-instant scanning of barcodes, recognition of print advertisements in USA-based magazines and newspapers (helping to find web search results about the brand or product) and support for solving Sudoku puzzles. Goggles 1.4 (May 2011) introduced the ability to suggest better results to Google (effectively partially **crowdsourcing search results curation** to users), improved **business card recognition** (the card is now recognized as a contact, not just general text – so the information is easier to add to the address book in the telephone) and improved search history[23] (support for personal notes, sharing with friends). Version 1.5 (June 2011) added support for recognition of Cyrillic characters (in addition to Latin character sets) and copying of the recognized text into clipboard for later processing, which effectively turned the user's mobile phone into a very **simple OCR-capable scanner**. Version 1.6 (September 2011) brought the opt-in "search from camera" feature which must be manually enabled (all snapped photographs are then analyzed in the background and if they contain recognized objects, the user is notified).

The now-current Android version 1.7 (released in December 2011) added an innovation which could hint at the things to come: **continuous mode**. There is now no need to photograph the object first and recognize it later. The analysis has become real-time, no shutter-press required (fast internet connection and good light are needed, however). This version does not yet support text recognition for continuous mode (unlike, e.g., Word Lens application, which features live real-time translation of text signs within the field of view) but it does work **for books, products, artwork, and landmarks**. Another important update is the ability to photograph a document (or a text snippet) and search for its text on the web. For example, one could take a snapshot of a news article and search the web for the full text of the article's online version.

In February 2012, New York Times reported that Google was on the path to this year's release of **wearable LCD projection glasses** with built-in camera for monitoring the surrounding and real-time display of relevant information within the user's field of view, working similarly to fighter pilots' HUDs (head-up displays) or the vision of Terminator from the Hollywood movie. The "real physical Goggles" were said to be developed by Google X, a secretive lab working on things like self-driving cars, "space elevators and dozens of other futuristic projects." Another story appeared on a New York Times "bits" blog on April 4, 2012, when the project was officially confirmed and given a name

[23] There is currently no desktop version of Google Goggles, but it is possible to view previous visual queries at https://www.google.com/goggles/history.

("Project Glass"). Some further details were also announced. Google co-founder Sergey Brin was seen wearing a prototype of these glasses on April 5 and the technology blogger Robert Scoble who reported this said that they were **"many months, if not years" away** from being productized[24].

Somewhat similar **"smart glasses"** projecting context-sensitive information to the user are no longer a sci-fi fantasy, and they are already available for sale. Recon Instruments, for example, sells MOD Live: a device built into **skiing goggles**, displaying speed, jump analytics (useful to snowboarders), altitude, GPS location, temperature and other data. When connected to an Android phone, these glasses can display caller ID for incoming calls, show incoming text messages (which we think is a distraction on the slope), etc. There is no built-in camera which could be directly used by Google Goggles or a similar application. MOD / MOD Live devices can be currently fit in select Zeal Optics[25], Uvex, Alpina and Briko goggles.

There are also entertainment and other uses for "virtual reality glasses". In March 2012, Epson released an Android-based **see-through wearable display** Moverio BT-100 which is optimized for entertainment experience[26] and could be probably adapted for use with augmented reality (AR)[27] applications. Other device manufacturers include Rochester, NY-based Vuzix (with products ranging from cheap video eyewear to sophisticated AR glasses selling for $5000) and Lumus Optical from Israel, which according to NPR, already have working prototypes similar to Google's. There is a wide range of products among these manufacturers, using both binocular and monocular projection and having other technical differences because the use cases are also different. When the glasses are coupled with a camera, gyroscopes and other sensors to detect the position of the head and fine movement, they can be used for sophisticated applications. In the future, projection of graphics may be done through bionic contact lenses and there are interesting research projects going on that show promise.

Software solutions for projecting overlay images on top of the video recorded by the telephone's camera (similar to the possible future version or a fork of Google Goggles) already exist. Such applications are called "augmented-reality browsers" and their examples include Layar, Wikitude and

[24] See Robert Scoble's tweet at https://twitter.com/scobleizer/status/188176052776992768.

[25] Also available as an integrated package (goggles + MOD / MOD Live device).

[26] Examples: watching videos and playing music. The device reportedly supports Dolby Mobile surround, Adobe Flash, etc..

[27] Reality can be augmented in various degrees, marrying the "offline world" with the "online world": from simple display of relevant information based on location or other context to complex mixture of physical world view with computer-generated data. This continuum of virtuality is nicely illustrated in Nokia's "Mobile Mixed Reality: The Vision" paper.

Junaio. If Google Googles is extended or forked to become a fully capable AR browser, it will not be the first. But, in case it is open to third-party developers, it may become a very interesting AR platform.

Google's venture into wearable computing can have major impact: first, it is easy to see usefulness of a **well-integrated product** which includes both hardware and software (or firmware) that cooperate well together. Second, Google's sheer size can have a big impact on the rapidity of proliferation of this technology.

Augmented reality has a **number of potential uses** for serendipitous discovery: walking around and seeing which houses are for sale[28], seeing reviews of restaurants in a new city, noticing that theater tickets are available for this week's new play; all of these can be quite helpful. If the system is designed well, only showing personally relevant information instead of distracting (and worsening the already bad information overload), then it can be genuinely useful and (when coupled with usable communication features) even completely **replace a smartphone**.

There are, however, certain worries. Google has already been using **face recognition** technology (e.g. for automatic tagging of user photos on the Google+ network). Combined with Project Glass, this feature could help recognize people walking down the street and display information about them. While some users claim this would be exciting (e.g. seeing the person's name, common friends on social networks, latest status update = possible conversation starter), there is also a significant **potential for abuse**. Having location and various personal data from social networks exposed to strangers, moreover in the physical environment, may not be desirable. The potential of being instantly profiled and tracked by third parties may not sound very appealing to many. **Relationship status, health status, financial status, past or present location or other data could be leaked** to the third parties en masse, often because a person is simply unaware or unable to tighten their security settings online or because there may be a security problem with the software or the provider. According to the New York Times (issue from February 23, 2012), "This month, the Electronic Privacy Information Center, a research and advocacy group for Internet privacy, asked the Federal Trade Commission to suspend the use of facial recognition software until the government could come up with adequate safeguards and privacy standards to protect citizens." We think, however, that it is questionable what the FTC can do. Even if the officially sold devices would have to adhere to certain standards, it may be possible to re-flash the firmware and use various **workarounds to avoid the tightened privacy settings**, especially on open systems like Android. Also, if there is an opt-out

[28] In Slovakia, an application using the Layar platform can be found at vidimbyty.sk.

procedure, it can be bypassed (or if there's opt-in, it can be ignored by the device, which can keep trying to profile people in the streets regardless of the settings). If the data exists and is stored and shared, the potential for abuse is there.

Google Goggles is a useful application and is getting more useful over time. It is possible future integration with the glasses (from Google or from other manufacturers), **blending the physical world and the "mirror worlds"** can bring interaction with information to a completely new level for many people. Even if the basic software application is free of charge, it is not completely free by any means. The payment is in form of giving up privacy (of the user and the people around the user) and data that is sent to Google.

Smart phones are becoming "smarter", **machine vision is becoming mainstream**. There are likely to be many surprises and disruptions ahead and they are hard to predict. It is clear, however, that the ways we interact with information and how we search are changing dramatically. We think it would be naïve to think that the bulk of innovation is already behind us. In our opinion there are many opportunities and significant changes coming up.

5 The Past and The Future

There has been a major **rise of the mobile technology**, enabled by the availability of new devices (smartphones, tablets) and new software and services. Smart mobile devices have become significantly cheaper and are now more commonplace than ever. For some users, they have at least partially replaced the desktop (for communication / e-mail, staying up-to-date, entertainment, reading,…). The devices have powerful capabilities (fast multi-core CPUs, GPUs, networking capability, gigabytes of storage) and a number of sensors (incl. GPS, noise-cancelling microphones, high-quality front and rear cameras and others) which almost "beg" software developers to find their innovative uses. Part of this use can be in search-related and task-oriented applications, as we have seen in the above examples.

Evolving technologies have been connected with **social and lifestyle changes**. A number of people have started living an "always on" life, started using a growing number of computing devices (smartphone, tablet, desktop / laptop, etc.), are used to receiving information in near-real-time, and are expecting the devices to respond in near-real-time as well[29]. As the availability

[29] Notice the push for "instant on" / boot speedup, browser acceleration etc..

of information increases, mental capacity to process it does not[30]. The ways users interact with information changes and so do the tools which are being used every day.

Search has **come a long way** from simple "string matching" and typing keywords toward gradually more understanding of language and intent behind the queries. Of course, there are many languages in the world and the complexity of the task needs to be appreciated. There is still a long way to go toward realizing the semantic web and other grand visions – but in spite of this, there are technologies that are pushing search and information discovery forward. As we have seen, IBM Watson understands "just enough" language to answer the questions. Siri can interact with "just enough" smartphone applications to be useful.

6 Conclusion

We have pointed to the technology trends and several tools to show how technologies powering search have been evolving recently and what infrastructure exists for their development in the future. As unfancy as it may seem, we believe that applications such as these are **likely not going to fully replace traditional search** as we know it (text-based, with keyboard input) anytime soon – a simple search (the kind we use at google.com) is often faster than having to formulate a question and saying it out loud. Also, it is often more convenient (it may be rather unpleasant to speak aloud in front of others), and it may be easier when exploring a new topic: often the searchers are not even sure what they are looking for and it may be easier to type a few keywords, scan through the many results on the screen and further refine the query. In the mobile environment, however, location-based technologies (using current context) and task-oriented technologies (ability to launch applications instead of just a simple search) can be helpful. Being able to search by voice and search by sight complements the ways to find out about the world around and enables new ways of interaction with digital information.

Many technologies we mentioned here are client-server, which leads to a large amount of **personal and behavioral details being transferred** over the network and stored with the search provider. This has serious implications – both negative (e.g. privacy issues) and positive (system that knows the user intimately can be an extremely helpful personal assistant). It will be interesting

[30] This "failure of filters" (to use Clay Shirky's term) often leads to feeling of being overloaded. The researchers around prof. Jela Steinerová at the Comenius University in Bratislava recently studied these phenomena as part of their grant focused on information ecology.

to observe which of the trends will have an impact on the search market and what the future disruptions will be.

Bibliography

[1] Ahmed,Saeed. 2009. *Fast Internet access becomes a legal right in Finland*. URL: http://edition.cnn.com/2009/TECH/10/15/finland.internet.rights/ (2012-03-27)
[2] *Ask Siri to help you get things done*. 2012. URL: http://www.apple.com/iphone/features/siri.html (2012-03-27)
[3] *Augmented Planet Books*. 2012. URL: http://www.augmentedplanet.com/resources/books/ (2012-03-26)
[4] *Augmented Reality Browser: Layar*. 2012. URL: http://layar.com/ (2012-03-26)
[5] *Augmented Reality Standards*. 2012. URL: http://www.perey.com/ARStandards/ (2012-03-27)
[6] Belimpasakis, Petros et al. 2010. *Mixed Reality Web Service Platform (MRS-WS)*. URL: http://research.nokia.com/page/9351 (2012-03-26)
[7] Bilton, Nick. 2011. *What's Really Next for Apple in Television*. URL: http://bits.blogs.nytimes.com/2011/10/27/whats-really-next-for-apple-in-television/ (2012-03-27)
[8] Bilton, Nick. 2012a. *Google to Sell Heads-Up Display Glasses by Year's End*. URL: http://bits.blogs.nytimes.com/2012/02/21/google-to-sell-terminator-style-glasses-by-years-end/ (2012-03-26)
[9] Bilton, Nick. 2012b. *Behind the Google Goggles, Virtual Reality*. URL: http://www.nytimes.com/2012/02/23/technology/google-glasses-will-be-powered-by-android.html (2012-03-27)
[10] Bilton, Nick. 2012c. *Google Begins Testing Its Augmented-Reality Glasses*. URL: http://bits.blogs.nytimes.com/2012/04/04/google-begins-testing-its-augmented-reality-glasses/ (2012-04-06)
[11] Bissacco, Alessandro et al. 2010. *Translate the real world with Google Goggles*. URL: http://googlemobile.blogspot.com/2010/05/translate-real-world-with-google.html (2012-03-26)
[12] Bixby, Joshua. 2010. *Are your website's performance goals audacious enough?* URL: http://www.webperformancetoday.com/2010/09/23/are-your-performance-goals-audacious-enough/ (2012-03-28)
[13] Broum, Milan. 2010. *Open your eyes: Google Goggles now available on iPhone in Google Mobile App*. URL: http://googlemobile.blogspot.com/2010/10/open-your-eyes-google-goggles-now.html (2012-03-27)
[14] Chan, Alice. 2012. *Mercedes-Benz Brings Siri to Their Cars*. URL:

http://www.psfk.com/2012/02/mercedes-benz-siri.html (2012-03-27)
[15] Chitu, Alex. 2009. *1000 Machines Find the Results for a Google Query*. URL: http://googlesystem.blogspot.com/2009/02/machines-search-results-google-query.html (2012-04-02)
[16] Chitu, Alex. 2011. *Continuous Mode in Google Goggles*. URL: http://googlesystem.blogspot.com/2011/12/continuous-mode-in-google-goggles.html (2012-03-26)
[17] Chitu, Alex. 2012. *Google's Project Glass*. URL: http://googlesystem.blogspot.com/2012/04/googles-project-glass.html (2012-04-06)
[18] Christian, Brian. 2011. *Mind vs. Machine*. URL: http://www.theatlantic.com/magazine/print/2011/03/mind-vs-machine/8386/ (2011-02-22)
[19] *Dragon TV*. 2012. URL: http://www.nuancemobilelife.com/dragontv/ (2012-04-06)
[20] Gondoľ, Ján. 2011. Watson: význam počítača, ktorý porazil šampiónov v hre Jeopardy! In: *Kognice a umělý život XI*. Opava: Slezská Univerzita v Opavě, 2011, s. 73-76. ISBN 9788072486441.
[21] *Google Goggles Home Page*. 2012. URL: http://www.google.com/mobile/goggles/ (2012-04-06)
[22] *Google Goggles Release Notes*. 2012. URL: http://support.google.com/websearch/bin/answer.py?hl=en&answer=181358 (2012-03-26)
[23] *Google Goggles Search History*. 2012. URL: https://www.google.com/goggles/history (2012-03-27)
[24] Gross, Doug. 2011. *Snide, sassy Siri has plenty to say*. URL: http://edition.cnn.com/2011/10/18/tech/mobile/siri-answers-iphone-4s/ (2012-03-27)
[25] *Guide to Self-tracking tools*. 2012. URL: http://quantifiedself.com/guide/ (2012-03-28)
[26] Gundotra, Vic. 2009. *Mobile Search for a New Era: Voice, Location and Sight*. URL: http://googlemobile.blogspot.com/2009/12/mobile-search-for-new-era-voice.html (2012-03-27)
[27] Henn, Steve. 2012. *Google's Goggles: Is The Future Right Before Our Eyes?* URL: http://www.npr.org/blogs/alltechconsidered/2012/02/24/147364732/googles-goggles-is-the-future-right-before-our-eyes (2012-03-27)
[28] *Introducing Amazon Silk*. 2011. URL: http://amazonsilk.wordpress.com/2011/09/28/introducing-amazon-silk/ (2012-03-26)
[29] Jackson, Nicholas. 2011. *United Nations Declares Internet Access a Basic*

Human Right. http://www.theatlantic.com/technology/archive/2011/06/united-nations-declares-internet-access-a-basic-human-right/239911/
[30] *Junaio Augmented Reality*. 2012. URL: https://play.google.com/store/apps/details?id=com.metaio.junaio (2012-03-27)
[31] *Layar*. 2012. URL: https://play.google.com/store/apps/details?id=com.layar (2012-03-28)
[32] *Lumus Home*. 2012. URL: http://www.lumus-optical.com/ (2012-03-26)
[33] *Make the Web Faster*. 2012. URL: https://developers.google.com/speed/ (2012-04-06)
[34] Lynn, Samara. 2011. *Dissecting IBM Watson's Jeopardy! Game*. URL: http://www.pcmag.com/article2/0,2817,2380351,00.asp (2011-02-20)
[35] McGlaun, Shane. 2012. *Epson ships Moverio BT-100 Android see-through glasses*. URL: http://www.slashgear.com/epson-ships-moverio-bt-100-android-see-through-glasses-28220338/ (2012-04-06)
[36] *Mobile Mixed Reality: The Vision*. 2009. URL: http://research.nokia.com/files/insight/NTI_MARA_-_June_2009.pdf (2012-03-28)
[37] *MOD Live GPS Heads up Display*. 2012. URL: http://www.reconinstruments.com/products/mod (2012-03-27)
[38] Nachman, George. 2011. *Google Goggles learns Russian and gets a new view*. URL: http://googlemobile.blogspot.com/2011/06/google-goggles-learns-russian-and-gets.html (2012-03-27)
[39] Neven, Hartmut. 2010. *Integrating translation into Google Goggles*. URL: http://googletranslate.blogspot.com/2010/02/integrating-translation-into-google.html (2012-03-27)
[40] Nielsen, Jakob. 2009. *Powers of 10: Time Scales in User Experience*. URL: http://www.useit.com/alertbox/timeframes.html (2012-03-24)
[41] Nielsen, Jakob. 2010. *Website Response Times*. URL: http://www.useit.com/alertbox/response-times.html (2012-03-24)
[42] Nikolopoulos, Spiros. 2011. *Study on Mobile Image Search*. URL: http://mklab.iti.gr/files/Nikolopoulos_MobileImageSearchStudy_2011.pdf (2012-03-26)
[43] *Nokia Mixed Reality*. 2009. URL: http://www.youtube.com/watch?v=CGwvZWyLiBU (2012-03-27)
[44] *Official Google Mobile Blog: Google Goggles*. 2012. URL: http://googlemobile.blogspot.com/search/label/google%20goggles (2012-03-28)
[45] *Open Compute Project: Hacking Conventional Computing Infrastructure*. 2012. URL: http://opencompute.org/ (2012-03-26)

[46] *Openflow: enabling innovation in your network*. 2012. URL: http://www.openflow.org/ (2012-03-27)
[47] Palm, Leon et al. 2011. *Google Goggles gets faster, smarter and solves Sudoku*. URL: http://googlemobile.blogspot.com/2011/01/google-goggles-gets-faster-smarter-and.html (2012-03-28)
[48] Petrou, David. 2011. *Continuous improvements with Google Goggles 1.7*. URL: http://googlemobile.blogspot.com/2011/12/continuous-improvements-with-google.html (2012-03-27)
[49] Pogue, David. 2011. *Siri is One Funny Lady*. URL: http://pogue.blogs.nytimes.com/2011/10/14/siri-is-one-funny-lady/ (2012-03-28)
[50] *Project Glass*. 2012. URL: http://g.co/projectglass + redirects to: https://plus.google.com/111626127367496192147/posts (2012-04-06)
[51] *Project Glass: One day...* 2012. URL: http://www.youtube.com/watch?v=9c6W4CCU9M4 (2012-04-06)
[52] *Recon Instruments*. 2012. URL: http://www.youtube.com/user/ReconInstruments (2012-03-27)
[53] Roemmele, Brian. 2012. *Not Your Dad's Voice Recognition System*. URL: http://www.quora.com/Apple-Products-and-Services/Why-is-Siri-important/answer/Brian-Roemmele (2012-04-02)
[54] Schroeder, Stan. 2011. *Google+ Gets Face Recognition, Deeper Gmail Integration*. URL: http://mashable.com/2011/12/09/google-plus-face-recognition/ (2012-03-26)
[55] Schwartz, Elliot. 2009. *Find what's nearby and try Labs features with Google Maps for Android*. URL: http://google-latlong.blogspot.com/2009/12/find-whats-nearby-and-try-labs-features.html (2012-03-27)
[56] Scoble, Robert. 2012. URL: https://twitter.com/scobleizer/status/188176052776992768 (2012-04-06)
[57] *ShopSavvy*. 2012. (2012-03-27) URL: https://play.google.com/store/apps/details?id=com.biggu.shopsavvy
[58] *Siri - Frequently Asked Questions*. 2012. URL: http://www.apple.com/iphone/features/siri-faq.html (2012-03-26)
[59] *Siri, the Virtual Personal Assistant for the Apple iPhone 4S: Breakthrough technology originally developed by SRI*. 2012. URL: http://sri.com/about/siri.html (2012-03-27)
[60] Nielsen Research. 2012. *Smartphones Account for Half of all Mobile Phones, Dominate New Phone Purchases in the US*. URL: http://blog.nielsen.com/nielsenwire/online_mobile/smartphones-account-for-half-of-all-mobile-phones-dominate-new-phone-purchases-in-the-us (2012-04-06)

[61] *Smart TV Motion Control, Voice Control and Face Recognition - Samsung CES 2012 Press Conference*. 2012. URL: http://www.youtube.com/watch?v=5C1nADiC6OE (2012-04-06)
[62] Smullyan, Jacob. 2011. *Share and personalize your Google Goggles experience with Goggles 1.4*. URL: http://googlemobile.blogspot.com/2011/05/share-and-personalize-your-google.html (2012-03-27)
[63] *SnapTell*. 2012. URL: https://play.google.com/store/apps/details?id=com.snaptell.mobile.client.android (2012-03-26)
[64] *SPDY - The Chromium Project*. 2012. URL: http://www.chromium.org/spdy (2012-03-28)
[65] *Talk To The Phone: What Does Siri Say To You?* 2012. URL: http://www.talktothephone.com/ (2012-03-27)
[66] *The Difference Engine: The answering machine*. 2011. URL: http://www.economist.com/blogs/babbage/2011/02/artificial_intelligence (2012-03-29)
[67] *The IBM Jeopardy! Challenge FAQs*. 2011. URL: http://www.research.ibm.com/deepqa/faq.shtml (2011-02-19)
[68] *TILE-Gx Processor Family*. 2012. URL: http://www.tilera.com/products/processors/TILE-Gx_Family (2012-03-28)
[69] Tsotsis, Alexia. 2011. *Google's Plan To Compete With Apple's Multi-Platform Siri? Google "Assistant"*. URL: http://techcrunch.com/2012/03/02/2011-was-the-year-of-social-for-google-2012-is-the-year-of-assistant/ (2012-03-28)
[70] *Velocity 2010: Urs Holzle*. 2010. URL: http://www.youtube.com/watch?v=MStKwEff_kY (2012-03-27)
[71] *Vidím byty: Nový pohľad na nehnuteľnosti*. 2012. URL: http://www.vidimbyty.sk/ (2012-03-26)
[72] Vodenski, Pavel. 2011. *Your smartphone camera is now smarter with Goggles 1.6*. URL: http://googlemobile.blogspot.com/2011/09/your-smartphone-camera-is-now-smarter.html (2012-03-27)
[73] *Voice Actions for Android*. 2012. URL: http://www.google.com/mobile/voice-actions/ (2012-03-27)
[74] *Vuzix Consumer Product Browser*. 2012. URL: http://www.vuzix.com/consumer/products_browse.html (2012-03-28)
[75] Weintraub, Seth. 2012. *HUD Google Glasses are real and they are coming soon*. URL: http://9to5google.com/2012/02/06/hud-google-glasses-are-real-and-they-are-coming-soon/ (2012-03-27)
[76] *Wikitude World Browser*. 2012. URL: https://play.google.com/store/apps/details?id=com.wikitude (2012-03-28)

Modeling of the Collective Behavior of Chemical Swarm Robots

Peter GRANČIČ and František ŠTĚPÁNEK[1]

Abstract. The present work describes the mathematical concepts for modeling the collective behavior of chemical swarm robots. Chemical swarm robots are autonomous Brownian particles designed to encapsulate, deliver and release specific chemical cargo. In the desired applications such as targeted drug delivery and distributed chemical processing, the ability of the robots to localize a given target is of crucial importance. In porous environments, where passive or random Brownian motion becomes inefficient, the motion of robots must be enhanced by employing a self-propulsion mechanism, similar to bacterial chemotaxis. Within such target localization mission, the robots coordinate their motion according to the concentration gradients of the chemical signals released in response to external stimuli. The example results show that relatively simple signaling strategy becomes a very efficient tool in guiding a swarm of chemical robots towards a given target.

1 Introduction

The reader may be surprised by the rather unusual connection of chemistry and robotics in the title of the present work, but according to the original definition of a "robot", introduced by Karel Čapek in his drama R.U.R. (Rossum's Universal Robots), the first robots were not based on electro-mechanical devices but their constituent material was "… some kind of colloidal jelly that not even a dog would eat" (R.U.R., prologue). Using the present-day terminology, Čapek's robots were based on soft matter.

In order to give a more precise definition, chemical swarm robots [1, 2] are autonomous programmable environment-responsive colloidal particles designed to execute chemical tasks within their environment. Further, these

[1] Chemical Robotics Laboratory, Institute of Chemical Technology, Technická 5, 16 228 Prague, Czech Republic, E-mail: frantisek.stepanek@vscht.cz.

artificially synthesized entities are expected to behave in a similar way as natural colonies of micro-organisms. To coordinate their mutual actions, the robots communicate using chemical signals that are released from their bodies in response to local stimuli.

Conceptually, chemical swarm robots are inspired by numerous examples of natural swarms, such as the slime mould *Dictyostelium discoideum* [3, 4]. A common feature of these natural swarm systems is their capability to accomplish tasks that are far beyond the capability of a single individual. The bodies of living organisms are based on soft matter with cells as the fundamental building blocks. Using cell-like entities rather than bulk materials as the basic building blocks of functional devices offers several advantages such as the potential for adaptive change of size and shape, robustness against local damage, and the possibility to switch between the aggregated (multicellular) and distributed (swarm of single-cellular) forms of existence.

Physically, chemical robots could be realized as internally structured particles in the size range up to tens of micrometers [5, 6], comparable to single-cell organisms such as protozoa. The body of a chemical robot can be divided into several internal compartments in which chemical substances can be stored (chemical payload) and released in response to an external stimulus. In analogy to vacuoles found in living cells, these compartments can be formed from a variety of materials ranging from liposomes to hollow-core mesoporous silica microparticles [7-14]. Besides defining the shape and size of the robot, the semipermeable shell controls the diffusion rate of molecules (chemical payload) between the internal volume and the outside environment. The outer shell can be further chemically modified to allow selective binding to specific substrates. Finally the permeability of the inner reservoirs and the outer skin should be a function of the intensive variables of the environment (such as temperature, pH, etc.) controlling the release rate of given chemical cargo.

Chemical swarm robots are designed to fulfill specific goals related to their chemical activity, such as the localization and selective binding to a given target substrate, controlled release of a chemical payload in the target area, and absorption or elimination of chemical contaminants present in their environment. In targeted drug delivery, a drug encapsulated inside the bodies of chemical swarm robots can be carried towards the required tissue by the blood stream and selectively bind to it via ligand-receptor interactions. An external stimulus such as threshold concentration of a specific substance dispersed in the target tissue causes structural changes of the inner or outer membranes of chemical robots, resulting in drug release. Such strategy for targeted drug release would be highly desirable for example in the case of chemotherapy for cancer treatment as it would enable to use of highly potent drugs while reducing the negative side effects.

A key feature of individual chemical robots is their ability to move independently in their environment. The synthesis of self-propelled colloidal particles capable to act collectively in applications such as chemical sensing and bio-sensing [15, 16], particle assembly, and targeted drug delivery [7-12, 17-19] represents an area of immense scientific and technological interest. When, for example, a target (e.g., a small colony of pathogens) needs to be localized, pure diffusive transport of colloidal particles by means of random Brownian motion becomes inefficient, especially in porous environments characterized by aberrant branching and high tortuosity of the available pore space [20]. To either increase the fraction of the particles that reach the target or to decrease the overall time required for the target to be localized, the use of alternative propulsion methods needs to be explored. The task of target localization by colloidal particles leads to two fundamental questions: (i) how to coordinate the collective motion of the particles and (ii) how to boost the movement of each individual particle in the correct direction?

The migration of colloidal particles in a fluid medium can be enhanced either passively by involving an external force field (e.g., fluid flow or magnetic field) or actively by employing a self-propulsion process [21]. A large number of self-propelled artificial nano- and micro-particles were reported recently, ranging from bubble-based micro-tube rockets [22], asymmetric surface nano-catalysts such as bi-metallic nano-swimming rods [23] or Janus particles [24], bio-molecule-powered chemical swimmers [25], devices based on chemical oscillations [26, 27], etc. Among various processes of particle self-propulsion, phoretic effects that occur at the particle surface as a response to local concentration gradients [28-31] are of particular interest because they allow the particles to act collectively by coordinating their actions through chemical signals. Regardless of the actual mechanism used, the phoretic velocity is usually considered to be proportional to the concentration gradient of the chemical signals which can be either produced by a chemical reaction at the particle surface [21, 32] or released from the internal volume of the particles (triggered release) in response to an external stimulus (e.g. change in temperature or concentration, presence of radiation, etc.).

The topology of the environment in which the chemical robots operate is an important factor to consider because it strongly affects both the spatial distribution of the robots and the diffusion of chemical signals. Therefore, a random porous medium [33] in which the robots are supposed to locate a target is introduced. The robots communicate by releasing the chemical signals in the proximity of the target. The chemical signals serve as chemo-attractant by means of the diffusiophoresis effect.

The present work is concerned with computer-aided design methodology for chemical swarm robots which is based on mathematical modeling of their

collective behavior in an environment with complex topology, taking into account all relevant physical laws. The design of chemical swarm robots is a parametrically rich problem, therefore computational modeling is necessary in order to systematically investigate the effect of key parameters such as the particle size, diffusion coefficient of the chemical signals, the trigger threshold for signal release, or the number of robots present in the target area, on the success of the mission prior to its attempted physical realization. The output characteristics of such numerical simulation are (i) the percentage of chemical robots that reach the target within a specified time (the target localization success rate), (ii) the statistical distribution of target localization times and (iii) the statistical distribution of the target residence time (the overall time the robots spent in the target location) which can then be used as design or optimization criteria.

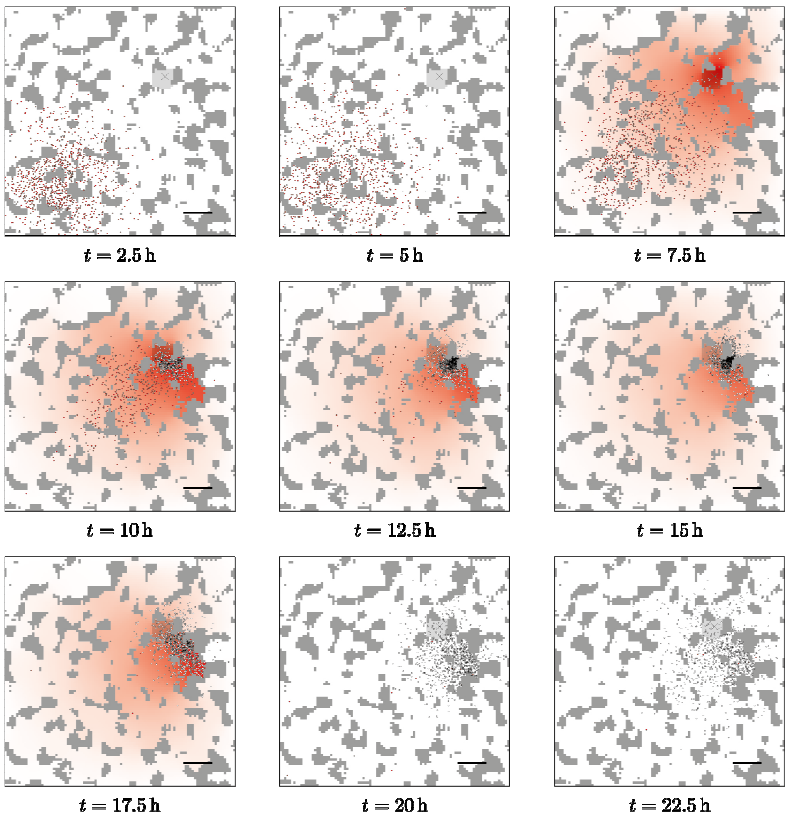


Figure 1. The progress of a target localization mission by chemical swarm robots in porous environment. The target is indicated by a cross and the overall target area Ω_T is shaded by the light grey color. The concentration of chemical signals in both the robots and the domain is indicated by the intensity of the red color. The parameters values are $\alpha = 0.56$ μm^2 s^{-1}, $\beta = 0.1$ s^{-1}, $D_S = 1000$ μm^2 s^{-1}. Scale bar, 50 μm.

In contrast to "classical" swarm systems where individual agents are either mechatronic devices or software bots [34-36], the mathematical concepts that are concerned with modeling of the collective behavior of chemical swarm robots take into account the physical nature of the microscopic chemical environment in which the robots operate via corresponding physical laws (e.g., random Brownian motion of the colloidal particles, diffusion of chemical signals, etc.). In general, chemical robots are considered to move in a liquid phase with the presence of solid particles, solid-liquid interfaces and membranes. It is somewhat logical that while decreasing the size of such functional devices, forces that are not normally considered in the design of classical robotics (such as adhesion forces, etc.) begin to dominate and strongly determine the behavior of the robots. This is also the reason why the modeling methodology differs significantly to those used in classical robotics and multi-agent systems. Therefore, the mutual interactions among robots may involve both attractive and repulsive forces, interfacial contacts, possibility of cluster formation, adsorption and chemical modification [37].

2 Simulated experiment

In the present work, the following scenario is considered: a 2-dimensional porous simulation domain contains a target. A group of chemical robots fully loaded with chemical signals is injected to the opposite side of the simulation domain. The robots are supposed to locate the target by a combination of random and oriented movements. Initially, the robots move exclusively by random Brownian motion. Once the first robot enters the proximity of the target it begins to release the chemical signals to attract its peers by means of the diffusiophoretic mechanism. Subsequent motion of the robots then depends on the balance of random and oriented movements and the robots may or may not reach the target as coordinated swarm. An example of such target localization mission by chemical swarm robots is displayed in Figure 1. The governing equations for the movement of individual robots, the release kinetics and diffusion of chemical signals, and for the simulation domain topology are described in Section 3.

For each robot, the following information is recorded: (i) the target localization time t_L, i.e., the time, when the robot entered the proximity of the target for the first time and (ii) the target residence time t_R, i.e., the total time the robot spent in the target location. For the overall population of chemical robots, the target localization and residence times are represented by distribution functions. Based on these distributions, three measures are introduced: (i) the mode target localization time $\bar{t}_L$, defined as the maximum on

the target localization time density function, corresponding to the time when the main swarm entered the proximity of the target, (ii) the success rate u_∞, corresponding to the limit of the cumulative target localization distribution function, representing the total fraction of robots that reached the target during the simulation and (iii) the mean target residence time $\overline{t}_R$, defined as the mean value on the residence time density distribution function, corresponding to the average time the particles spent in the proximity of the target. For details see Figure 2.

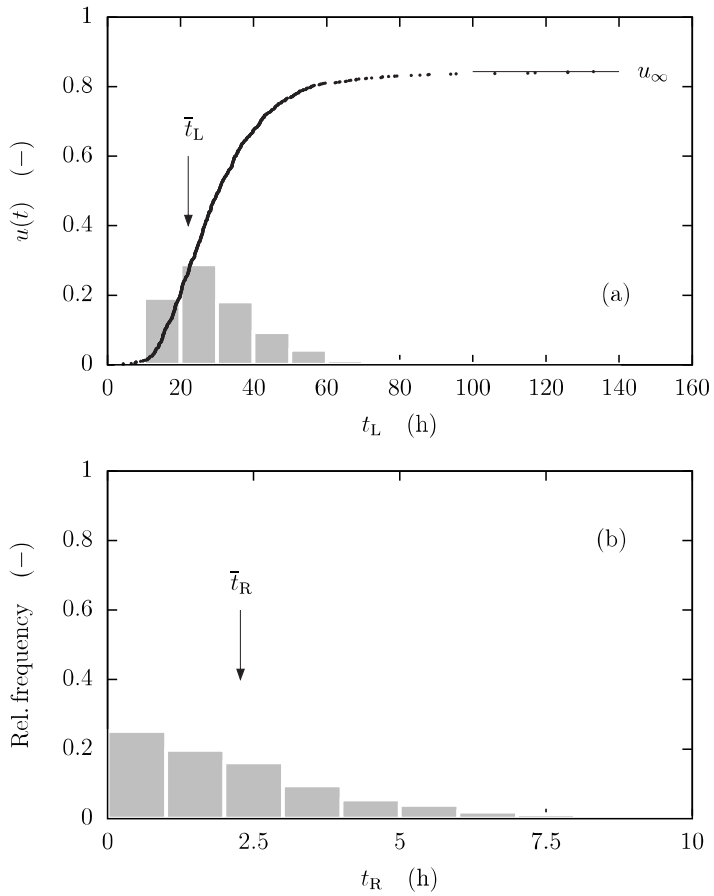


Figure 2. The target localization (a) and residence (b) time distributions with corresponding performance criteria – the mode target localization time $\overline{t}_L$, the target localization success rate u_∞ and the mean target residence time $\overline{t}_R$.

3 Modeling methodology

In order to model the events that determine the collective behavior of chemical swarm robots a combination of discrete (to calculate the motion of individual robots) and continuous (to calculate the actual concentrations of chemical signals) modeling methods is used. Both methods are coupled via the chemotaxis phenomena (e.g., diffusiophoresis) in which certain number of the robots release chemical signaling substance from their interior, while others move against the concentration gradient of the released chemical signaling substance.

3.1 Motion of the robots

The robots can move only within the available pore space and are allowed to leave the simulation domain eventually. When the effects of hydrodynamic flow field in the liquid medium arising from the motion of the robots and the mutual interactions between the robots as well as with the solid walls present in the domain are neglected, the Langevin equation for the position $\boldsymbol{x}(t)$ of a single robot within the simulation domain becomes

$$\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = \boldsymbol{v}(t, \boldsymbol{x}) + \boldsymbol{r}(t), \tag{1}$$

where $\boldsymbol{v}(t, \boldsymbol{x})$ is the propulsion velocity caused by chemical signals and $\boldsymbol{r}(t)$ is the random displacement due to random Brownian motion of the robots. The random displacement term $\boldsymbol{r}(t)$ is generated as Gaussian white noise with zero mean and the correlation function $\left\langle r_i(t) r_j(t') \right\rangle = 2D\delta_{ij}\delta(t - t')$, where D is the translational diffusion coefficient of the robot that can be related to the robot's hydrodynamic radius R via Einstein-Stokes equation

$$D = \frac{k_\mathrm{B} T}{6\pi\eta R}, \tag{2}$$

where η represents the dynamic viscosity of the surrounding liquid medium, k_B is the Boltzmann's constant and T the thermodynamic temperature.

Although equation (1) originates from Newton's second law of motion it does not contain any acceleration term. This is due to the fact that colloidal and micro-scale particles (such as the chemical robots considered here) exhibit instantaneous loss of inertial memory caused by large solvent damping when dispersed in a liquid medium [38, 39]. For a detailed derivation see Appendix 1. By definition, the first term of the right-hand side of equation (1) applies only within the simulation domain, i.e., where the information on the chemical signals concentration gradient is provided. However, robots that left the

simulation domain remain subject to random Brownian motion and may eventually return to the domain.

The propulsion velocity of the robots caused by the chemo-attractant (chemical signal) $\boldsymbol{v}(t, \boldsymbol{x})$ is linearly proportional to the logarithm of the concentration gradient [28, 40]

$$\boldsymbol{v}(t,\boldsymbol{x}) = -\alpha \cdot \nabla \ln c_S(t,\boldsymbol{x}), \tag{3}$$

with the proportionality constant α often referred to as the diffusiophoretic mobility [24, 25, 28, 40] or the gradient-sensing strength [41]. The diffusiophoretic mobility α is defined such that the units are those of a translational diffusion coefficient D, which makes their values directly comparable. Numerous expressions exist to calculate the diffusiophoretic mobility reflecting the actual phoretic mechanism. In the case of electrolytes (fast diffusing salts), the diffusiophoretic mobility scales according to, $\alpha \sim k_B T/(\eta\, l_B)$ adopting values of $\alpha \sim 10^{-9}$ m^2 s^{-1}, where l_B is the Bjerrum length, typically $l_B \sim 10^{-9}$ m [25, 28-30].

Table 1. List of the simulation parameter values.

Parameter	**Value**
Size of the simulation domain	100 × 100 cells
Cell size, Δx	4 μm
Number of robots, N	1000
Radius of robots, R	1 μm
Diffusion coefficient of the robots, D	0.22 μm^2 s^{-1}
Time step, Δt	0.05, 0.5 s
Maximum simulation run-time, t^{max}	5×10^5 s
Dynamic viscosity of the surrounding liquid medium, η	10^{-3} Pa s
Diffusiophoretic mobility, α	0.01 – 100 μm^2 s^{-1}
Diffusion coefficient of chemical signals, D_S	10 – 3162 μm^2 s^{-1}
Chemical signal release rate constant, β	0.01 – 1 s^{-1}
Chemical signal concentration in a single robot, c_S^{init}	10^{-5} – 100 mol dm^{-3}
Surface fraction of the robots, φ	0.196
Initial distance from target, d	255 μm

3.2 Diffusion problem

The information on the time dependence of the concentration gradients of the chemical signals in the domain is obtained by the numerical solution of the diffusion problem

$$\frac{\partial c_S(t,\boldsymbol{x})}{\partial t} = D_S \nabla^2 c_S\left(t,\boldsymbol{x}\right) + S\left(t,\boldsymbol{x}\right), \tag{4}$$

where D_S is the diffusion coefficient of the chemical signals and $S(t, \boldsymbol{x})$ is a location-dependent source term representing the release rate of the chemical signals from the bodies of the robots. Equation 4 is accompanied by the boundary conditions

$$c_S(t, \boldsymbol{x} \in \Omega_{ext}) = 0, \tag{5}$$

and

$$\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla c_S(t, \boldsymbol{x} \in \Omega_{int}) = 0, \tag{6}$$

denoting the absence of chemical signals outside the domain Ω_{ext} and zero-flux at the solid walls Ω_{int}, respectively, and the initial condition

$$c_S(t = 0, \boldsymbol{x}) = 0, \tag{7}$$

denoting the absence of chemical signals within the domain at the beginning of the simulation.

Each robot can be present in two possible states, either an "on" or "off" state. Initially, all the robots are present in the "off" state and are fully loaded with the chemical signals. If a robot passes through the proximity of the target (defined as the target area Ω_T, see Appendix 2 for definition), it changes its state irreversibly into an "on" state causing the release of chemical signals. The release of the chemical signals is described by the first order kinetics

$$\frac{dc_{S,i}\left(t\right)}{dt} = -\beta\left[c_{S,i}\left(t\right) - c_S\left(t,\boldsymbol{x}\right)\right], \tag{8}$$

where β is the chemical signal release rate constant, $c_{S,i}(t)$ is the concentration of chemical signals within the body of i-th robot and $c_S(t, \boldsymbol{x})$ is the signal concentration at the robot's location. Equations (4) and (8) are coupled through the source term $S(t, \boldsymbol{x})$. Once equation (4) is spatially discretized, the source term is obtained by a superposition of the chemical signal release rates from all robots present in the j-th cell

$$S_j\left(t,\boldsymbol{x}\right) = -\sum_i \varphi \frac{dc_{S,i}\left(t\right)}{dt} \quad \forall i : \boldsymbol{x}_i \in j\text{-}th\ cell, \tag{9}$$

with φ being the ratio of an area occupied by single robot compared to the area of a single cell.

3.3 Simulation domain

In the present work, the robots are considered to move within a porous environment. It is therefore necessary to introduce a general definition of the simulation domain. The 2-dimensional simulation domain Ω is discretized by

means of the linked-cell method [42], with two types of cells defined by the phase function $\phi(\boldsymbol{x})$

$$\varphi(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \text{pore,} \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The method of Gaussian-correlated random porous media [33] is employed to generate the positions for each type of the cells. Let $X(\boldsymbol{x})$ be a set of independent normally distributed random variables defined on a discrete square grid (each grid point is cell-centered) with spatially periodic conditions. A Gaussian-correlated random field $Y(\boldsymbol{x})$ with correlation length L is then constructed from $X(\boldsymbol{x})$ by applying the linear filter

$$Y(\boldsymbol{x}_C) = \sum_{|\boldsymbol{x}-\boldsymbol{x}_C| \le L} X(\boldsymbol{x}) exp[(\boldsymbol{x}-\boldsymbol{x}_C)^2 / L^2], \tag{11}$$

and renormalization [43]. The process is then repeated several times and the resulting normalized random field corresponds (represents the values of) to the phase function $\phi(\boldsymbol{x})$.

The random porous domain can be characterized by two measures: (i) the porosity ε, defined as the ratio of the empty (pore) cells over the total number of cells within the domain and (ii) the tortuosity τ, defined as the effective path length connecting two randomly selected points within the domain compared to their Euclidean distance. The set of empty cells within the simulation domain is further divided into two subsets: the initial position area Ω_I and the target area Ω_T. See Appendix 2 for detailed explanation, how these areas are defined.

Table 1 shows the values of the input parameters, used for generating the results discussed below.

4 Results and discussion

In the following section, results obtained from the simulated experiment are presented and discussed in terms of the selected performance criteria with respect to the model parameters that can be adjusted during the design of real chemical robots. The results section begins with results obtained on a completely empty and open domain ($\varepsilon = \tau = 1$) and these results are later compared to those obtained on an open domain containing geometrical obstacles, generated by employing the method of Gaussian-correlated random porous medium described above.

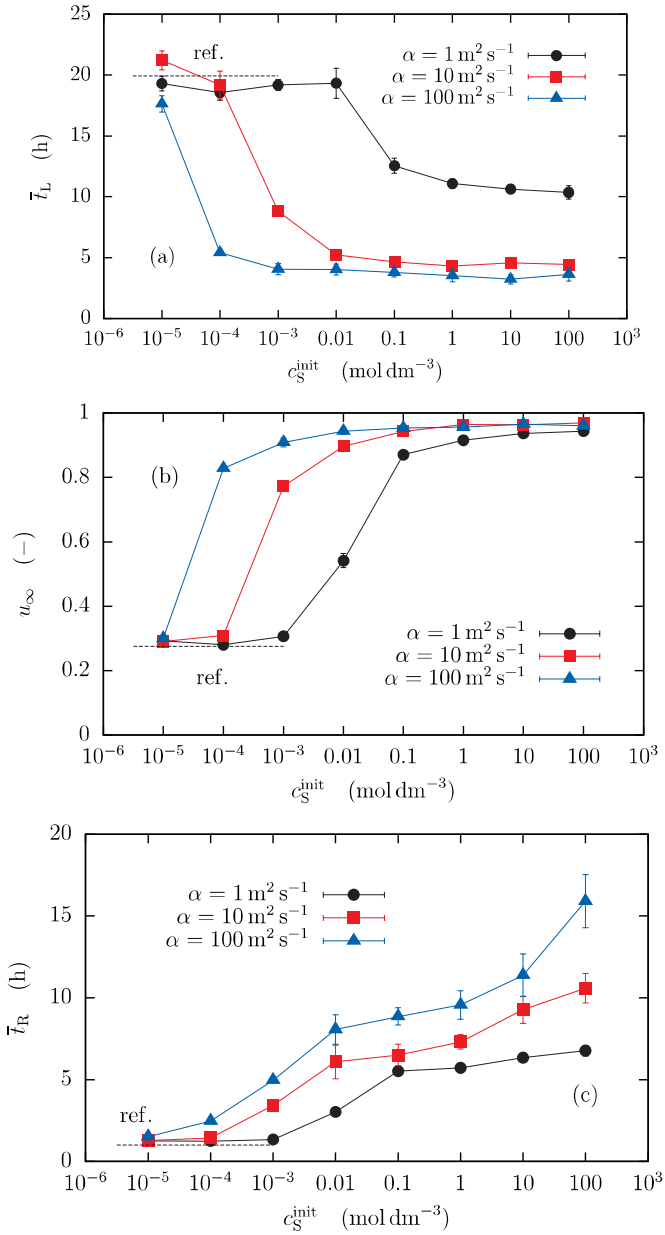


Figure 3. The dependence of the performance criteria on the initial chemical signal concentration $c_\mathrm{S}^\mathrm{init}$ recorded within an empty simulation domain: (a) the target localization time $\bar{t}_\mathrm{L}$, (b) the target success rate u_∞ and (c) the target residence time $\bar{t}_\mathrm{R}$. The dashed lines correspond to the reference value (ref.) obtained in the absence of chemical signals. The remaining model parameters are $\beta = 1$ s^{-1} and $D_\mathrm{S} = 100$ μm^2 s^{-1}.

The movements of the chemical robots are coordinated via chemical signals released in response to external stimuli. The understanding of factors that control the diffusivity of both the robots and the chemical signals is one of the most important tasks for computer-aided design of real chemical robots. Tackling the problem of diffusivity represents an optimization problem of finding a desired type of the chemical robots' behavior determined by the values of parameters that contribute to the overall diffusivity: (i) the chemical signal release rate constant β, i.e., the rate by which the chemical signals are released from the internal compartments of the robots, (ii) the diffusion coefficient of the chemical signals D_S, i.e., the rate by which the chemical signals are spread and eventually mediate the mutual communication among the robots, (iii) the total amount of chemical signals available c_S^{init}, i.e., the initial concentration of the chemical signals within the bodies of the robots, (iv) the diffusiophoretic mobility α, i.e., the extent of response of the robots to concentration gradients of the chemical signals and (v) the geometrical complexity of the environment in which the robots operate, expressed via the ratio of porosity ε and tortuosity τ factors, ε/τ. For each parameter a parametric study (a series of simulations) was performed by varying its values within physically realistic limits while keeping the values of all remaining parameters fixed. The effect of each of the parameters is later analyzed by means of the selected performance criteria, defined above. The value ranges for the parameters were listed in Table 1.

4.1 The effect of the initial chemical signal concentration

Since the chemical signals are carried by the robots and released in response to external stimuli within the target location, it is important to note that the overall amount of the chemical signals available is limited. Therefore, the initial chemical signal concentration c_S^{init} represents one of the adjustable parameters to be considered.

Figure 3 shows the dependence of the performance criteria, i.e, the target localization time $\bar{t}_L$, the target localization success rate u_∞, and the target residence time $\bar{t}_R$ on the initial chemical signal concentration c_S^{init}. By varying the initial chemical signal concentration c_S^{init} through many orders of magnitude, sigmoidal curves of the target localization time $\bar{t}_L$ and target success rate u_∞ are obtained. The asymptotes represent a limiting behavior that occurs either at extremely low or high initial concentrations of the chemical signals. The exact position of the inflection point on these curves depends on diffusiophoretic mobility factor α. The larger the diffusiophoretic mobility, the

lower concentration of chemical signals is required for a successful target localization mission by chemical swarm robots to take place.

Slightly different in shape are the curves of the dependence of target residence time $\bar{t}_R$ on the initial concentration of the chemical signals c_S^{init}. It can be stated that larger amounts of the chemical signals are increasing the overall time when the robots remain in the target location. From the dependence in Figure 3c it is not possible to conclude whether a limit for large values of the initial concentration of the chemical signals exists.

4.2 The effect of the diffusiophoretic mobility

The diffusiophoretic mobility α can be understood as a measure of the ability of the robots to sense and respond to the chemical signals. Therefore, values of the diffusiophoretic mobility α that are similar or smaller to the actual diffusion coefficient of the robots ($D = 0.22\ \mu m^2\ s^{-1}$) are causing the performance criteria (the target localization time $\bar{t}_L$ and the target localization success rate u_∞) to become comparable to those recorded in the absence of chemical signals. This "breaking point" is clearly visible in Figure 4a. The position of this point in the dependence of the target localization time $\bar{t}_L$ on the diffusiophoretic mobility α can be adjusted by the chemical signal release rate constant β. On the other hand, large values of the diffusiophoretic mobility α result into a behavior that reaches certain limit represented by the asymptote in Figure 4a. Figure 4b confirms a very similar dependence of another performance criterion, the target localization success rate u_∞. Clearly, the steeper and more durable the chemical signal concentration gradient (with the steepness being controlled by the interplay of the chemical signal release rate constant β and the chemical signal diffusion coefficient D_S) the more efficient becomes the signaling strategy. The effect of the chemical signal release rate constant β is important only for large values of the initial chemical signal concentration c_S^{init}. Once c_S^{init} decreases to lower values the effect of β is no longer significant.

A very interesting behavior is observed in the case of the target residence time $\bar{t}_R$. Figure 4c shows that there is a certain range for the values of the diffusiophoretic mobility α for which the target residence time can be increased significantly. The values outside this range do not have desired influence on the target residence time $\bar{t}_R$. This leads to an important observation that is very often intrinsic to complex systems, such as the system of chemical swarm robots present here. It is not always possible to satisfy all the possible performance criteria. This "trade-off" leads to an important conclusion for the design of real chemical robots: a decision must be made whether it is more important to deliver sufficient amount of the chemical cargo carried by the

robots into the target location within the shortest time possible or to design the robots to be able to spent much longer time scales in the target location that may allow them to execute their pre-programmed chemical operations.

Furthermore, by changing the values of remaining parameters that control the chemical signal concentration gradient shape and time of duration (such as the chemical signal release rate constant β) both the position and the magnitude of the maximum in the dependence of the target residence time on the diffusiophoretic mobility in Figure 4c can be affected significantly.

The shape of the dependence of the target residence time on the diffusiophoretic mobility in the case of the parameter values $\beta = 0.1\ s^{-1}$ and $D_S = 1000\ \mu m^2\ s^{-1}$ can be explained by the following consideration: at low values of the diffusiophoretic mobility $\alpha < 0.1\ \mu m^2\ s^{-1}$, the target residence time tends to decrease to the value obtained when no signals are present. On the other hand, when the oriented movements begin to dominate the motion of the robots over the random movements, the target residence time tends to increase. However, at certain critical value of the diffusiophoretic mobility $\alpha > 10\ \mu m^2\ s^{-1}$ the motion of the robots becomes completely controlled by the oriented movements caused by the chemical signals. Since the chemical signals are released once the robots pass through the target area (i.e., at the boundary of the target area) and their response to the chemical signals is relatively strong, a fraction of the swarm may eventually appear outside the target area which leads to overall decrease of the target residence time. Very similar considerations can be expected in the case of the other two curves in Figure 4c.

4.3 The effect of the chemical signal release rate constant

Figure 4a shows the localization time $\bar{t}_L$ recorded at various values of the signal release rate constant β: for large values of the diffusiophoretic mobility α no effect at all can be observed while varying the value of the chemical signal release rate constant β. However, once the value of the diffusiophoretic mobility α becomes comparable to the diffusion coefficient D of the robots themselves ($D = 0.22\ \mu m^2\ s^{-1}$), larger values of the chemical signal release rate constant $\beta \sim 1\ s^{-1}$ are causing an undesired behavior of the robots that is observable for both measures, the target localization time $\bar{t}_L$ and the success rate u_∞. The overall time required for the robots to reach the target area becomes prolonged as a consequence of the almost equal contributions to their motion that arise from the random and oriented movements. While similarity of the values of α and D_S can lead to the overall increase in the performance of the chemical swarm robots when the signals are released at mid distances between

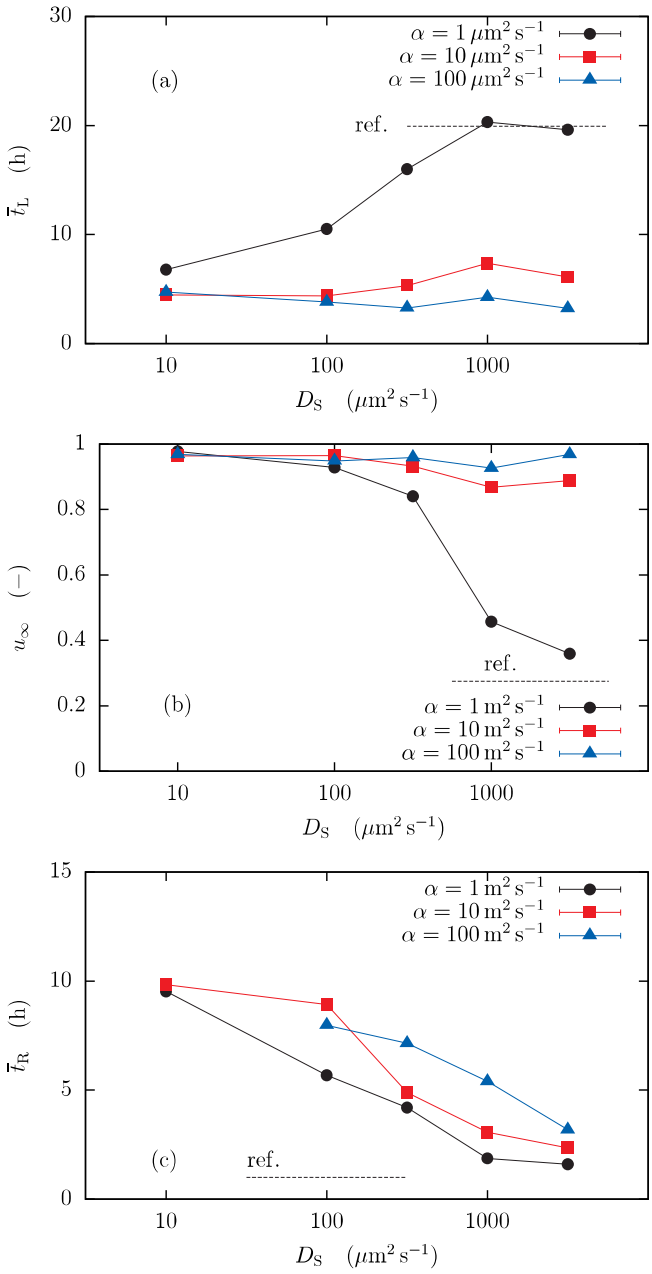


Figure 5. The dependence of the performance criteria on the chemical signal diffusion coefficient D_S recorded within an empty simulation domain: (a) the target localization time $\bar{t}_\mathrm{L}$, (b) the target success rate u_∞ and (c) the target residence time $\bar{t}_\mathrm{R}$. The dashed lines correspond to the reference value (ref.) obtained in the absence of chemical signals. The values of the remaining parameters are $\beta = 1\ \mathrm{s}^{-1}$ and $c_\mathrm{S}^\mathrm{init} = 100\ \mu\mathrm{m}^2\ \mathrm{s}^{-1}$.

the initial and target locations [2], when the signals are released in the target location the performance of the robots tends to decrease. Similarly, at the same values of the diffusiophoretic mobility α, the target localization success rate u_∞ decreases (cf. Figure 4b).

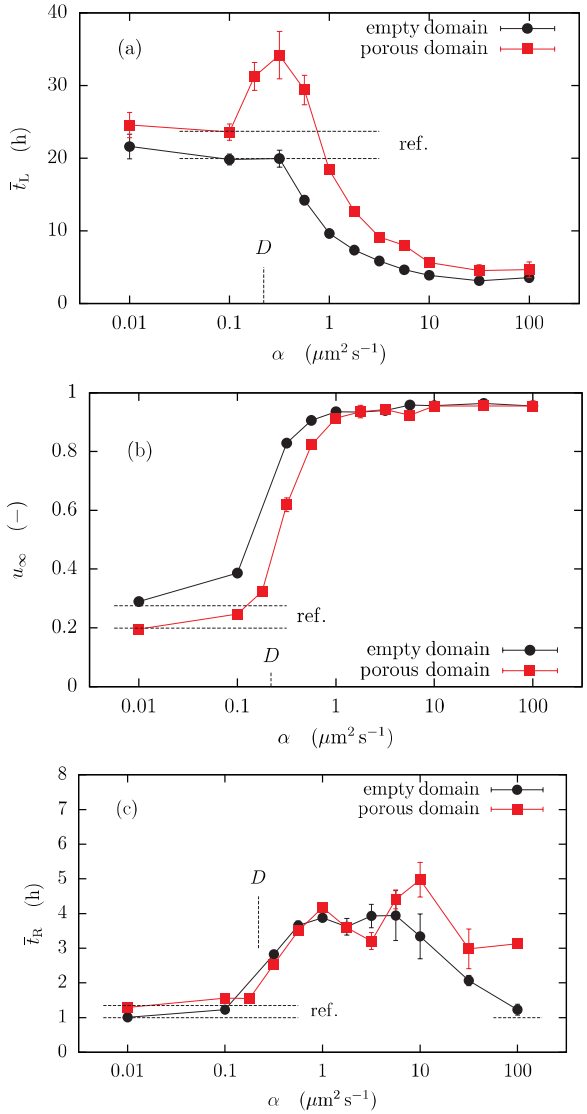


Figure 6. Comparison of the dependence of the performance criteria on the diffusiophoretic mobility α recorded within an empty and porous simulation domain: (a) the target localization time $\bar{t}_L$, (b) the target success rate u_∞ and (c) the target residence time $\bar{t}_R$. The dashed lines correspond to the reference value obtained in the absence of chemical signals (ref.) and to the diffusion coefficient of the chemical robots D. The values of the additional parameters are c_S^{init} = 100 μm^2 s^{-1}, β = 0.1 s^{-1} and D_S = 1000 μm^2 s^{-1}.

4.4 The effect of the chemical signal diffusion coefficient

For the purpose of generality, a relatively wide value range of the chemical signal diffusion coefficient D_S is considered in the following text. Figure 5a shows that the target localization time $\bar{t}_L$ can be significantly decreased when the signal diffusion coefficient D_S decreases. This decreasing effect is the most significant in the case of weak diffusiophoretic mobility, $\alpha = 1$ μm^2 s^{-1}, and fast chemical signal release rate, $\beta = 1$ s^{-1}. Medium and high values of the diffusiophoretic mobility ($\alpha = 10, 100$ μm^2 s^{-1}) together with medium and low chemical signal rate constants ($\beta = 0.01, 0.1$ s^{-1}) affect the target localization time $\bar{t}_L$ only weakly.

An analogous situation can be observed in the case of the target localization success rate u_∞: the same values of the parameters that decrease the target localization time $\bar{t}_L$ increase the target localization success rate u_∞ (cf. Figure 5b). For medium and high values of the diffusiophoretic mobility α and high values of the chemical signal release rate β, the influence of the diffusion coefficient of chemical signals D_S is negligible for both, the target localization time $\bar{t}_L$ as well as the target localization success rate u_∞.

On the other hand, the decrease of value of the chemical signal diffusion coefficient D_S tends to increase the target residence time $\bar{t}_R$ in almost all observed cases (cf. Figure 5c).

However, the readers should be aware of the fact that only a very narrow range of the chemical signal diffusion coefficient D_S is valid for the diffusiophoresis effect with respect to electrolytes to be the driving method of the robots self-propulsion. This fact has two important consequences: first, the realistic values of the chemical signal diffusion coefficient D_S are around 1000 μm^2 s^{-1} (relatively narrow interval) and second, the diffusiophoretic mobility α measured for colloids with the hydrodynamic radius of 100 nm is in the range of several hundred (using the same units) which corresponds to a relatively strong self-propulsion [29, 30, 40]. Taking this into account, the overall effect of the chemical signal diffusion coefficient D_S could be considered as weak at low values of the diffusiophoretic mobility α and high values of the chemical signal release rate β, no matter what is its actual value. In fact, the presence of the signal itself is beneficial enough when compared to target discovery realized purely by chance by means of the random Brownian motion of the robots.

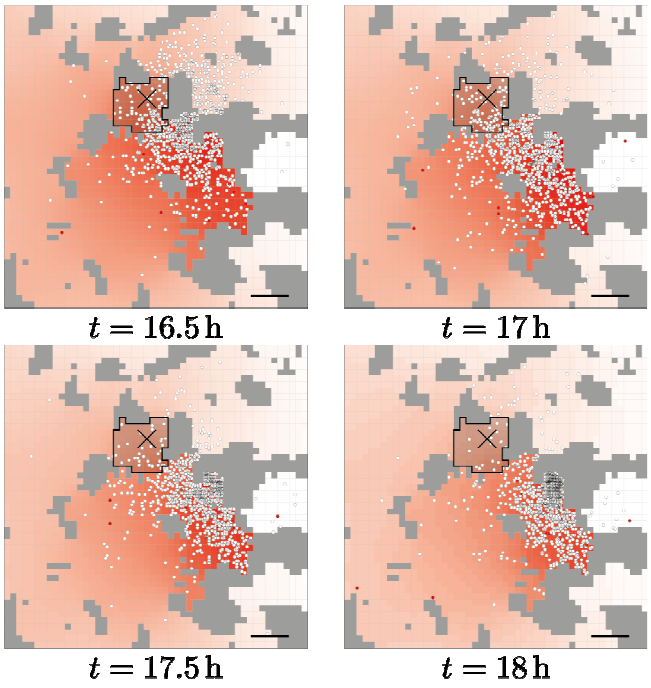


Figure 7. Example of the anomalous behavior of chemical swarm robots for large values of the diffusiophoretic mobility. The target area is outlined by the black solid line and the target is indicated by the black cross. The values of the parameters are $c_S^{init} = 100$ mol dm^{-3}, $\beta = 0.1$ s^{-1}, $\alpha = 3.2$ μm^2 s^{-1} and $D_S = 1000$ μm^2 s^{-1}. Scale bar, 25 μm.

4.5 The effect of the domain topology

So far the discussion was based on simulation results performed within a completely open (non-porous) domain. The dependencies of the performance criteria that were recorded within an open domain can be considered as some form of an ideal system. However, in real world, the robots are expected to operate often in porous environments where various geometrical obstacles such as solid-liquid interfaces are present. In such case, the observed behavior deviates from the one in open domain. This fact will be in further illustrated on the performance criteria with respect to the diffusiophoretic mobility recorded using a porous domain.

It is important to note that the initial distance from the target in the porous domain is chosen such that the shortest path to the target via which the robots are supposed to travel has the same length as the Euclidean distance between the target and initial positions of the robots in an empty domain discussed in the previous sections. Consequently, the robots are expected to use the chemical

signals to navigate their peers in the correct direction by performing a series of correct turns to overcome the geometrical obstacles in the porous domain. The number of the required correct turns can be related to the tortuosity τ of the simulation domain. Hence, the higher the tortuosity τ of the porous domain the larger number of correct turns is required for the robots to successfully localize the target.

Figure 6a shows significant deviations on the dependence of the target localization time $\bar{t}_L$ on the diffusiophoretic mobility α recorded within a porous domain with porosity $\varepsilon = 0.76$ and tortuosity $\tau = 2.2$ ($\varepsilon/\tau = 0.35$) from the ideal behavior recorded in an open domain. There is a maximum approximately at $\alpha \sim 0.3$ μm^2 s^{-1} that indicates a significant delay in the target localization time of the robots. Further, the inflection point is shifted towards higher values of the diffusiophoretic mobility α.

The presence of the solid walls affects the shape and time duration of the chemical signal concentration gradients. For the diffusiophoretic mobility values comparable to the robots' diffusion coefficient ($D = 0.22$ μm^2 s^{-1}), it becomes highly probable that the robots will be attracted to a blind-end corner within the porous domain where the signals tend to accumulate. A large number of the robots can become mislead by the signals and migrate to a wrong direction causing significant delays in the target localization time. By reviewing the dependence of the target localization success rate u_∞ on the diffusiophoretic mobility α one realizes that although the time required by the robots to localize the target area increased, the fraction of the robots that entered the target area remained unaffected (cf. Figure 6a-b).

Figure 6c shows the dependence of the target residence time $\bar{t}_R$ on the diffusiophoretic mobility α recorded in a porous domain. The rapid decrease of the target residence time at the diffusiophoretic mobility values around 3 μm^2 s^{-1} can be explained by the following consideration: due to the presence of a blind-end corner close to the target area, the chemical signals tend to accumulate at the solid walls of the corner. Figure 7 shows that although the robots begin to release the chemical signals in the target area, by following the chemical signal concentration gradients they eventually end up in the near-by blind-end corner. The chemical signals released from the newly arriving robots create a positive feedback that reinforces the accumulation of robots in the blind-end corner. Consequently, it becomes extremely difficult for the swarm to leave its position, as it has to wait for the chemical signals to diffuse away. Although this phenomenon prolongs the overall time the robots spend in their location, it may or may not be beneficial for the target localization mission depending on where the position of such blind-end corner appears.

5 Conclusions

The mathematical methodology suitable for modeling of the collective behavior of chemical swarm robots was introduced. The modeling methodology allows capturing of all the necessary physical concepts valid for modeling the motion of micro-scale particles in porous environment filled with a liquid medium.

The mathematical model was used for solving a target localization problem by chemical swarm robots, a problem considered to be of immense practical importance for many of the expected applications. The results demonstrate that even a simplistic approach (by neglecting mutual interaction among the robots and between the robots and solid walls) can lead to complex modes of behavior of the robots, a common feature of systems composed of large number of relatively simple agents.

A parametric study with respect to parameters that affect the diffusivity of the robots (the initial concentration of the chemical signals within the bodies of the robots, the chemical signal release rate constant, the chemical signal diffusion coefficient, the diffusiophoretic mobility and the geometrical complexity of the porous medium) was performed. The study reveals that by adjusting the parameters of the model has a significant impact on the behavior of the robots. In general, all parameters values that result in durable and steep chemical signal concentration gradients can be considered to be beneficial for at least one of the selected performance criteria.

Further, it has been shown that with respect to the chosen performance criteria a "trade-off" principle exists that needs to be taken into account during the design of real chemical swarm robots. This may lead to immense implications for the design of real chemical swarm robots with respect to their expected applications.

The results obtained within a porous domain shown increasing probability of the occurrence of anomalous behavior that may eventually lead to a significant decrease of the robots' performance. An important conclusion here is the fact that for a successful target localization mission to take place, the properties of the environment where the robots are supposed to operate must be taken into account during the design of the robots.

Appendix 1

The translational motion of *i*-th robot in a stagnant liquid phase is described by the Langevin equation [44, 45]

$$m_i \frac{d^2 \boldsymbol{x}_i(t)}{d t^2} = -\xi_i \frac{d \boldsymbol{x}_i(t)}{d t} + \boldsymbol{F}_i^C(\boldsymbol{x},t) + \sum_{j \neq i} \boldsymbol{F}_{j,i}^I(\boldsymbol{x},t) + \boldsymbol{F}_i^R(t), \tag{12}$$

Where m_i and $\boldsymbol{x}_i(t)$ is the robots mass and its position vector, respectively. The friction coefficient ξ_i is determined by the Stokes' law, $\xi_i = 6\pi\eta R_i$, where η is the dynamic viscosity of the surrounding liquid and R_i is the radius of i-th robot [46]. According to Newtons' second law the motion of i-th robot is a response to the forces acting on it. In general the forces acting on i-th robot involve the diffusiophoresis (chemotaxis) force $\boldsymbol{F}_i^{\mathrm{C}}(\boldsymbol{x},t)$, the robot-robot interaction force $\boldsymbol{F}_{j,i}^{\mathrm{I}}(\boldsymbol{x},t)$ and the dissipative random force $\boldsymbol{F}_i^{\mathrm{R}}(t)$, reflecting the effects of the solvent molecules.

For colloidal and micro-scale particles, the acceleration term on the right-hand side of equation (12) can be neglected due to large solvent damping causing instantaneous loss of inertial memory

$$m_i \frac{\mathrm{d}^2 \boldsymbol{x}_i(t)}{\mathrm{d}t^2} = 0. \tag{13}$$

Equation (12) then reduces to a first-order stochastic differential equation [38, 39]

$$\frac{d\,\boldsymbol{x}_i(t)}{d\,t} = \frac{1}{\xi_i}\left(\boldsymbol{F}_i^{V}(\boldsymbol{x},t) + \sum_{j\neq i} \boldsymbol{F}_{j,i}^{I}(\boldsymbol{x},t) + \boldsymbol{F}_i^{R}(t) \right). \tag{14}$$

By neglecting the mutual interactions between the robots, $\sum_{j\neq i} \boldsymbol{F}_{j,i}^{I}(\boldsymbol{x},t)/\xi_i = 0$, and replacing the remaining terms on the right-hand side of equation (14) by $\boldsymbol{v}_i(\boldsymbol{x},t) = \boldsymbol{F}_i^{\mathrm{C}}(\boldsymbol{x},t)/\xi_i$ and $\boldsymbol{r}_i(t) = \boldsymbol{F}_i^{\mathrm{R}}(t)/\xi_i$, equation (14) becomes

$$\frac{\mathrm{d}\,\boldsymbol{x}_i(t)}{\mathrm{d}t} = \boldsymbol{v}_i(\boldsymbol{x},t) + \boldsymbol{r}_i(t), \tag{15}$$

Since all the robots are considered to be of the same size, indexing can be avoided.

Appendix 2

The set of all empty cells within the simulation domain is divided into two subsets, the initial area Ω_{I} and the target area Ω_{T}. The process of division of the simulation domain is done in the following way: an empty cell in one corner of the domain is selected with the coordinates $\boldsymbol{x}_0$. This cell represents the inflow of a chemical substance that is further spread through the domain by diffusion process. Then a second empty cell is selected in the opposite side of the domain with the coordinates $\boldsymbol{x}_1$. This cell represents the outflow of the diffusing chemical substance. The identity of the substance is arbitrary; it only serves as a tool to measure distances within the domain. Then a diffusion problem is solved numerically until a steady-state concentration profile $c(\boldsymbol{x})$ is obtained

$$0 = \nabla^2 c(\boldsymbol{x}), \tag{16}$$

with the boundary conditions

$$c(\boldsymbol{x} = \boldsymbol{x}_0) = 1, \tag{17}$$

signifying the inflow of the chemical substance,

$$c(\boldsymbol{x} = \boldsymbol{x}_1) = 0, \tag{18}$$

signifying the outflow of the chemical substance and

$$\boldsymbol{n}(\boldsymbol{x}) \cdot \nabla c(\boldsymbol{x} \in \Omega_{int}, \Omega_{ext}) = 0, \tag{19}$$

signifying zero-flux at the internal Ω_{int} and external Ω_{ext} boundaries, respectively.

Once the steady-state concentration profile is obtained a path is generated starting at the inflow cell $\boldsymbol{x}_0$ and following steepest descent of the chemical substance concentration (in the direction of the outflow cell $\boldsymbol{x}_1$) by visiting neighboring cells in a step-by-step process. During each step the overall length of the path is incremented and the process ends when certain predefined value of the path length is obtained.

Finally, the initial Ω_{I} and target area Ω_{T} is then obtained by adding neighboring cells up to certain radius on both sides of the path. The robots are then randomly distributed within the initial area Ω_{I}.

Acknowledgements: This work has been supported by the European Research Council (project no. 200580-Chobotix).

References

[1] Grančič, P., Štěpánek, F.: *Chemical Swarm Robots*. Handbook of Collective Robotics – Fundamentals and Challenges, Pan Stanford Publishing (S. Kernbach, ed.) Singapore, 2011.

[2] Grančič, P., Štěpánek, F.: Active targeting in a random porous medium by chemical swarm robots with secondary chemical signaling. *Physical Review E*, **84** (2011), pp. 021925.

[3] Kessin, R.: *Dictyostelium: Evolution, Cell Biology, and the Development of Multicellularity*, Cambridge University Press, Cambridge, 2001.

[4] Ševčíková, H., Čejková, J., Krausová, L., Přibyl, M., Štěpánek, F. and Marek, M.: A new traveling wave phenomenon of dictyostelium in the presence of camp, *Physica D*, **239** (2010), pp. 879–888.

[5] Dohnal, J., Štěpánek, F.: Inkjet fabrication and characterisation of calcium alginate microcapsules, *Powder Technology*, **200** (2010), pp. 254–259.

[6] Čejková, J., Hanuš, J., Štěpánek, F.: Investigation of internal microstructure and thermo-responsive properties of composite

PNIPAM/silica microcapsules. *Journal of Colloid and Interface Science*, **346** (2010), 352-360.

[7] Fornasieri, G., Badaire, W., Backov, R., Mondain-Monval, O., Zakri, U., Poulin, P.: Mesoporous and homothetic silica capsules in reverse-emulsion microreactors. *Advanced Materials*, **16** (2004), pp. 1094-1097.

[8] Amstad, E., Reimhult, E.: Nanoparticle actuated hollow drug delivery vehicles. *Nanomedicine*, **7** (2012), pp. 145-164.

[9] Delcea, M., Yashchenok, A., Videnova, K., Kreft, O., Mohwald, H., Skirtach, A. G.: Multicompartmental micro- and nanocapsules: Hierarchy and applications in biosciences. *Macromolecular Bioscience*, **10** (2010), pp. 465-474.

[10] Wang, Y.J., Price, A.D., Caruso, F.: Nanoporous colloids: Building blocks for a new generation of structured materials. *Journal of Materials Chemistry*, **19** (2009), pp. 6451-6464.

[11] Malmsten, M.: Soft drug delivery systems. *Soft Matter*, **2** (2006), pp. 760-769.

[12] Stadler, B., Price, A. D., Chandrawati, R., Hosta-Rigau, L., Zelikin, A. N., Caruso, F.: Polymer hydrogels capsules: en route toward synthetic cellular systems. *Nanoscale*, **1** (2009), pp. 68-73.

[13] Peters, R. J. R. W., Louzao, I., van Hest, J. C. M.: From polymeric nanoreactors to artificial organelles. *Chemical Science*, **2** (2012), pp. 335-342.

[14] De Cock, L. J., De Koker, S., De Geest, B. G., Grooten, J., Vervaet, C., Remon, J. P., Sukhorukov, G. B., Antipina, M. N.: Polymeric multilayer capsules in drug delivery. *Angewandte Chemie – International Edition*, **49** (2010), pp. 6954-6973.

[15] Wu, J., Balasubramanian, S., Kagan, D., Manesh, K. M., Campuzano, S., Wang, J.: Motion-based DNA detection using catalytic nanomotors. *Nature Communications*, **1** (2010), pp. 36.

[16] Balasubramanian, S., Kagan, D., Hu, C. M. J., Campuzano, S., Lobo-Castanon, M. J., Lim, N., Kang, D. Y., Zimmerman, M., Zhang, L. F., Wang, J.: Micromachine-enabled capture and isolation of cancer cells in complex media. *Angewandte Chemie – International Edition*, **50** (2011), pp. 4161-4164.

[17] Allen, T. M., Cullis, P. R.: Drug delivery systems: Entering the mainstream. *Science*, **303** (2004), pp.1818-1822.

[18] Mohamed, F., van der Walle, C. F.: Engineering biodegradable polyester particles with specific drug targeting and drug release properties. *Journal of Pharmaceutical Sciences*, **97** (2008), pp. 71-87.

[19] Kagan, D., Laocharoensuk, R., Zimmerman, M., Clawson, C., Balasubramanian, S., Kong, D., Bishop, D. Sattayasamitsathit, S., Zhang,

L., Wang, J.: Rapid delivery of drug carriers propelled and navigated by catalytic nanoshuttles. *Small*, **6** (2010), pp. 2741-2747.
[20] Chilkoti, A., Dreher, M. R., Meyer, D. E., Raucher, D.: Targeted drug delivery by thermally responsive polymers. *Advanced Drug Delivery Reviews*, **54** (2002), pp. 613-630.
[21] Ebbens, S. J., Howse, J. R.: In pursuit of propulsion at the nanoscale. *Soft Matter*, **6** (2010), pp. 726-738.
[22] Kagan, D., Balasubramanian, S., Wang, J.: Chemically triggered swarming of gold microparticles. *Angewandte Chemie – International Edition*, **50** (2011), pp. 503-506.
[23] Paxton, W. F., Kistler, K. C., Olmeda, C. C., Sen, A., St Angelo, S. K., Cao, Y. Y., Mallouk, T. E., Lammert, P. E., Crespi, V. H.: Catalytic nanomotors: Autonomous movement of striped nanorods. *Journal of the American Chemical Society*, **126** (2004), pp. 13424-13431.
[24] Howse, J. R., Jones, R. A. L., Ryan, A. J., Gough, T., Vafabakhsh, R., Golestanian, R.: Self-motile colloidal particles: From directed propulsion to random walk. *Physical Review Letters*, **99** (2007), pp. 048102.
[25] Palacci, J., Abécassis, B., Cottin-Bizonne, C., Ybert, C., Bocquet, L.: Colloidal motility and pattern formation under rectified diffusiophoresis. *Physical Review Letters*, **104** (2010), pp. 138302.
[26] Tinsley, M. R., Taylor, A. F., Huang, Z., Showalter, K.: Complex organizing centers in groups of oscillatory particles. *Physical Chemistry Chemical Physics*, **13** (2011), pp. 17802-17808.
[27] Hara, Y., Maeda, S., Hashimoto, S., Yoshida, S.: Molecular Design and Functional Control of Novel Self-Oscillating Polymers. *International Journal of Molecular Sciences* **11** (2010), pp. 704-718.
[28] Anderson, J. L.: Colloid transport by interfacial forces. *The Annual Review of Fluid Mechanics*, **21** (1989), pp. 61-99.
[29] Abécassis, B., Cottin-Bizonne, C., Ybert, C., Ajdari, A., Bocquet, L.: Boosting migration of large particles by solute contrasts. *Nature Materials* **7** (2008), pp. 785.
[30] Abécassis, B., Cottin-Bizonne, C., Ybert, C., Ajdari, A., Bocquet, L.: Osmotic manipulation of particles for microfluidic applications. *New Journal of Physics*, **11** (2009), pp. 075022.
[31] Brady, J. F.: Particle motion driven by solute gradients with application to autonomous motion: Continuum and colloidal perspectives. *Journal of Fluid Mechanics*, **667** (2011), pp. 216-259.
[32] Ibele, M. E., Mallouk, T. E., Sen, A.: Schooling Behavior of Light-Powered Autonomous Micromotors in Water. *Angewandte Chemie – International Edition*, **48** (2009), pp. 3308-3312.
[33] Adler, P., Thovert, J.: Real porous media: Local geometry and

macroscopic properties. *Applied Mechanics Reviews*, **51** (1998), pp. 537-585.
[34] Goldbarg, E., Goldbarg, M., de Souza, G.: *Particle Swarm Optimization Algorithm for the Traveling Salesman Problem*. InTech, Croatia, 2008.
[35] Blum, C.: Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews*, **2** (2005), pp. 353–373.
[36] Dorigo, M., Gambardella, L. M., Birattari, M., Martinoli, A., Poli, R., Stűtzle, T.: *Ant Colony Optimization and Swarm Intelligence*, Springer-Verlag, Berlin, 2006.
[37] Israelashvili, J.: *Intermolecular and Surface Forces*, 2nd Edition, Academic Press, London, 1998.
[38] Satoh, A.: *Introduction to Molecular-Microsimulation of Colloidal Dispersions*. Elsevier Science, Amsterdam, 2003.
[39] Berglund, N., Gentz, B.: *Noise-Induced Phenomena in Slow-Fast Dynamical Systems. A Sample-Paths Approach*. Springer Verlag, London, 2006.
[40] Palacci, J., Cottin-Bizonne, C., Ybert, C., Bocquet, L.: Osmotic traps for colloids and macromolecules based on logarithmic sensing in salt taxis. *Soft Matter*, **8** (2012), pp. 980-994.
[41] Sengupta, A., Kruppa, T., Löwen, H.: Chemotactic predator-prey dynamics. *Physical Review E*, **83** (2011), pp. 031914.
[42] Griebel, M., Knapek, S. and Zumbusch, G.: *Numerical Simulation in Molecular Dynamics*. *Numerics, Algorithms, Parallelization, Applications*. Springer-Verlag, Berlin-Heidelberg, 2007.
[43] Štěpánek, F.: Computer-aided product design. Granulate dissolution. *Chemical Engineering Research and Design*, **82** (2004), pp. 1458-1466.
[44] Snook, I.: *The Langevin and Generalised Langevin Approach to the Dynamics of Atomic, Polymeric and Colloidal Systems*, 1st Edition., Elsevier, Amsterdam, 2007.
[45] Coffey, W., Kalmykov, Y., Waldron, J.: *The Langevin Equation. With Applications to Stochastic Problems in Physics, Chemistry and Electrical Engineering*. 2nd Edition, World Scientific, Singapore, 2004.
[46] Batchelor, G.: *Introduction to Fluid Dynamics.* Cambridge University Press, Cambridge, 1967.

Warren McCulloch & Walter Pitts – Foundations of logical calculus, neural networks and automata

Vladimír KVASNIČKA and Jiří POSPÍCHAL[1]

Abstract. In 1943 was published a paper of Warren McCulloch & Walter Pitts entitled „*A logical calculus of the ideas immanent to nervous activity*“, which is now considered as one of the seminal papers that initiated the formation of artificial intelligence and cognitive science. In this paper, concepts of logical (threshold) neurons and neural networks were introduced. There was proved that an arbitrary Boolean function may be represented by a feedforward (acyclic) neural network composed of threshold neurons, i. e. this type of neural network is a universal approximator in the domain of Boolean functions. Later, S. Kleene and N. Minsky extended this theory by a study of relationships between neural networks and finite state machines (Mealy automata). They proved two important theorems. The first one claims that for an arbitrary neural network (composed of logical neurons) there exists an equivalent finite state machine. In a similar way, the second theorem claims that for an arbitrary finite state machine there exists an equivalent recurrent neural network. From these important properties it immediately follows that symbolic and subsymbolic approaches to the study of cognitive properties of human mind are mutually equivalent.

1 Introduction and basic concepts

Logical neurons and neural networks were initially studied in 1943 by Warren McCulloch and Walter Pitts´s paper [6] „*A logical calculus of the ideas immanent to nervous activity*", which is considered as a milestone of connectionist metaphor in artificial intelligence and cognitive science. This paper demonstrated that neural networks are universal approximators for a domain of Boolean functions, i. e. an arbitrary Boolean function can be

[1] Faculty of informatics and information technologies, Slovak Technical University in Bratislava, Ilkovičova 2, 812 19 Bratislava, E-mail: vladimir.kvasnicka@stuba.sk, jiri.pospichal@stuba.sk.

represented by a feedforward neural network composed of threshold neurons. But, we have to mention from the very beginning that this work is very difficult to read, its mathematical-logical part was probably written by Walter Pitts, who was in both sciences total autodidact. Thanks to logician S. Kleene [2] and computer scientist M. Minsky [7,8] this work has been "translated" at the end of fifties into a form using standard language of contemporary logic and mathematics and its important ideas became generally available and accepted.

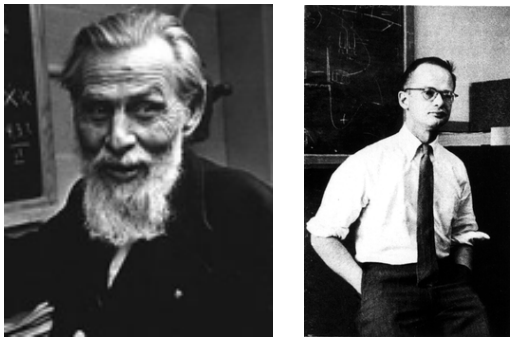


Figure 1. Warren McCulloch (1889 - 1969) and Walter Pitts (1923 - 1969)

An elementary unit of neural networks is ***threshold (logical) neuron*** of McCulloch and Pitts. It has two binary values (i. e. either state 1 or state 0). It may be interpreted as a simple electrical device - relay. Let us postulate that a dendritic system of threshold neuron is composed of excitation inputs (described by binary variables $x_1, x_2, ..., x_n$, which amplify an output response) and inhibition inputs (described by binary variables $x_{n+1}, x_{n+2}, ..., x_m$, which are weakening an output response), see fig. 2.

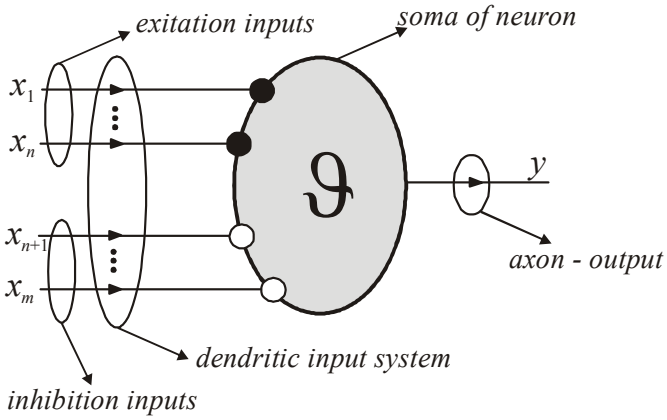


Figure 2. Diagrammatic visualization of McCulloch and Pitts neuron, which is composed of dendritic system for information input (excitation or inhibition) activities, and axon for

information output. A body of neuron is called the soma, it is specified by a threshold coefficient ϑ.

An activity of threshold neuron is set to one, if the difference between a sum of excitation input activities and a sum of inhibition activities is greater than or equal to the threshold coefficient ϑ, otherwise it is set to zero

$$y = \begin{cases} 1 & \left(x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m \geq \vartheta\right) \\ 0 & \left(x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m < \vartheta\right) \end{cases} \tag{1}$$

If we introduce a simple step function

$$s(\xi) = \begin{cases} 1 & (\xi \geq 0) \\ 0 & (\xi < 0) \end{cases} \tag{2a}$$

then an output activity may be expressed as follows:

$$y = s\left(\underbrace{x_1 + \ldots + x_n - x_{1+n} - \ldots - x_m}_{\xi} - \vartheta \right) \tag{2b}$$

An entity ξ is called the internal potential. This relation (2) may be alternatively interpreted such that excitation activities are incoming to the neuron through connections evaluated by positive unit weight coefficients (w = 1), whereas inhibition activities are incoming through connections evaluated by negative unit weight coefficients (w = -1). Then an activity of neuron may be expressed by a simple formula

$$y = s\left(\underbrace{w_1 x_1 + \ldots + w_m x_m}_{\xi} - \vartheta \right) = s\left(\sum_{i=1}^{m} w_i x_i - \vartheta \right) \tag{3}$$

where weight coefficients are specified by

$$w_{ij} = \begin{cases} 1 & \left(connection\ j \to i\ is\ of\ excitation\ character\right) \\ -1 & \left(connection\ j \to i\ is\ of\ inhibition\ character\right) \\ 0 & \left(connection\ j \to i\ is\ nonexisting\right) \end{cases} \tag{4}$$

In a neural network, weight coefficients are fixed and they are determined by a topology of syntactic tree, which specifies a given Boolean function.

Let us note that the above mentioned simple principles (1-4) “all or none” for neurons have originated in late twenties and early thirties of former century by English physician and electro-physiologist Sir E. Adrian, when he studied output neural activities by making use, in that time, of very modern electronic equipment based on electron-tube amplifiers and cathode-ray tubes for a visualization of measurements.

In the original paper [6] McCulloch and Pitts have discussed a possibility that inhibition is absolute, i. e. any active inhibitory connection forces the neuron into the inactive state (with zero output state). The paper itself shows that this form of inhibition is not necessary, and that „subtractive inhibition“ based on formulae (1-4) gives the same results.

Simple implementations of elementary Boolean functions of disjunctions, conjunctions, implication, and negation are presented in fig. 3. Let us study a function of disjunction for $n = 2$, if we use formulae (1-2) we get

$$y_{OR}(x_1, x_2) = s(x_1 + x_2 - 1) \tag{5}$$

Functional values of this Boolean function are specified in tab. 1. It immediately follows from this table that a function y_{OR} simulates Boolean function of disjunction

Table 1. Disjunctive Boolean function

#	x_1	x_2	$y_{OR}(x_1,x_2)$	$x_1 \vee x_2$
1	0	0	$s(-1)$	0
2	0	1	$s(0)$	1
3	1	0	$s(0)$	1
4	1	1	$s(1)$	1

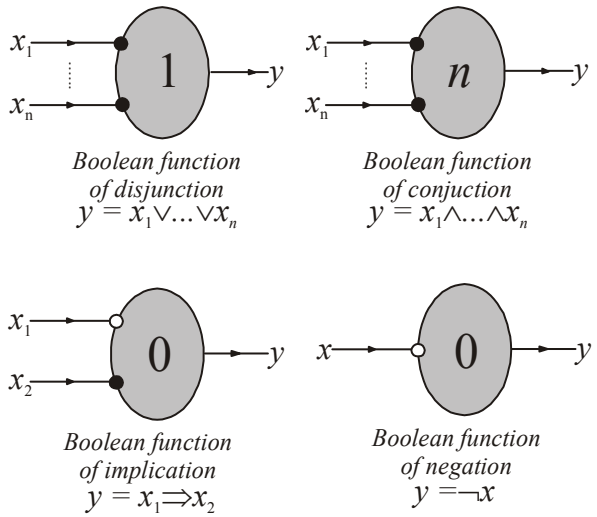


Figure 3. Three different implementations of threshold neurons, which specify Boolean functions of disjunction, conjunction, implication, and negation, respectively. Excitatory connections are terminated by black dot whereas inhibition connections by open dots.

2 Boolean functions

Each Boolean function [5,8] is represented by a syntactic tree (derivation tree), which represents a way of its recurrent building, going bottom up, initiated by Boolean variables and then terminated (at a root of tree) by a composed Boolean function (formula of propositional logic), see fig. 4, diagram A. Syntactic tree is a very important notion for a construction of its subformulae, each vertex of tree specifies a subformula of the given formula: lowest placed vertices are assigned to trivial subformulae p and q, forthcoming two vertices are assigned subformulae $p \Rightarrow q$ and $p \wedge q$, highest placed vertex – root of the tree – is represented by the given formula $(p \Rightarrow q) \Rightarrow (p \wedge q)$.

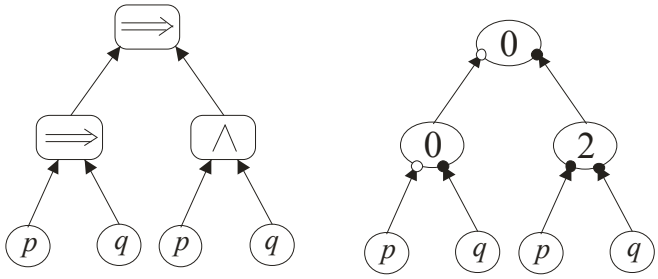


Figure 4. (A) Syntactic tree of a Boolean function (propositional formula) $(p \Rightarrow q) \Rightarrow (p \wedge q)$. Bottom vertices correspond to Boolean variables (propositional variable) p and q, vertices from the next levels are assigned to connectives implication and conjunction, respectively. An evaluation of the syntactic tree runs bottom up. (B) Neural network composed of logical neurons of connectives, which appear in a given vertex of the syntactic tree of diagram A. We see that between syntactic tree and neural network these exists very closed one-to-one correspondence, their topologies are identical, they are different only in vertices. Pictorially speaking, we may say that a neural network representing a Boolean function φ can be constructed from its syntactic tree by direct substitution of its vertices by proper logical neurons.

We see that for an arbitrary Boolean function we may simply construct a neural network, which simulates functional value of the Boolean function, see fig. 4, where this process is outlined for formula $(p \Rightarrow q) \Rightarrow (p \wedge q)$. It means that these results may be summarized in a form of a theorem.

Theorem 1. Each Boolean function, represented by a syntactic tree, can be alternatively expressed in a form of neural network composed of logical neurons that correspond to connectives from the given formula.

This theorem belongs to basic results of the seminal paper of McCulloch and Pitts [6]. It claims that an arbitrary Boolean function represented by a syntactic tree, may be expressed in a form of neural network composed of simple logical neurons that are assigned to logical connectives from the tree. It means that neural networks with logical neurons are endowed by an interesting property that these networks have a property of universal approximator in a domain of Boolean functions. The above outlined constructive approach based on an existence of syntactic tree for each Boolean function is capable of accurate simulation of any given Boolean function.

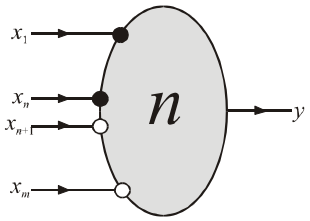


Figure 5. A logic neuron for simulation of an arbitrary conjunctive clause, which is composed of propositional variables or their negations that are mutually connected by conjunctions, $y = x_1 \wedge ... \wedge x_n \wedge \neg x_{n+1} \wedge ... \wedge \neg x_m$.

Architecture of neural network based on the syntactic tree, which is assigned to an arbitrary Boolean function, may be substantially simplified to the so-called 3-layer neural network composed of

(1) a layer of input neurons (which copy input activities, they are not computational units),

(2) a layer of hidden neurons, and

(3) a layer of output neurons;

where neurons from two juxtaposed layers are connected by all possible ways by connections. This architecture is a minimalistic and could not be further simplified. We demonstrate a constructive way how to construct such a neural network for an arbitrary Boolean function.

Applying simple generalization of the concept of logical neuron, we may immediately show that a single logical neuron is capable of simulating a conjunctive clause $x_1 \wedge ... \wedge x_n \wedge \neg x_{n+1} \wedge ... \wedge \neg x_m$, see fig. 5. This Boolean function is true only for variables satisfying $x_1 = ... = x_n = 1$ and $x_{n+1} = ... = x_m = 0$, for all other cases of variables its truth value is 0 (false)

$$val_\tau\left(x_1 \wedge ... \wedge x_n \wedge \neg x_{n+1} \wedge ... \wedge \neg x_m\right) = \begin{cases} 1 & \left(pre\ \tau = \tau_0\right) \\ 0 & \left(pre\ \tau \neq \tau_0\right) \end{cases} \tag{6}$$

where $\tau_0 = (x_1/1,...,x_n/1,x_{n+1}/0,...,x_m/0)$ is a specification of truth values of variables. It can be easily verified that this conjunctive clause is simulated by logical neuron illustrated in fig. 5, its output activity is determined by simple formula

$$y = s(x_1 + ... + x_n - x_{n+1} - ... - x_m - n) \tag{7}$$

Its functional value is equal to 1 if and only if

$$x_1 + ... + x_n - x_{n+1} - ... - x_m \geq n \tag{8}$$

This simple condition is achieved if the first *n* input (excitation) variables are equal to 1 and further (*m*-*n*) input (inhibition) variables are equal to 0.

Table 2. Functional values of a Boolean function.

#	x_1	x_2	x_3	$y = f(x_1, x_2, x_3)$	*clause*
1	0	0	0	0	-
2	0	0	1	0	-
3	0	1	0	1	$\neg x_1 \wedge x_2 \wedge \neg x_3$
4	0	1	1	1	$\neg x_1 \wedge x_2 \wedge x_3$
5	1	0	0	0	-
6	1	0	1	1	$x_1 \wedge \neg x_2 \wedge x_3$
7	1	1	0	0	-
8	1	1	1	0	-

In the theory of Boolean functions is proved very important theorem that each Boolean function may be equivalently written in a form of disjunctive normal form [5,8]

$$\varphi = \bigvee_{\substack{\tau \\ (val_\tau(\varphi)=1)}} x_1^{(\tau)} \wedge x_2^{(\tau)} \wedge ... \wedge x_n^{(\tau)} \tag{9}$$

where

$$x_i^{(\tau)} = \begin{cases} x_i & (\text{if } val_\tau(x_i) = 1) \\ \neg x_i & (\text{if } val_\tau(x_i) = 0) \end{cases} \tag{10}$$

In order to illustrate this theorem let us study a Boolean function with functional values specified in tab. 2, where in its rows 3, 4 and 6 are "one" (true) values and in all other rows the function is false. Applying formula (9) we get an „analytic“ form of the given Boolean function specified initially by tab. 2

$$y = f(x_1, x_2, x_3) = (\overline{x}_1 \wedge x_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge x_2 \wedge x_3) \vee (x_1 \wedge \overline{x}_2 \wedge x_3) \tag{11}$$

This Boolean function may be further simplified in such a way that the first and second clauses are simplified

$$\left(\overline{x}_1 \wedge x_2 \wedge \overline{x}_3\right) \vee \left(\overline{x}_1 \wedge x_2 \wedge x_3\right) = \overline{x}_1 \wedge x_2 \wedge \underbrace{\left(\overline{x}_3 \vee x_3\right)}_{1} = \overline{x}_1 \wedge x_2 \tag{12}$$

Then a final "analytic" form of the studied Boolean function is

$$y = f\left(x_1, x_2, x_3\right) = \left(\overline{x}_1 \wedge x_2\right) \vee \left(x_1 \wedge \overline{x}_2 \wedge x_3\right) \tag{13}$$

Summarizing our considerations, a clause $x_1^{(\tau)} \wedge x_2^{(\tau)} \wedge \ldots \wedge x_n^{(\tau)}$ may be expressed by single logical neuron, see fig. 5. Outputs from these neurons are mutually connected by a neuron, which represents a disjunction (see fig. 3). A final form of the Boolean function (11) is outlined in fig. 6. Results of this illustrative example may be summarized in a form of the following theorem.

Theorem 2. An arbitrary Boolean function *f can be simulated by a 3-layer neural network.*

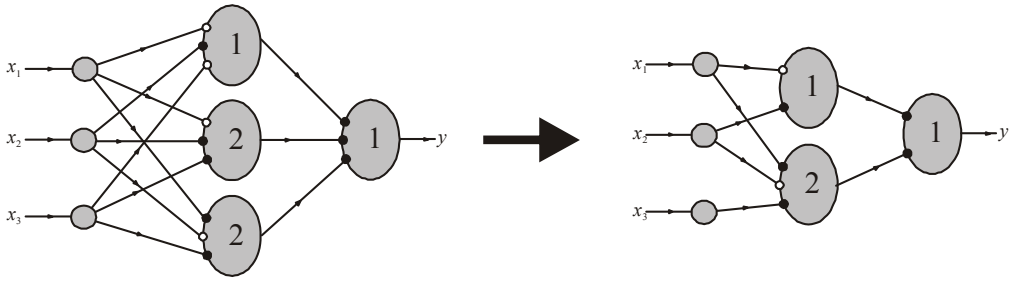


Figure 6. The 3-layer neural network, which simulates Boolean function specified by tab. 2, hidden neurons represent single conjunctive clauses specified in tab. 2, their disjunction is realized by single output "disjunctive" neuron. This neural network may be further simplified in such a way that the first two clauses are combined into a simpler conjunctive clause, see (12-13).

A general form of the 3-layer neural network is illustrated by fig. 7.

We have to note, that according to the theorem 2, the 3-layer neural networks composed of logical neurons are a universal computational device for a domain of Boolean function; each Boolean function may be represented by this "neural device" called the neural network. This fundamental result of McCulloch and Pitts' paper [6] preceded modern result from the turn of the eighties of last century, after which 3-layer feed-forward neural networks with a continuous activation function are a universal approximator of continuous functions specified by a table of functional values [3,12,13]. Moreover, since the proof of theorem 2 was realized in a constructive manner, we know a simple systematic approach how to construct this neural network for an arbitrary Boolean function. Unfortunately, an optimal form of the constructed neural network is not solved by the theorem 2.

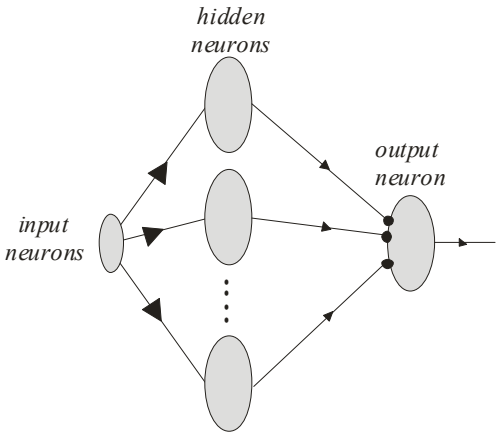


Figure 7. A schematic outline of 3-layer neural network. Going from the left to right, first comes an input layer, which is not a calculating device. The second layer is composed of hidden neurons, which represent single conjunctive clauses of the given Boolean function. The third (last) layer is composed of single output neuron, which performs an addition (disjunction) of activities produced by hidden neurons.

In general, there may exist a neural network composed of smaller number of hidden neurons than the one constructed in the systematic manner from the proof of theorem 2. In the theory of Boolean function, many optimization methods have been elaborated to achieve a "minimal" form of the given Boolean function (e. g. Quin and McCluskey's method [4]). If such an optimization technique is applied in our considerations how to construct a neural network for an arbitrary Boolean function, we arrive at an interesting constructive method that produces neural network composed of minimal number of logical neurons.

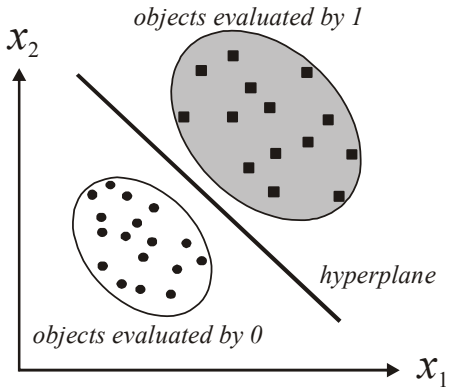


Figure 8. An illustrative outline of the concept "linear separability", where round (square) objects are separated by a hyperplane $w_1x_1+...+w_nx_n-\vartheta = 0$ such that in the first half-space there are situated objects of one kind, whereas in the second half-space there are situated objects of another kind.

We may put a question what kind of Boolean functions a single logical neuron is capable to classify correctly [7,3]? This question may be relatively quickly solved by geometric interpretation of computations running in logical neuron. In fact, logical neuron divides an input spaces onto two halfspaces by a hyperplane $w_1x_1 + w_2x_2 + ... + w_nx_n = \vartheta$, for weight coefficients $w_i = 0, \pm 1$. Then we say that a Boolean function $f(x_1, x_2, ..., x_n)$ is *linearly separable*, if and only if there exists such a hyperplane $w_1x_1 + w_2x_2 + ... + w_nx_n = \vartheta$, which separates a space of input activities in such a way that in the first part of space are situated objects evaluated by 0, whereas in the second part of space are situated objects evaluated by 1 (see fig. 8).

Theorem 3. Logical neurons are capable to *simulate* correctly *only those Boolean functions that are linearly separable.*

A classical example of a Boolean function, which is not linearly separable is a logical connective "exclusive disjunction", which may be formally specified as a negation of a connective of equivalence, $(x \oplus y) \Leftrightarrow \neg(x \equiv y)$, in computer-science literature this connective is usually called the XOR Boolean function, $\varphi_{XOR}(x, y) = x \oplus y$, its functional values are specified in tab. 3.

Table 3. XOR Boolean function

#	x	y	$\varphi_{XOR}(x,y)$
1	0	0	0
2	0	1	1
3	1	0	1
4	1	1	0

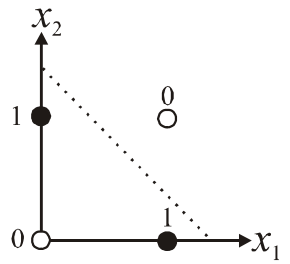


Figure 9. A diagrammatic outline of XOR Boolean function in a state space of its arguments, where objects represented by open (filled) circles are evaluated by 0 (1) . We see from the figure that there could not exist a straight-line (a hyperplane), which divides the whole plane into two sub-planes such that each sub-plane contains two object of the same kind.

If we introduce its functional values into a state space x - y we get a diagram displayed in fig. 9, which is evidently linearly inseparable.

Applying a technique from the first part of this chapter, we may construct a neural network, which simulates this inseparable Boolean function. From its functional values presented in tab. 3 we may directly construct its an equivalent form composed of two clauses

$$\varphi_{XOR}(x_1, x_2) = (\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2) \tag{14}$$

Then this Boolean function is simulated by the following neural network displayed in fig. 10.

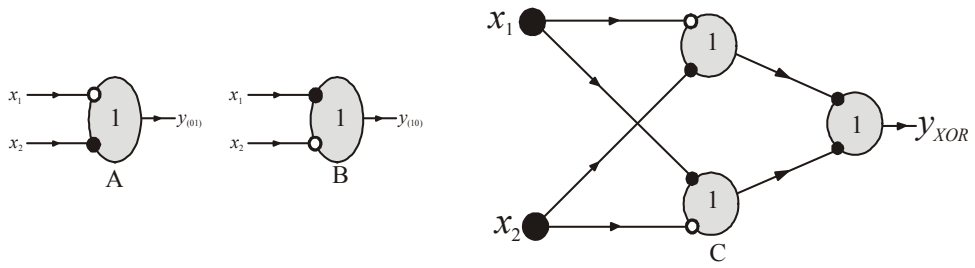


Figure 10. Diagrams A and B simulate single conjunctive clauses from (14). Diagram C represents 3-layer neural network, which hidden neurons are taken from diagrams A and B, respectively. An output neuron corresponds to a disjunctive connective.

Example 1. Construct a neural network, which simulates an addition of two binary numbers:

$$\begin{array}{c} \alpha_1 \\ \alpha_2 \\ \hline \beta_1\beta_2 \end{array}$$

Single output binary variables are specified by $\beta_2 = \alpha_1 \oplus \alpha_2$ and $\beta_1 = \alpha_1 \wedge \alpha_2$. If we use (14), then the second output variable may be written in a form $\beta_2 = (\neg\alpha_1 \wedge \alpha_2) \vee (\alpha_1 \wedge \neg\alpha_2)$, the corresponding network is displayed in fig. 11.

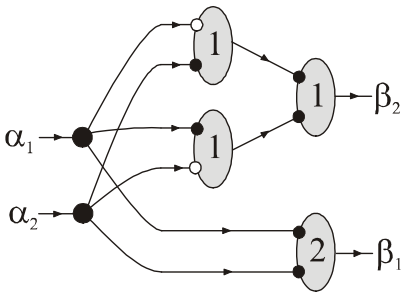


Figure 11. A neural network, which performs an addition of two one-bit variables.

In the previous part of this Chapter there was demonstrated that a single logical neuron is capable to emulate only those Boolean functions that are linearly separable. This severe restriction may be removed if we introduce the higher-order logical neurons [7], which output activity is specified by a generalization of (3) using terms of higher orders

$$y = s\left(\underbrace{\sum_{i=1}^{n} w_i x_i + \sum_{\substack{i,j=1 \\ (i<j)}}^{n} w_{ij} x_i x_j + ... + \vartheta}_{\xi} \right) \tag{15}$$

If an internal potential ξ is determined only as a linear combination of input activities (i. e. only by the first summation term), then the logical neuron is a standard one and it is called "the first order logical neuron". After Minsky and Papert [7], this property of the higher-order neurons may be summarized as a theorem.

Theorem 4. An arbitrary Boolean function *f* is simulated by a logical neuron of properly high order.

This theorem claims that each Boolean function may be simulated by a single logical neuron of sufficiently high order; there exist such weight coefficients and a threshold that for each specification of input variables $x_1, x_2, ..., x_n$, the calculated output activity is equal to a required value.

Example 2. Let us study, as an illustrative example, the Boolean function XOR, which is not linearly separable. Its functional values are presented in tab. 3. Let activity of a logical neuron be determined by a quadratic potential (i. e. we study a logical neuron of the second order)

$$y = s\left(\underbrace{w_1 x_1 + w_2 x_2 + w_{12} x_1 x_2}_{\xi} - \vartheta \right) \tag{16}$$

For XOR we obtain from single rows in tab. 3 these inequalities

$$\begin{aligned} -\vartheta &< 0 \\ w_2 - \vartheta &\geq 0 \\ w_1 - \vartheta &\geq 0 \\ w_1 + w_2 + w_{12} - \vartheta &< 0 \end{aligned} \tag{17}$$

If we solve successive this system of inequalities, we arrive at a solution

$$\vartheta = 1,\ w_1 = w_2 = 1,\ w_{12} = -2 \tag{18}$$

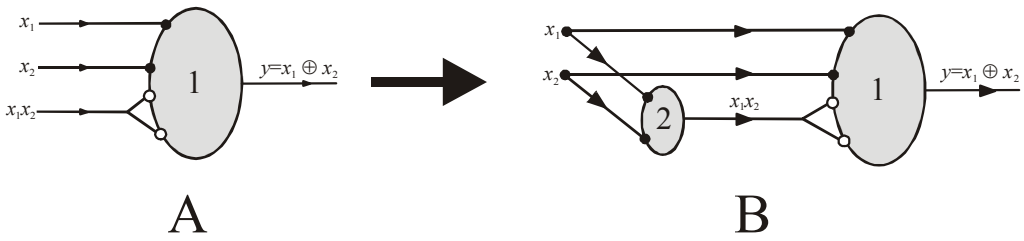


Figure 12. (A) A diagrammatic outline of the second-order logical neuron, which simulates Boolean function *XOR*, where excitation input variables are specified by variables x_1 and x_2, an inhibition activity is assigned to a product x_1x_2. An output activity z is specified by a step function $z = s(x_1 + x_2 - 2x_1x_2 - 1)$. By direct verification for different values of input activities we will see that this single second-order logical neuron simulates the *XOR* function. A fork of inhibitive input means that this input activity is taken into account twice. (B) A transformation of logical neuron of the second order, which simulates the connective *XOR* (diagram A), onto a neural network composed entirely of neurons of the first order. This transformation is based on a construction of product x_1x_2 by making use of single logical neuron (simulating a connection of conjunction), an output from this neuron is used as doubled inhibition input for the output neuron. Thus derived architecture is probably the simplest possible which may be constructed from simple (first order) logical neurons (cf. fig. 9).

In the example 2 we have shown that linearly inseparable function *XOR* may be implemented by making use of a logical neurons with three inputs x_1, x_2, and x_1x_2. In this connection we have to solve an additional problem of calculation of the product x_1x_2, which may be simply performed by a logical connective of conjunction, $x_1x_2 = x_1 \wedge x_2$. If these operation will be performed by a logical neuron of conjunction (see fig. 12, diagram B), then we may create the simplest neural network, which is composed of two neurons, where there are used only two input activities x_1 and x_2. It means that a logical neuron of the second order is capable to simulate correctly Boolean function *XOR*, which is linearly inseparable in 2-dimensional phase space x_1-x_2, but it is linearly separable in 3-dimensional phase space x_1-x_2- x_1x_2 , see fig. 13.

A concept of linearly separable Boolean function can be easily generalized to a quadratic (cubic) separability by making use a concept of quadratic (cubic) hypersurface.

Definition 1. A Boolean function *f* is called quadratic separable if and only if there exist such weight coefficients w_i, w_{ij}, and threshold coefficient ϑ that for each specification of variables $x_1, x_2, ..., x_n$ the following inequalities are satisfied

$$y_{req}\left(x_1,x_2,...,x_n\right)=1 \Rightarrow \sum_{i=1}^{n} w_i x_i + \sum_{\substack{i,j=1 \\ (i<j)}}^{n} w_{ij} x_i x_j \geq \vartheta$$
$$y_{req}\left(x_1,x_2,...,x_n\right)=0 \Rightarrow \sum_{i=1}^{n} w_i x_i + \sum_{\substack{i,j=1 \\ (i<j)}}^{n} w_{ij} x_i x_j < \vartheta \qquad (19)$$

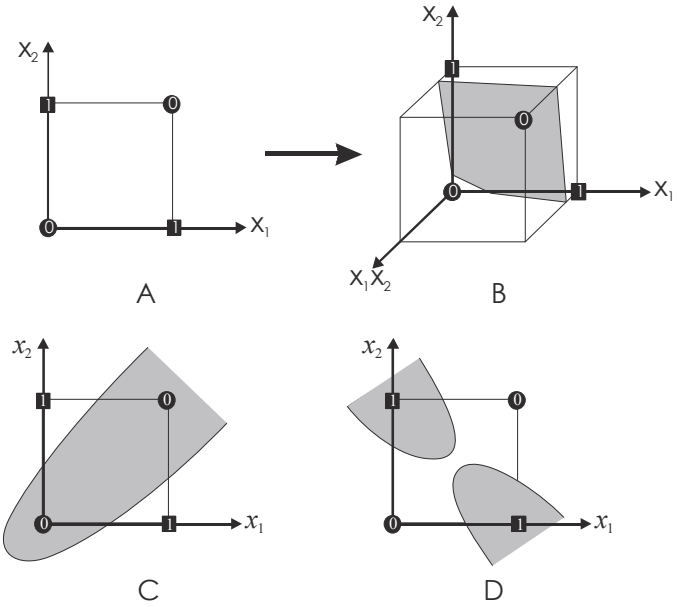


Figure 13. A diagrammatic representation of *XOR* Boolean function. (A) If *XOR* function is represented in 2-dimensional state space x_1-x_2, then objects with unit classification are not linearly separable from objects with zero classification. (B) If *XOR* Boolean function is represented in 3-dimensional phase space x_1-x_2-x_1x_2 , then there exists a hyperplane, which mutually separates objects with different classification. A projection of this hyperplane into a plane x_1-x_2 gives a quadratic curve, which separates objects with different classification, see diagram C and D.

The above outlined approach to a study of separability of Boolean functions can be generalized in a form of a theorem.

Theorem 5. An arbitrary Boolean function *f* can be correctly simulated by a higher-order logical neuron.

This theorem means that for each specification of variables $x_1,x_2,...,x_n$ there exist a higher-order logical neuron (i. e. its weight coefficients and threshold factor), which correctly specifies the given Boolean function for all possible values of its arguments.

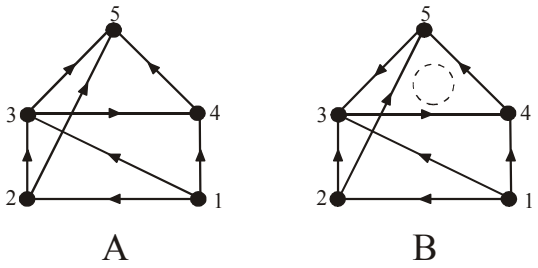


Figure 14. Oriented connected graphs that represent a topology of neural network. The vertex indexed by 1 represents an input neuron, vertices indexed by 2, 3, 4 represent hidden neurons, and finally, the vertex indexed by 5 represents an output neuron. Diagram A is an acyclic graph, whereas diagram B is a cyclic graph (it was created from the l.h.s. graph by reversing orientation of an edge 3-4) .

3 Formal specification of neural networks

From our previous discussion it follows that a concept of neural network belongs to fundamental notions of general theory of neural networks (not only those networks that are composed of logical neurons). Neural network is defined as an ordered triple

$$\mathcal{N} = (G, \boldsymbol{w}, \vartheta) \tag{20}$$

where G is a connected oriented graph, $\boldsymbol{w}$ is a matrix of weight coefficients, and ϑ is a vector of threshold coefficients.

Up to now we did not use time information in an explicit form. We postulate that time t is a discrete entity and is represented by natural integers. Activities of neurons in time t are represented by a vector $\boldsymbol{x}^{(t)}$, in the time $t = 0$ a vector $\boldsymbol{x}^{(0)}$ specifies initial activities of a given neural network. Relation (4) for an activity of the ith neuron in time t is specified by

$$x_i^{(t)} = s\left(\sum_j w_{ij} x_j^{(t-1)} - \vartheta_i \right) \tag{21}$$

where summation runs over all neurons that are predecessors of the ith neuron, activities of these neurons are taken in the time t-1. As an example, let us study a neural network displayed in fig. 14, where the neural network is specified by an acyclic graph, activities of single neurons are determined by (21) as follows:

$$
\begin{aligned}
x_1^{(t)} &= \textit{external input} \\
x_2^{(t)} &= s\left(-x_1^{(t-1)} - 0\right) \\
x_3^{(t)} &= s\left(x_1^{(t-1)} + x_2^{(t-1)} - 2\right) \qquad \left(t = 1,2,...,t_{max}\right) \\
x_4^{(t)} &= s\left(-x_1^{(t-1)} + x_3^{(t-1)} - 1\right) \\
x_5^{(t)} &= s\left(x_2^{(t-1)} + x_3^{(t-1)} + x_4^{(t-1)} - 1\right)
\end{aligned}
\tag{22}
$$

As a side notice, in a consequence of the fact that the neural network is acyclic, in the course of calculation of an activity $x_i^{(t)}$ we need to know activities of the predecessor neurons in the previous time t-1. Neural network $\mathcal{N}$ may be understood as a function, which maps an activity vector $\boldsymbol{x}^{(t-1)}$ in the time t-1 onto an activity vector $\boldsymbol{x}^{(t)}$ in the time t

$$\boldsymbol{x}^{(t)} = F\left(\boldsymbol{x}^{(t-1)}; \mathcal{N}\right) \tag{23}$$

where the function F contains as a parameter the specification $\mathcal{N}$ of the given network.

According to a topology of graphs G from (20), neural networks are divided into two big classes: if graph G is acyclic, then the neural network is called *feedforward*, in the opposite case, if graph G is cyclic, then the network is called recurrent (see fig. 15).

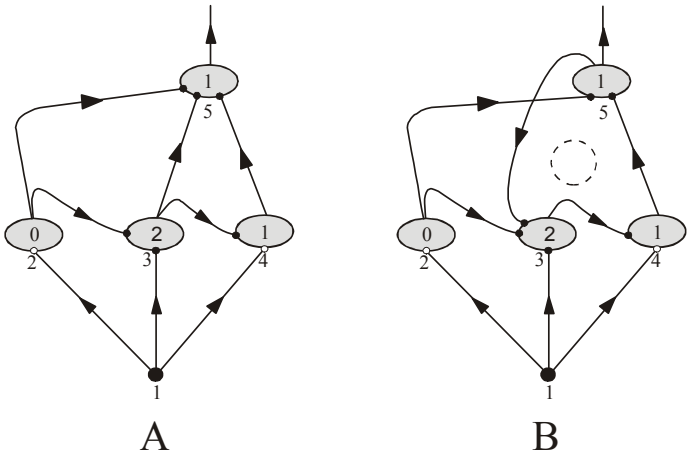


Figure 15. Neural networks that are both specified by oriented graphs outlined in fig. 14. (A) Feedforward neural network specified by the acyclic graph G displayed in fig. 14, diagram A. (B) Recurrent neural network specified by the cyclic graph G displayed in fig. 14, diagram B.

If initial values of activities of neurons indexed by 2-5 for t=1 are zero and input activities are specified by a binary vector of length t_{max}=10 are $\boldsymbol{x}_1$=(1101101010), then activities of hidden and output neurons from networks specified in fig. 14, diagram A, are presented in the following table for times $1 \leq t \leq 10$.

t	x_1	x_2	x_3	x_4	x_5
1	1	0	0	0	0
2	1	0	0	0	0
3	0	1	0	0	0
4	1	1	0	0	1
5	1	0	1	0	1
6	0	0	0	0	1
7	1	1	0	0	0
8	0	0	1	0	1
9	1	1	0	1	1
10	0	0	1	0	1

In general, we may say, that neural network forms a mapping (with parameters specified by graph topology G, weight coefficients $\boldsymbol{w}$, and threshold coefficients ϑ) of a sequence of input activities onto a sequence of output activities

$$(0001110111) = \tilde{F}(1101101010, \textit{parameters of network}) \tag{24}$$

Recurrent neural networks [3,12,13] are specified by a cyclic oriented graph, see diagram B, fig. 14. In this case we may say that this type of recurrent network has a ***memory.*** As a consequence of an existence of closed oriented cycles in recurrent networks, a repeating character of dependency of some activities from other neurons may appear. For instance, in the course of calculation of the activity x_2 in time t, as a consequence of oriented cycles we have to know activities of neurons 1, 2, and 5 in time t-1. Moreover, if we calculate an activity x_5 in a time t-1, then we must know activities neurons indexed by 2 and 4 in time t-2. From this simple discussion it follows that an activity of neuron indexed by 5 in time t is determined by previous activities in times t-1 and t-2. In forthcoming steps the "window to history" may be extended, this fact specific for recurrent networks is called the *„the memory of recurrent networks "*.

For a similar sequence of input activities as was used in the previous illustrative example, $\boldsymbol{x}_1$=(1101101010) and for similar initial activities of other neurons for t=1 (activities of neurons 2-5 in t=1 are zero), by using relations (21) we get

activities of the neural network for different increasing time, which are outlined in the following table.

t	x_1	x_2	x_3	x_4	x_5
1	1	0	0	0	0
2	1	0	0	0	0
3	0	0	0	0	0
4	1	1	0	0	0
5	1	0	1	0	1
6	0	0	1	0	0
7	1	1	0	1	0
8	0	0	1	0	1
9	1	1	0	1	0
10	0	0	1	0	1

Similarly as in previous example of feedforward neural network (see fig. 15, diagram A and eq. (24)), also a recurrent neural network (see fig. 15, diagram B) can be interpreted as a mapping of input sequence $\boldsymbol{x}_1$=(1101101010) onto an output sequence $\boldsymbol{x}_5$=(0000100101).

4 Finite state machine (automaton) [2,7,9]

A finite state machine is schematically outlined in fig. 16, this machine works in discrete time events 1, 2,..., t, t+1,... . It contains two tapes of input symbols and output symbols, respectively, where output symbols are s determined by input symbols and internal states s of the machine (see fig. 16)

$$state_{t+1} = f\left(state_t, input\ symbol_t\right) \tag{25a}$$

$$output\ symbol_{t+1} = g\left(state_t, input\ symbol_t\right) \tag{25b}$$

where functions f and g specify the given machine and are considered as its basic specification:

(1) Transition function f determines the next state, this is fully specified by an actual state and an input symbol,

(2) Output function g determines an output symbol, this is fully specified by an actual state and an input symbol.

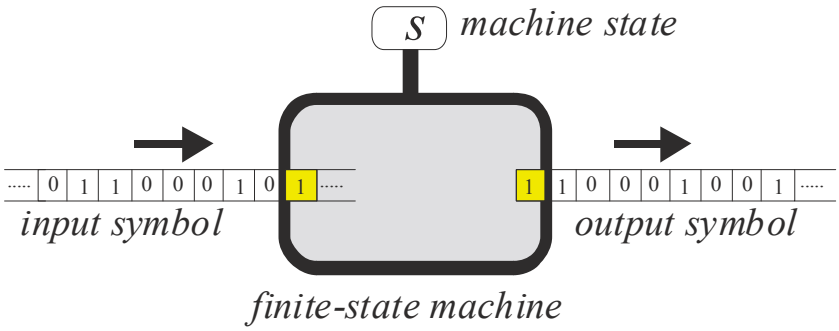


Figure 16. A finite state machine works in discrete time steps 1, 2,...,*t*, *t*+1, ,... . It contains two heads, one for reading of an input symbol and another one for printing of output symbol. In each time step *t* the machine is in specific internal state *s*, in the forthcoming time step *t*+1 an internal state *s* is determined by internal state for a time step *t* and an input symbol also in time step *t* (see relations (26a-b)).

Definition 2. A finite state machine (with an output, called alternatively the Mealy automaton) is defined by an ordered 6-tuple $M=(S,I,O,f,g,s_{ini})$, where $S=\{s_1,...,s_m\}$ is a finite set of internal states, $I=\{i_1,i_2,...,i_n\}$ is a finite state of input symbols, $O=\{o_1,o_2,...,o_p\}$ is a finite set of output symbols, $f: S\times I\to S$ is a transition function, $g: S\times I\to O$ is an output function, and $s_{ini}\in S$ is an initial state.

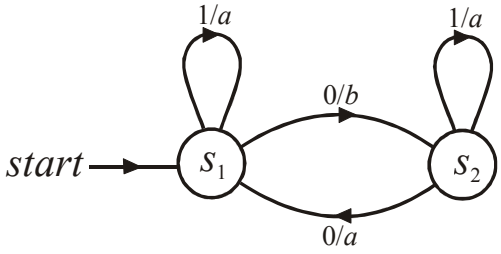


Figure 17. An example of finite state machine, which is composed of two states, $S=\{s_1,s_2\}$, two input symbols, $I=\{0,1\}$, two output symbols, $O=\{a,b\}$, and an initial state s_1. Transition and output functions are specified by tab. 4.

Table 4. Transition and output functions of a finite state machine displayed in fig. 17.

	f		g	
state	*transition function*		*output function*	
	0	1	0	1
s_1	s_2	s_1	b	a
s_2	s_1	s_2	a	a

Transition and output functions may be used for a construction of a model of a finite state machine, see fig. 17.

Sequences of internal states and output symbols for a finite state machine displayed in fig. 16 are determined by tab. 5 for an input sequence of symbols (100111010...). This device may be interpreted as a mapping of input string of symbols onto output string of symbols

$$G\left(\underbrace{100111010...}_{\text{input string } x}; f, g\right) = \underbrace{\square\ abaaaabaa...}_{\text{output string } y}$$

where a symbol □ in an output string means an "empty token", symbols of output string are shifted by one time step with respect to the input string. A mapping G is composed of functions f and g, which specify a „topology" of the finite state machine. For a construction of relationship between neural network and finite state machine we specify this approach as follows: Let $\boldsymbol{i} = i^{(1)}i^{(2)}i^{(3)}...i^{(t)}...$, $\boldsymbol{o} = o^{(2)}o^{(3)}o^{(4)}...o^{(t+1)}...$, and $\boldsymbol{s} = s^{(1)}s^{(2)}s^{(3)}...s^{(t)}...$ be strings of input symbols, output symbols, and internal states, respectively (see tab. 5). Single symbols from these strings are in two mutual relationships (see fig. 18)

$$s^{(t+1)} = f\left(s^{(t)}, i^{(t)}\right) \tag{26a}$$

$$o^{(t+1)} = g\left(s^{(t)}, i^{(t)}\right) \tag{26b}$$

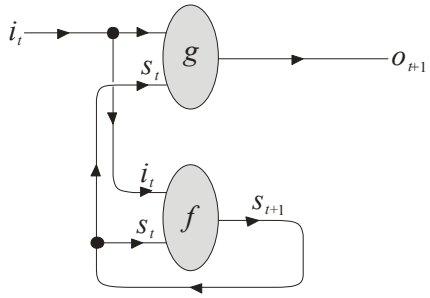


Figure 18. A diagrammatic outline of finite state machine represented by transition and output functions f and g, respectively (see eq. (26a-b)).

The first equation (26a) specifies the next internal state $s^{(t+1)}$ by a transition function f, input symbol $i^{(t)}$, and internal state $s^{(t)}$. In a similar way, the second equation (26b) specifies the new output symbol $o^{(t+1)}$ by an output function g, a previous internal state $s^{(t)}$, and an output symbol $i^{(t)}$. We say that a neural network is *equivalent* to a finite state machine if and only if responses of both devices are identical for the same input. For this equivalence it is not important a way of mapping of input symbols onto output symbols, i. e. a type of calculation accompanying this transformation, a substantial feature here is an equality of output strings for the same input strings for both devices (neural network and finite state machine).

Table 5. Sequences of input symbols, internal states, and output symbols for a finite state machine displayed in fig. 15.

input symbol	1	0	0	1	1	1	0	1	0	..
internal state	s_1	s_1	s_2	s_1	s_1	s_1	s_1	s_2	s_2	..
output symbol	□	*a*	*b*	*a*	*a*	*a*	*a*	*b*	*a*	*a*

Theorem 6 [8]**.** Each neural network can be represented by an equivalent finite state machine with output.

Proof of this theorem is simple and constructive, we show how we can construct for a given neural network single elements from the definition 2, $M=\left(S,I,O,f,g,s_{ini}\right)$. First of all we divide a binary vector of neural-network activities $\boldsymbol{x}$ onto a direct sum $\boldsymbol{x}=\boldsymbol{x}_I\oplus\boldsymbol{x}_H\oplus\boldsymbol{x}_O$, where its components are binary vector of input activities $\boldsymbol{x}_I$, hidden activities $\boldsymbol{x}_H$, and output activities $\boldsymbol{x}_O$, respectively.

(1) The internal-state set S is composed of all possible binary vectors $\boldsymbol{x}_H$, $S=\left\{\boldsymbol{x}_H\right\}$. Let the neural network be composed of n_H hidden neurons, then a cardinality of S is 2^{n_H} .

(2) The set of output symbols is composed of all possible binary vectors $\boldsymbol{x}_I$, $I=\left\{\boldsymbol{x}_I\right\}$, a cardinality of this set is 2^{n_I}, where n_I is number of input neurons.

(3) The set of output symbols is composed of all possible binary vectors $\boldsymbol{x}_O$, $O=\left\{\boldsymbol{x}_O\right\}$, a cardinality of this set is 2^{n_O}, where n_O is number of output neurons.

(4) A function $f : S \times I \to S$ assigns to each couple of internal state and input symbol a new internal state. This function is specified by a mapping (23) produced by the given neural network

$$\boldsymbol{x}_H^{(t+1)} = F\left(\boldsymbol{x}_I^{(t)} \oplus \boldsymbol{x}_H^{(t)}; \mathcal{N}\right) \tag{27}$$

This mapping assigns a new internal state in a time *t* to a couple composed of internal state and input symbol in time *t*-1.

(5) Function $g : S \times I \to O$ assigns a new output symbol to each couple of internal state and input symbol. This function is specified by a mapping

$$\boldsymbol{x}_O^{(t+1)} = \tilde{F}\left(\boldsymbol{x}_I^{(t)} \oplus \boldsymbol{x}_H^{(t)}; \mathcal{N}\right) \tag{28}$$

(6) An initial internal state s_{ini} is usually selected such that all activities of hidden neurons are vanishing (zero).

Summarizing, for a given neural network we unambiguously specify a finite state machine, which is equivalent to the given neural network. This means that any neural network may be represented by an equivalent finite state machine, Q.E.D.

A proof of inverse theorem with respect to theorem 5 (i. e. each finite state machine may be represented by an equivalent neural network) is not a trivial one, the first who proved this inverse form was Minsky in 1967 in his famous book "*Computation: Finite and Infinite Machines*" [7] by making use of very sophisticated constructive approach. Our goal is to construct for a given finite state machine an equivalent neural network.

Theorem 6 [8]**.** Each finite state machine with output (i. e. the Mealy automaton) can be represented by an equivalent recurrent neural network.

Example 3. In this example we present a simple illustrative proof of the above theorem 6. The constructed neural network will correspond to an example of finite state machine with state diagram displayed in fig. 17. This machine is determined for transition and output functions (see tab. 4), which may be expressed as two Boolean function:

(1) Transition function $state_{t+1} = f\left(state_t, input\ symbol_t\right)$:

state,input symbol	*transition function f*
$(s_1,0) \to (0,0)$	$(b) \to (1)$
$(s_1,1) \to (0,1)$	$(a) \to (0)$
$(s_2,0) \to (1,0)$	$(a) \to (0)$
$(s_2,1) \to (1,1)$	$(a) \to (0)$

(2) Output function $output\ symbol_{t+1} = g\left(state_t, output\ symbol_t\right)$:

state, output symbol	*output function g*
$(s_1,0) \rightarrow (0,0)$	$(s_2) \rightarrow (1)$
$(s_1,1) \rightarrow (0,1)$	$(s_1) \rightarrow (0)$
$(s_2,0) \rightarrow (1,0)$	$(s_1) \rightarrow (0)$
$(s_2,1) \rightarrow (1,1)$	$(s_2) \rightarrow (1)$

This means that both functions *f* and *g* are specified as Boolean functions

$$f\left(x_1, x_2\right) = \neg x_1 \wedge \neg x_2$$

$$g\left(x_1, x_2\right) = \left(\neg x_1 \wedge \neg x_2\right) \vee \left(x_1 \wedge x_2\right)$$

A representation of both these functions in a form of neural network composed of logical neurons is displayed in fig. 19.

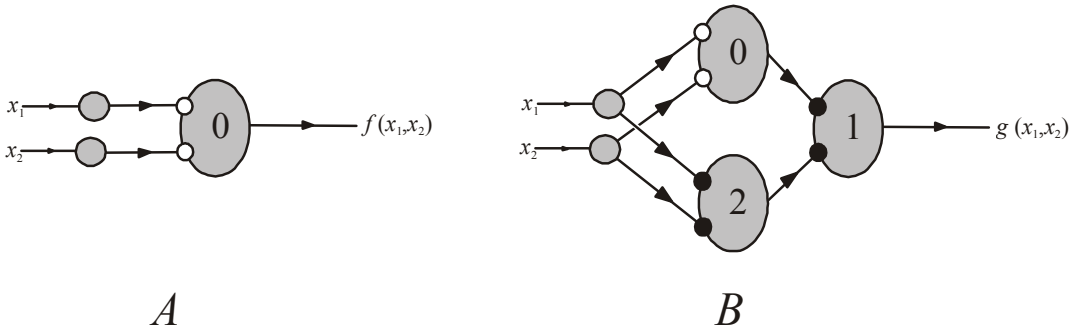


Figure 19. Boolean functions *f* and *g* from example 3.

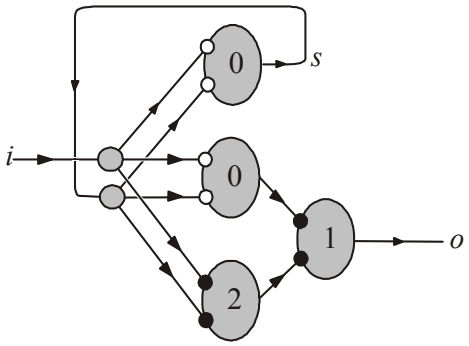


Figure 20. A recurrent neural network, which represents a finite state machine displayed in fig. 17. This network was created by a substitution of Boolean functions *f* and *g* from fig. 19 to diagram displayed in fig. 18.

Let us note that this simple example may serve as a sufficient illustrative specification of a way how to produce a constructive proof of theorem 6, i. e. for any finite state machine (specified by functions f and g) we know a way of construction of an equivalent recurrent neural network. In the first step we construct a neural representation of functions f and g by making use the method outlined in section 2 for construction of Boolean function. In the second step the functions f and g are substituted by their neural representations in general diagram displayed in fig. 18, which specifies finite state machine. This second step may be understood as a finalization of proof of theorem 6, we have demonstrated a constructive method for a construction of neural network equivalent to the given finite state machine.

To summarize our results, we have demonstrated that neural networks composed of logical neurons are powerful calculation device: (1) feedforward neural networks represented by acyclic graph are a universal approximator of Boolean functions and (2) between finite state machine and neural network there exists a property of mutual equivalency. An arbitrary finite state machine may be simulated by a recurrent neural network, and conversely, an arbitrary neural network (feedforward of recurrent) may be simulated by a finite state machine. Both these properties have been proved in constructive way, i. e. we have an algorithm how to construct another device if we know its counterpart. There exists a substantial limitation based on the fact that connection between neurons and their specification as excitatory or inhibitory and also values of threshold coefficients are specified by an architecture of network. In other words, neural networks composed of logical neurons are incapable of learning; a Boolean function (or Boolean functions, if neural network has more than one output neuron) is fully fixed in the course of its counterpart finding process.

5 A view of artificial intelligence and cognitive science on the problem of relationship between mind and brain

In the first part of this section we give a general view of artificial intelligence and cognitive science on the complex mind – brain as a device, which transforms input data $\boldsymbol{x}$ (produced by sight, hearing, smell and so on) onto motor impulses $\boldsymbol{y}$ (whereas this transformation is depending on an internal state $\boldsymbol{s}$ (see fig. 21, diagram A)). The brain may be considered as a huge parallel computer realized by a neural network, which transforms input information $\boldsymbol{x}$ onto output information $\boldsymbol{y}$, where this transformation is affected by internal state (see fig. 21, diagram B). This "neuroscience interpretation" of brain on a microscopic (neural) level does not allow a direct study of higher cognitive activities (solution of problems, understanding of human speech, etc.). We

don't say that it is fundamentally impossible, but it is very clumsy and complicated. For instance, a complexity of this problem is similar to a study of macroscopic problem "surface tension" of water by applying methods of quantum mechanics. Of course, in principle this way of study is possible, but it is very numerically as well as theoretically demanding problem. If we apply here a "phenomenological" approach based on macroscopic thermodynamics, then it is substantially simpler than a pure microscopic approach based on quantum mechanic. In the macroscopic approach we may formulate the problem of "surface tension" very quickly in terms of experimentally measured entities; we get a formula, which is immediately experimentally verified. There exists analogical situation for a study of mind – brain relationship. Neural (connectionist) view is effective only for studies of elementary cognitive activities (e. g. initial transformation of visual information from eye retina). Higher cognitive activities are studied entirely by symbolic or cognitivistic approaches based on an idea that human brain is a computer, which activities are based on the following principles. These principles form a basis of the so-called symbolic paradigm:

(1) It transforms symbols by simple syntactic rules onto other symbols, whereas

(2) sought are symbolic representations implemented by applying a language of thinking, and

(3) mental processes are causal sequences of symbols generated by syntactic rules.

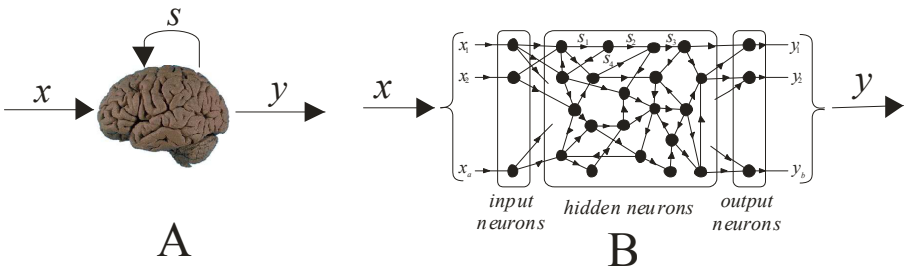


Figure 21. (A) A cybernetic interpretation of brain as a device, which transforms input $\boldsymbol{x}$ onto output $\boldsymbol{y}$, where this transformation is affected by internal state $\boldsymbol{s}$. It means that we may get two different responses $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ on the same input $\boldsymbol{x}$. (B) Connectionist (neural) model of the brain implemented by a neural network, which is composed of (1) input neurons (e. g. perception neurons of eye retina), (2) hidden neurons, which are performing a transformation process of input onto output, and (3) output neurons (e. g. neurons controlling motor activities). Activities of hidden neurons form internal states of neural network, different initial values of their activities cause different responses to the same input activity $\boldsymbol{x}$.

An application of term „computer“ usually evokes an idea of sequential computer with von-neumann architecture (e. g. personal computers are endowed by this architecture), where a strict demarcation line between hardware and software is possible; on the same computer may be performed huge number different programs – software. For these computers, a strict dichotomy exists between hardware and software. Unfortunately, a paradigm of a mind as a computer evokes for many people an idea that there is possible to separate brain from the mind, as two "independent" phenomena, where a brain plays a role of a hardware and the mind a role of software (performed on the hardware - brain).

Let us turn our attention to a modern "neuroscience" approach for an understanding of a relationship between brain and mind [1,10], which is based on the connectionist conception of brain and mind. A basal model of brain (based on experimental neuroscience knowledge) consists in facts that it is formed of neurons that are mutually interconnected by directed (one way) synaptic connections (see fig. 21, diagram B). Thereafter we say that *a capability of brain performing not only cognitive activities but also being a memory should be coded in its architecture*. It means that a computational paradigm of human brain must be formulated in such a way that the brain is a parallel and distributed computer composed of a few milliards (GB) neurons, which are mutually interconnected by one-way connections into extremely complex network. A program in this parallel computer is a built-in function of its architecture, i. e. human brain is a single-purpose parallel computer represented by its neural network, which could not be reprogrammed without changes of its architecture. This "neuroscience" contemplations may be summarized in a general conclusion that the brain and mind form one integral unit, where the mind should be understood as a "program" performed by the brain. The brain and mind are nothing but two different views on the same object brain-mind:

> (1) If we speak about a brain, we thought its "hardware" structure biologically realized by neurons and their synaptic connections (formally represented by a neural network), and conversely,
>
> (2) If we speak about a mind, we thought its cognitive and other activities performed by a neural network (which formally represents the brain).

We say a few remarks on relationship between a distributed representation (called the connectionism or subsymbolism) and a localistic representation (called the symbolism or between cognitivism) in theory of

mind. A dichotomy between these representations is a consequence of their sharp separation. In connectionistic representation there is explicitly considered a lowest level based of neural structures and patterns formed from their neuron activities (going from bottom to up, see fig. 22). A attempt to solve this complicated problem was done by P. Smolensky, he suggested a hierarchic connectionist model where we may move in two opposite direction, from bottom to up and from up to bottom, to harmonize a relationship between connectionism and symbolism. Going from bottom to up in this model, at its end we start to discover patterns with high level of implicitness, which may be interpreted as symbols. Going in an opposite direction, from up to down, at the end, symbolic notions are becoming more explicit, discovered structures with high level of explicitness may be simply interpreted as neural networks of their "subparts"

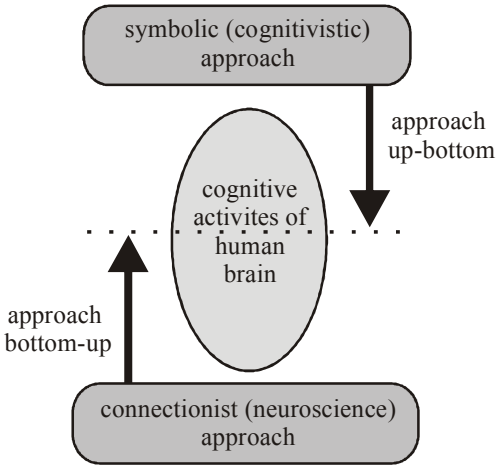


Figure 22. A diagrammatic outline of relationship between connectionism and symbolism in interpretations of cognitive activities of human brain. An interpretative efficiency of this diagram consists in theorems proved by S. Kleene and N. Chomsky [2,8], which state that each neural network is equivalent to a finite state machine, and vice versa. It means that going from bottom to up, we look for symbolic correlates of neural activities. Reversely, going from up to bottom, we look for connectionist correlates for symbolic notions. Henceforth, we may say that between connectionist and symbolic approaches for a study of cognitive activities there doesn't exist an exclusive disjunction. The main criterion of inclusion of the first or second approach consists in an effectiveness and easiness of study of cognitive activities. Recently, there is used a compromise solution that higher level activities are considered on symbolic level (though there exist good connectionist models), whereas low level cognitive activities are considered on connectionist level. For completeness, we mention that D. Gabbay published a seminal book Neural-Symbolic Cognitive Reasoning (Springer, 20008) in which he and his coworkers demonstrated connectionist approaches based on neural networks for a study of logical reasoning.

A realistic interpretation of both these approaches is that they offers different views at the same problem. While the symbolism is appropriate for interpretations of higher-order cognitive activities of human brain, its counterpart is appropriate for low-level cognitive activities (e. g. perception). An alternative interpretation of this view is that symbolism could be understood as an approach bottom-up, which interprets higher cognitive activities by making use of different approaches that are known from artificial intelligence. We have to remember that a suggested model must have connectionistic plausibility; i. e. a substrate of human thinking is brain with entirely connectionist architecture. On the other hand, connectionist approaches to a study and interpretation of cognitive activities of the human brain, are fully based on neural networks and represent up-bottom approach. In the course of application of connectionist methods there is necessary to introduce hypothetical blocks (modules) that perform special activities, which are closely related with block structure of symbolic approaches. In an ideal case, we shall expect that both these approaches are met at halfway denoted by dashed line in fig. 22. For instance, connectionist approaches offer an interpretation of modules used in symbolic approaches. In other words, the connectionism offers for symbolic approaches a "microscopic theory" for its phenomenological notions, which is in accordance with recent concepts of a structure and physiology of human brain.

At the end of this section we say few remarks about a memory, on its symbolic and/or subsymbolic realization. Simplest approach to memory is a symbolic one, the memory is identified with a set of believes or different knowledge items (called the theory in mathematical logic [5])

$$T = \{\varphi_1, \varphi_2, ..., \varphi_n\} \tag{29}$$

where φ_i is a propositional formula. We will postulate that this theory T is consistent, i. e. from this theory may be derived either χ or its negation $\neg\chi$ (but not both simultaneously); in the opposite case we can derive from the theory T any arbitrary formula, which is a nonsense. For such an incorrect theory we may perform a process of revision with a goal to remove possible inconsistencies from the given theory.

Substantially more complex problem is a specification of memory for subsymbolic approaches based on an identification of brain with a neural network. Since a brain is only a neural network, a memory must exist (without it the brain would not properly work as an effective computational device, which on the base of its former and actual experiences predicts a near future), the memories have to be, in some a way, built-in in our brain - neural network. It means that a memory couldn't be a local entity but it is distributed over the neural network. This conclusion is supported by many observation of

neuroscience; for instance by a degenerative illness of brain, which is usually running as „a graceful degradation of memory".

6 Discussion and final notes

McCulloch and Pitts's paper is very ostensibly ''neural'' in the sense that he used an approach for specification of neuron activities based on simple rule all-or-none. However, McCulloch–Pitts neural networks are heavily simplified and idealized when compared to the then known properties of neurons and neural networks. The theory did not offer testable predictions or explanations for observable neural phenomena. It was quite removed from what neurophysiologists could do in their labs. This may be why neuroscientists largely ignored McCulloch and Pitts's theory. For this scientific community, its main power is not consisting in a capability to produce verifiable hypothesis, but it consists in a fact that such extremely simple neural theory offers arguments of basal character for a discussion of "philosophical" problems about a brain and mind relationship. There can not be expected that a further "sophistication" of this theory (e. g. the rule "all-or-none" is substituted by another more realistic rule or "spiking" neurons are used, etc.) will negatively influence general results deduced from the model.

The 1943 paper by McCulloch and Pitts was influential in a large number of domains, some of them unexpected. In the realm of mathematics itself this paper is often given credit for founding of the important field known as finite state automata theory. However, its influence went even further. The paper was published at the height of the Second World War. At that time there were a number of projects in progress to build practical computing machines for various military purposes. The teams involved became aware of the McCulloch–Pitts paper very early on.

One of those influenced was John von Neumann, who is known as a creator of the so-called „von Neumann computer architecture", which was outlined in his famous 1945 technical report. He mentioned that in existing digital computing devices, various mechanical or electrical devices have been used as elements. It is worth mentioning that the neurons are definitely elements in the above sense. It is easily seen that these simplified neuron functions can be imitated by telegraph relays or by vacuum tubes. The proposed similarity between the computer and the architecture of the brain was taken very seriously by computer scientists at the time. When early computer scientists referred to computers as 'giant brains', they were not just using a metaphor, but were referring to what they thought were two computing systems based on the same principles but using different hardware. From the early

1940s the McCulloch–Pitts neuron was considered by many non-neuroscientists to be the most appropriate way to approach neural computation, largely because the work of McCulloch and Pitts was so well known.

Finally, M. Minsky in the early 1970s commented [7] the paper of McCulloch and Pitts as follows: The McCulloch and Pitts paper is not a correct for many biological neuroscientists in its initial domain of application – in this case brain theory, since the used rule "all-or-none" is very rough and simplifying from the standpoint of modern neurophysiology. But it is immensely valuable in many other places and at many different levels, and secondly, that a tight coupling between brain science and computer science has existed from the earliest beginnings of both fields, and has enriched both.

McCulloch and Pitts's views – that neural nets perform computations (in the sense of computability theory) and that neural computations explain mental phenomena – permanently belong to the mainstream theory of brain and mind. It may be time to rethink the extent to which those views are justified in light of current knowledge of neural mechanisms.

Acknowledgment. The authors acknowledge financial support from Slovak grant agency, grants VEGA 1/0553/12 and 1/0458/13.

References

[1] Anderson, J. A., Rosenfeld, E.: *Talking Nets: An Oral History of Neural Networks*. MIT Press, Cambridge, MA, 1998.

[2] Kleene, S. C.: Representation of events in nerve nets and finite automata. In Shannon, C. E., McCarthy, J. (eds.): *Automata Studies. Annals of Mathematics Studies*, Vol 34. Princeton University Press, Princeton, 1956, pp. 3-41.

[3] Kvasnička, V., Beňušková, Ľ., Pospíchal, J., Farkaš, I., Tiňo, P., Kráľ, A.: *Introduction into theory of neural networks* (in Slovak *Úvod do teórie neurónových sietí*.) IRIS, Bratislava, 1997.

[4] Kvasnička, V., Pospíchal, J.: *Algebra and discrete mathematics* (in Slovak *Algebra a diskrétna matematika*). Vydavateľstvo STU, Bratislava, 2008.

[5] Kvasnička, V., Pospíchal, J.: *Mathematical logic* (in Slovak *Matematická logika*). Vydavateľstvo STU, Bratislava, 2006.

[6] McCulloch, W. S., Pitts, W. H.: A Logical Calculus of the Ideas Immanent in nervous Activity. *Bulletin of Mathematical Biophysics* **5**(1943), 115 – 133.

[7] Minsky, M. and Papert, S.: *Perceptrons. An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, 1969.

[8] Minsky, M. L.: *Computation. Finite and Infinite Machines*. Prentice-Hall, Englewood Cliffs, NJ, 1967.

[9] Molnár Ľ., Češka, M., Melichar, B.: *Grammars and Languages* (in Slovak *Gramatiky a jazyky*). Bratislava, Alfa, 1987.

[10] Piccinini, G,: Synthese 141(2004), 175–215.

[11] Randell, B. (ed.): *The Origins of Digital Computers*. Springer, Berlin, 1973.

[12] Rojas, R.: *Neural Networks. A Systematic Introduction*. Springer, Berlin, 1996.

[13] Šíma, J., Neruda, R.: *Theoretical questions in neural networks* (in Czech *Teoretické otázky neurónových sítí*) Matfyzpress, Praha, 1999.

Clinical Decision Support Systems and Reasoning with Petri Nets

Fedor LEHOCKI[1] and Lucia CIBULKOVÁ[1]

Abstract. Modern knowledge representation is a very dynamic domain because of continuous research and development in medical informatics related to knowledge management. Decision support systems (DSS) in medicine provide recommendations to aid decision making of medical personnel. A typical component of DSS is the knowledge base, where the experts' knowledge is encoded. There exist several formalisms for encoding the expert's knowledge. Some of them are based on neural networks, semantic networks, machine learning, Petri nets or on a mixture of these techniques. Which formalism should be chosen depends on the medical domain of DSS application and the experience of the system designer with a particular technique. This chapter introduces the notion of clinical diagnostic systems with respective knowledge representation described by formalisms of logical and fuzzy Petri nets. The knowledge propagation (reasoning) is realized by a novel algorithm including the convergence to the unique stable recommendation of diagnostic system and its delivery in a finite time. Chapter also provides the extensive overview of decision support systems with aim to provide the basic orientation points in the domain and attract the new followers.

1 Introduction

Latest trends of unprecedented ageing population post new challenges, needs and demands of societies on provision of healthcare services. Based on recent UN report [1] by 2045 for the first time, the number of older persons (aged 60 years or over) will exceed the number of children (persons under age 15). In 2009 estimated number of elderly was around 700 million with prediction of

[1] Institute of Computer Science and Mathematics, Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava, Ilkovicova 3, 812 19 Bratislava, e-mail: fedor.lehocki@stuba.sk

[2] http://www.ezdravotnictvo.sk/Program-eHealth/Strategicke-dokumenty/Strategicke-ciele-eHealth/Stranky/default.aspx

rising up to 2 billion persons by 2050. During this period the potential support ratio – the number of persons aged 15 to 64 for each person aged 65 years or over – is expected to decline to 4 potential workers per older person. To address these changes in demography and growing costs of healthcare provision resulting from management of chronic diseases which are mainly related to elderly people, new, innovative paradigms of healthcare services need to be employed. Recent status report [2] informed about establishing globally the national e-strategies including eHealth. Also WHO in 2005 at its 58th World Health Assembly approved an eHealth resolution with a global commitment to all its member states to develop and implement strategic plans for implementing eHealth services in various domains of healthcare sector [3]. On the national level in 2008 government of the Slovak Republic approved the document related to strategic objectives of eHealth programme which officially initiated the development and implementation of electronic health services, e.g. EHR (electronic health record), ePrescription and others[2].

One of the eHealth services with growing potential of implementation to practice is telemedicine [4]. It can be defined as a provision of healthcare services on distance with application of information and communication technologies. When speaking about telemedicine services we address the 'complete loop' consisting of remote data collection, data transmission, expert review and feedback [5] (Fig. 1).

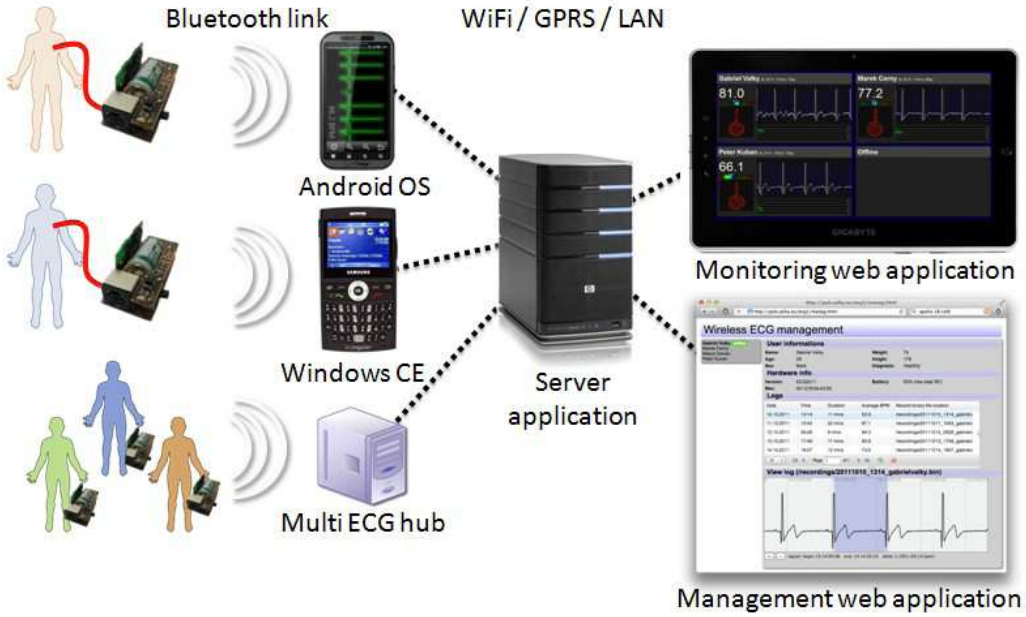


Fig. 1. Example of telemedicine service for ECG monitoring

Remote data collection is realized through involvement of sensors monitoring the health status of the patient, his physiological signals related to ECG, blood glucose, EEG, EMG, weight, SpO_2 etc. Recorded data are pre-processed either on the sensors or transmitted via the wireless link like Bluetooth to "personal-health-server" for further processing (for example to mobile device). This pre-processing enables to incorporate basic logic evaluating the data for alarms generation, notifying the patient about his health status, remind him to take medications etc. It also provides the means to optimize the measured signals for data transmission. To achieve cost effectiveness of the solution pre-processing also determines whether the patient status is critical and medical personnel need to be alerted and data immediately transmitted or the data can be stored and transmitted periodically, in pre defined time slots. The data transmission policy also contributes to intelligent power management of the sensors and personal health server. Various types of sensors (wearable, implantable, ingestible) have different demands for energy, therefore innovative approaches of powering are being researched, like energy scavenging from body heat, blood sugar etc. [6].

Realization of telemedicine services always depends on several factors - the application (e.g. chronic diseases, acute states, adherence to the treatment process etc.), target group (elderly, children, sportsmen) or environment (home monitoring, extreme conditions monitoring – military etc.).

Once the data have been transmitted to medical servers it is stored in EHR databases. These serve as an input data for clinical decision support systems (CDSS or DSS) which provide the complex analyses of the data resulting in generation of recommendations on the patient treatments for medical personnel, prediction of patient health status evolution etc. This "last mile" will be the main topic of this chapter involving the more detailed description of what decision support is, how it is performed, what are the main challenges of successful implementation of DSS in clinical practice (Section 2). Section 3 describes clinical guidelines as a knowledge repository for decision support. Careflows are described in Section 4 defining how the patient is managed over time (e.g. for chronic conditions) and how to incorporate the DSS into clinical workflow. The chapter concludes with description of formalism for knowledge representation and reasoning in DSS based on logical and fuzzy Petri nets for reasoning with uncertainties (Section 5). Final remarks about the topic are introduced in discussion.

2 Clinical Decision Support Systems

Progress in information processing methodology, information and communication technologies (ICT), medicine and healthcare enable to address

successfully the changes in needs, requirements and expectations of patients related to improving the provision of healthcare while maintaining or lowering the costs. Medical informatics is contributing to good medicine and good health for individual by enabling the computer based decision support for healthcare professionals and patients combined with relevant concepts for reasoning and knowledge representation, including comprehensive and accessible medical knowledge bases, controlled medical vocabularies, data mining and analysis for identifying new medical knowledge enabling effective health consulting [7]. Well organized healthcare is also achieved through effective architectures of health information systems and information management methods centered on the patient (not on institution).

Based on the definition in [8] clinical decision support is referred as a provision to patients and clinicians with computer-generated clinical knowledge and patient related information which is intelligently filtered and presented at the proper time, in the right form and at the point of care to enhance the delivery of healthcare. Some of the drivers for application of DSS relate to:

- Increasing amount of available patient-specific clinical information in electronic format resulting in difficulties to process by a clinician and leading to possible errors in patient status assessment and recommendations related to further care. This increases even more with gathering the complex patient genomic data related to the paradigm of personalized medicine.
- Evolution of available medical knowledge (it is estimated that every few years the available medical knowledge doubles).
- Improvement in implementation and adherence of clinicians to the best practices vs. actual practices.
- Need for interdisciplinary patient management (e.g. cancer patients).
- Support in patient care unfolding over time (chronic patients).

Classification of different types of clinical decision support systems [9] [10] complements the above definition with applications of DSS: monitoring, generation of alerts and reminders (techniques used are algorithmic and rule based methods), modeling and prediction (statistical modeling, calculators), information retrieval and focusing, producing better documentation (search engines), decision making (decision analyses and logical decision models), support for complex and multidisciplinary care, care planning, supporting better communication among providers (workflow management techniques).

The model of decision support system is depicted in Fig. 2. Various scenarios can be applied during DSS implementation, the model we introduce might be seen by other researchers differently, with different elements and their organization. The intention here is not to define the general model, we

introduce it for the purposes of the chapter and to provide the reader with easier orientation in the concept we introduce in further sections.

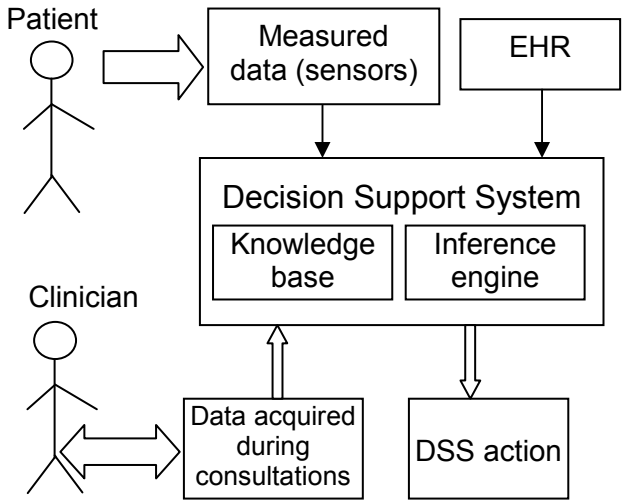


Fig. 2. Model of decision support system

DSS consists of knowledge base and inference engine. Knowledge base represents the repository of expert knowledge while inference engine is designed to derive the actions of DSS based on this knowledge, e.g. recommendations, alerts, reminders, prediction, decisions about further treatments etc. Input data which determines the outcomes of DSS (e.g. types of recommended diagnoses) can be gathered directly from the patient as mentioned in telemedicine scenario (measuring the data like ECG, SpO_2 with sensors). Other sources are patient data repositories in electronic health record (EHR) or information gathered during the consultation with a patient.

In further text we will devote sub sections to the topics of knowledge management tasks (subsection 2.1), knowledge representation formalisms (subsection 2.2), and design considerations for successful DSS implementation, standardization efforts and evaluation of outcomes (subsection 2.3). The exposition will be based on review paper from Peleg et. al. concerning decision support, knowledge representation and management in medicine [11].

2.1 Knowledge management in DSS

From the perspective of several drivers defining the needs for DSS implementation related to management of large amounts of patient data, evolving medical knowledge, management of multidisciplinary cooperation and chronic diseases, knowledge management can be perceived as a *process*

oriented task and as a *knowledge modeling task* to facilitate implementation of DSS into effective healthcare provision.

The former task includes tools and methodologies from business process management (BPM) [12] [13] for understanding of care processes (careflows, CfMS [14]) in which DSS needs to operate. This includes the healthcare organization goals, information flow and work flow, roles and responsibilities of medical personnel and communication and coordination issues of the care processes [11]. Besides the formal approach based on BPM methodology there exists also an informal approach – a workbook [15] to help implementers of DSS to define the stakeholders relevant to healthcare provision, goals and objectives of DSS, defining the system capabilities and finally selecting, deploying and monitoring DSS interventions. There are two different views on connection between DSS and clinical workflow. One view is that in order to be successful DSS need to fit into the existing clinical workflow. The other approach is that implementation of DSS into existing clinical workflow is perceived as the process improvement of that workflow and therefore the emphasis should be put on the change management of organizational development triggered by DSS implementation. We believe that both views can be perceived as a complementary to some point – including the DSS into existing clinical workflows minimises the disruption of clinicians from their daily work to learn new technology or to spend more time with "just another" technical gadget rather then with patients. On the other side even when DSS is part of the clinical workflow the clinicians are aware of its existence and functionality and the communication should be developed towards clinicians to show them benefits of DSS resulting in its acceptance and utilization.

Knowledge modeling task relates to two different types of knowledge. Explicit knowledge is represented in some formalized way accessible to other people (e.g. scientific literature, clinical guidelines). Tacit knowledge is "subjective" knowledge of the expert, his experience and ability to perform particular tasks and cannot be expressed easily. When formalized and made accessible to other people it becomes explicit knowledge. Knowledge modeling involves various stages of tacit and explicit knowledge management including acquiring, representation, sharing, evolution and delivery of knowledge to the user.

Knowledge acquisition from expert is a very demanding and time consuming process. It includes methodologies like interviews, observation of clinicians during their work, questionnaires. Acquisition of new knowledge from literature, EHR and other formalized knowledge repositories can be achieved with various data and knowledge mining tools and machine learning techniques [16]. Machine learning represents very helpful tool for automatic discovery of

knowledge based on learning from examples. The most common technique of machine learning is neural nets enabling knowledge discovery based on classification of examples.

Knowledge representation is devoted towards application of various formalisms (e.g. Petri nets, neural networks, ontologies, first order logic etc.) to achieve the representation of knowledge in a format understandable by humans and machines. When formalized, knowledge can be easily *shared* among healthcare institutions, various DSS (executable knowledge components) and experts. As mentioned before medical knowledge is rapidly *evolving* and doubles every few years. This is motivated by research in medical domains and technology advancement (e.g. gene mapping technologies creating vast amount of knowledge about human genome resulting in the new field of medicine – personalized medicine). From the DSS point of view this means updating knowledge base with new knowledge and ensuring the version management of knowledge base. *Delivery of knowledge to the user* includes recommendations based on the patient data, retrieval of reference information for explanation of recommendations and guidance on how to implement recommended actions.

2.2 Knowledge representation formalisms for DSS

There exist several well known "classical" formalisms for knowledge representation and the most of the DSS representations and reasoning about knowledge are based on them. These includes clinical algorithms, mathematical pathophysiological models, Bayesian statistical systems and influence diagrams, neural networks, fuzzy set theory, symbolic reasoning (expert systems, knowledge bases, rules).

Another wide spread formalism of knowledge representation is ontology. By definition "an ontology is specification of a conceptualization" [17]. This means that ontology is a description of the objects, concepts and other entities ("items") that exist in area of interest (e.g. cardiology domain) and relationships that hold among them [18]. Ontologies are designed for the purpose of knowledge sharing and reuse and are represented in formal language such as frames, description logic or rules. This makes ontologies suitable for decision support and explanation facilities because its representation allows logical inference (by using for example symbolic reasoning) over the set of objects, concepts and their relationships. *Protégé* is a free, open source ontology editor developed at Stanford University with the possibility to export the designed ontologies into variety of formats including RDF(S), OWL and XML Schema (http://protege.stanford.edu).

Ontologies are used to represent clinical practice guidelines in the computable form (computer interpretable guidelines, CIGs). Guidelines contain

the latest or best-practice clinical knowledge based on evidence from literature, studies from randomized clinical trials in specific areas of clinical expertise. They are intended to assist clinician and patient with decisions related to appropriate healthcare under specific clinical circumstances and therefore their goal is to improve the healthcare provision, reduce practice errors and to save costs. More on clinical practice guidelines and its computable form (CIGs) will be introduced in Section 3.

Another relatively wide spread knowledge representation formalism are rules, which are the most suitable for expressing single medical decisions implemented as alerts and reminders. The most common nature of medical decisions is that they are made under uncertainties. Fuzzy rules represent the reasoning which is closer to the human thinking and provide the possibility to operate with linguistic variables (e.g. high temperature). Both deterministic and fuzzy reasoning can be represented as IF…THEN rules in knowledge base using the formalism of logical or fuzzy Petri nets [19] (see Section 5).

2.3 Challenges for effective and successful DSS implementations

In their paper Sittig et. al. [20] describe ten challenges for "high quality, effective means of designing, developing, presenting, implementing, evaluating, and maintaining all types of clinical decision support capabilities for clinicians, patients and consumers". Those challenges were identified based on the experience of the authors and other experts in the domain of DSS implementation. To provide a broader overview of the factors leading to successful implementation of DSS we will introduce the theoretical background of formal model of knowledge-based decision making and clinical processes and distributed care services described in four thematic areas related to decision, process, knowledge and organization theory. This theoretical introduction is based on the work of Fox et. al. [21].

Reasons for definition of framework leading to successful deployment of DSS lie in somewhat slower take-up of DSS applications in clinical practice despite their unquestionable benefits to the provision of quality of healthcare services. Lack of wide spread of DSS can be found in insufficient integration with existing standards for knowledge representation and decision modeling (e.g. PRO*forma* [22]), effectiveness of capturing mechanisms resulting in reliable knowledge, complexity of decision making process resulting in difficulties in delivering more advanced forms of decision support (e.g. specifying decisions in the context of clinical workflows and care plans), social aspects of changes in clinical care, organizational and cultural aspects of DSS deployment [23]. Various initiatives on establishing national eHealth services [3] and examples of large group of DSS suppliers around the world (www.openclinical.org) confirm the potential of this field.

The theoretical and practical framework that will be introduced here represents one of many possible approaches for identification of successful factors for DSS implementation. Its intention is to motivate the reader for further study and research.

Theoretical foundations of DSS design consist of:

- *Decision theory* – includes descriptive and normative approaches. Understanding the nature of human-decision making (cognitive approach) in clinical setting makes the solid base to understand the challenges posed to DSS. Understanding the sources of errors in human judgment can prevent the possible design flaws in DSS. Large research tradition is also in rational reasoning and decision making based on developments in applied mathematics, statistics and computer science. Approach taken in [21] is based on cognitive view of rational decision making and logical inference resulting in derivation of conclusions from data and inference based on defensible norms.
- *Process theory* – includes formal representation and understanding of clinical processes and care plans. Majority of DSS are targeted towards individual points of care in a single point in time where alerts, reminders and decision are provided. It means that DSS developers concentrate on individual tasks rather than on the whole clinical processes which spread over time and are more complex. Research in BPM has concentrated on development of formal notations for modeling and automating the workflows but not on the integration of decision-making into the business processes. Clinical guidelines community had developed computational models that combine decision-making with clinical process modeling based on Task Network Models (TNMs) [24]. Intersection of BPM approach and DSS can be perceived in combining the TNM concept with Petri net formalisms [25] resulting in solid background for clinical workflows analyses.
- *Knowledge theory* – introduces formal representations of knowledge and provides overview of existing formalisms. We have already introduced some of the knowledge representation approaches in Section 2.2. Based on ongoing development in this domain it is not feasible to insist on any particular framework for knowledge representation. One of the promising approaches for medical data and knowledge representation will be semantically rich models – formal ontologies based on first-order representation techniques (description logics).
- *Organization theory* – introduces agents (persons, information systems and other entities involved in clinical process), shared care and understanding of distributed organizations. From the historical point of view management of the patient was held locally either with single

clinician or team of health professionals working in a single workplace (ambulance, clinic etc.). Nowadays clinical practice is more complex resulting in distributed and service oriented provision of healthcare. It means that patient is managed by several teams and in several specialist sites performing the local or distributed tasks. Therefore integration of DSS, workflows and advancement in multi-agent technology for healthcare are key areas for research.

Now that theoretical foundations of successful DSS implementation have been introduced ten challenges in DSS implementation can be introduced divided into three categories [20]:

- improvement of effectiveness of DSS interventions (improvement of human-computer interface, summarization of patient-level information, prioritization and filtering of recommendations to the user, combination of recommendations for patient with co-morbidities, using free text information to drive clinical decision support)
- creation of new DSS interventions (prioritization of DSS content development and implementation, mining of large clinical databases to create new DSS)
- dissemination of existing DSS knowledge and interventions (dissemination of best practices in DSS design, development, and implementation, creation of architecture for sharing executable DSS modules and services, creation of internet-accessible clinical decision support repositories).

Another possible approach defines following factors leading to successful DSS implementation [11]: computerization of decision support rather than paper-based approach, consideration of workflow integration, provision of timely advice, clinical effects and cost of the system should be evaluated, development of DSS with ability to be maintained and extended, evidence should be captured in machine-interpretable knowledge bases, establishment of public policies that provide incentives for implementing DSS, integration with IT environment of healthcare institution, provide ability to clinicians to change the knowledge base, DSS should provide direct recommendations rather than assessment of patient status that need to be further considered by clinician.

Interesting description of features identifying successful DSS implementation was introduced in [26]: automatic decision support as part of a clinical workflow (DSS should not be considered as a separate services and perceived by clinicians as something that need to be done additionally, putting burden to their limited time frames with the patients), requesting documentation for the reasons for not following DSS recommendations (if a clinician doesn't

carry out the recommended action, e.g. vaccine, he is asked to justify the decision with statement like "The patient refused" or "I disagree with recommendation"). This posts additional requirements to clinical workflow systems that is dynamicity of clinical processes. If clinician refuses the prescribed next step by DSS which is part of a process the clinical workflow system should be able to manage this situation and continue with effective patient management (more on the careflows will be introduced in Section 3). Further factors of successful DSS implementation relate to promoting action rather than inaction by DSS (in case that radiograph is not of clinical value, system should provide recommendation for alternative next step rather than just informing clinician that order of radiograph is cancelled), local users should be involved in development of DSS (system design is finalized after testing the prototypes with clinicians of the institution where DSS will be deployed. This ensures that clinicians will perceive the implementation of DSS as a common effort and lowers the risk for the system to be refused). Application of DSS is accompanied by conventional education (in case DSS is employed with aim to reduce the unnecessary ordering of abdominal radiographs, deployment of the system is accompanied by educating the clinicians on appropriate indications for ordering these radiographs). Considering the clinician-system interaction features, implementation of DSS into clinical workflow should save clinicians time and requires minimal effort from the user to work with it. Another interesting approach in ensuring the positive acceptance would be to align decision support system objectives with organization priorities, beliefs and financial interest of clinicians. Organization priorities might relate to patient safety and costs containment, belief of clinicians relate to their agreement with DSS recommendations (this is not the issue of validity or accuracy of action proposed by DSS, we assume that system is reliable; it relates for example to the agreement with the therapy of increasing use of beta blockers for patient with congestive heart failure). Regarding financial interest of clinicians question could be raised about existence of financial incentives to follow or reject advices proposed by DSS.

As mentioned above one of the factors for successful implementation of DSS is integration with existing information systems of healthcare institution (EHR, order entry systems). Therefore standardization in information system infrastructure relating to standard terminology, data model, data exchange format, and other clinical information services is of great importance. The most relevant standardization organizations in the field are Health level 7 (HL7) and European Committee for Standardization (CEN).

Mission of HL7 is to provide framework and related standards in medical informatics related to exchange, integration, sharing, and retrieval of electronic health information. It has special work group devoted to clinical

decision support which creates and promotes standards related to single-patient-focused health care decision support formalization. The group is also responsible for support and development of Arden Syntax for Medical Logic Systems as well as a standard for representation of clinical guidelines (www.hl7.org).

CEN is the major provider of European Standards (ENs) and technical specifications. Domain of medical informatics is covered by CENs Technical Committee 251 which covers the topics related to communications, terminology, security, safety and quality, and technology for devices interoperability (www.cen.eu).

Motivation for evaluation of decision support system is based on high cost of implementation and delivery of reliable recommendations. It can be realized as objectivist and subjectivist approach. One of possible approaches in the first method relies on comparison of DSS outputs against a gold standard (known diagnoses for specific input data). Examples of this approach can be found in [27]. Literature reviews [28] show that publications put the bias on overview of outcomes tending to favour successful projects or the bias can come from evaluation of systems done by their developers. Lack of performance could also be found in non-compliance of designers to one or more of the factors leading to successful implementation of DSS mentioned in section 2.3.

Subjectivist approach to evaluation uses techniques as observation of participants in their natural (clinical) setting, interviews, analysis of documents to study the impact of DSS implementation to clinical work [29].
Combination of qualitative and quantitative evaluation methods seems to be feasible approach in studying characteristics of DSS impact and outcomes [30].

3 Clinical Workflow Systems

As the human life is the most inestimable value in our society, developing quality IT support for healthcare organization is essential. The latest developments in the business applications has shown, that separating process logic from application code and thus using Workflow Management Systems is the most efficient way as it increases efficiency and enables money savings during various changes in organizations (e.g. introducing new products, reorganization of resources, integration with IS of newly acquired companies etc.). However the main goal of the companies is to profit and their processes don't change during time or the changes are minimal. Contemporary WMSs are designed mostly for their needs. Processes in healthcare are different in respect

with dynamicity of medical knowledge evolution and specific characteristic of each patient. We can divide them into two groups – organizational processes and careflows.

Organizational processes are similar to the ones in commercial companies. They ensure working of an organization itself, cooperation between organization units, making appointments, transporting the patient, sending and evaluating reports etc. Thus they ensure resources availability (physicians, rooms, technical equipment, and information) and patient management as patients may be treated by many institutions and physicians‘ responsibilities are widely shared. They remain stable.

Careflows are workflows in medical environment - the processes which include diagnosis and treatment and they should follow medical guidelines issued by recognized authorities and medical organizations. The combination of a Petri net-based formalism for modeling clinical tasks, with a WfMS for managing the organizational process, was dubbed a ‚careflow‘ system, in which the careflow process definition describes the tasks and defines their order of execution, while the execution engine provides some flexibility by allowing tasks to be skipped or substituted with other tasks outside those defined by the clinical guideline [31]. This type of processes change due to the changes in guidelines (e.g. changes arising from adaptation of national guidelines to local conditions of particular health provider organization) and they also have many exceptions as the services provided by healthcare organization are personalized and different for each patient and diagnose. IT support for careflows systems can make health services more efficient and cost-effective. These careflow systems have to be agile and cope with real world exceptions, uncertainty and changes and at the same time they should not restrict physicians and nurses, who are trained to decide which actions to take. Medical personnel have to be free to react and gain complete initiative. Thus, the difference between careflows and usual workflows lies in changes of an ongoing process instance during run-time compared to the definition of the general process model [32]. There are several kinds of build- and run-time flexibility: adaptation, flexibility and evolution.

Adaptation represents the ability of the implemented processes to cope with exceptional circumstances [33]. There are several PAISs (Process-aware Information Systems), which provide support for handling of such exceptions, e.g. YAWL and ADEPT2. YAWL provides support for handling of expected exceptions. Logically, if the exception is expected, it can be predefined in the process modeling. ADEPT2 supports handing of unexpected exceptions. They can be handled e.g. by adding, deleting or moving process activities during run-time. ADEPT2 enables instance-specific changes of the model, which means, that not the processes change, but the instances are dynamically adapted to the

situation. Thus, users can define their own ad-hoc adaptations and use their own knowledge gained in the past instead of bypassing the PAIS when exception occurs. All the deviations are documented in change logs [33].

Flexibility represents the ability of a process to execute on the basis of a loosely or partially specified model which is completed at run-time and may be unique to each process instance. Flexible frameworks are e.g. DECLARE and Alaska. They allow individual instances to determine their own processes, which are unique. They are loosely specified using constrained-based process models approach, which is an opposite to pre-specified process model. Prespecified models define exactly how all the tasks should be accomplished; define all the activities to be executed, their control flow and data flow. They are ideal for modeling of repetitive and predictable processes. However, medical processes require different, more agile, approach as they are dynamic and have to deal with uncertainty, variability and evolution of the process over time. On the other hand, constraint-based approach describes the set of activities which may be performed and the set of constraints preventing undesired process behavior, e.g. mutual exclusion of two activities. The advantages of this approach are that, it provides much more build-in flexibility and doesn't over-specify the model. The users can decide on to make their own choices about next actions. The disadvantage is that because of high number of run-time choices, more sophisticated user support is needed. As both approaches have their advantages and disadvantages, the best way is combining both paradigms [33].

Evolution represents the ability of a process implemented in a PAIS to change when the process evolves, e.g., due to legal changes or process optimizations, which is supported by approaches like WASA2, ADEPT2, and WIDE [33]. The pre-specified processes can also change because of errors in model design, technical problems or insufficient quality. After the change came, they have to be prespecified. The problem arises, when we have many long running process instances and the process model changes before they end. The above mentioned approaches allow migration of such process instances to the new model version while the consistency is ensured [33].

Exceptions from standard behavior can be classified according to various criteria. When speaking about predictability, there are expected and unexpected exceptions. *Unexpected exceptions* are totally new situations whose management is not incorporated into the process model. Regarding synchronicity, they can be synchronous and asynchronous. *Synchronous exceptions* are those, when physicians do not accept a guideline suggestion and decide to change a treatment plan, e.g. make a CT scan instead of X-ray. At *asynchronous exception* they decide to perform an action which has no relation with the current guideline task, e.g. the physician can decide to contact patient's

relatives to gain some information. Regarding impact size we take into consideration the number of processes, patients, current and future tasks which will be influenced [34]. For example the delay in performing a test which is needed for further diagnosis influences just the particular patient. However equipment failure influences all the patients who need to be examined with it.

There are several ways of exception handling. If the task is not mandatory, it can be bypassed. Mandatory task can be executed, redirected or delayed if the physician executes more urgent task. Later the physicians receive reminders about pending tasks and they have to either execute or replace them [34].

BPM technologies achieved remarkable success in industry with growing process orientation. However, they are still not adapted in healthcare as much as necessary. The reason for this is the rigidity brought by the first generation of workflow management systems which cannot work in always changing medical environment. Hospitals need an agile system, which would be able to respond to process changes and exceptional situations, variations in the disease development or treatment process. Advanced BPM technologies able to work with flexible, adaptive and evolutionary processes are needed. However, it is important, that they don't restrict physicians and nurses in their work, as they are trained to decide about next steps of patient management. Various process support paradigms which meet above mentioned requirements were suggested, e.g. adaptive processes, case handling, constraint-based process models.

4 Clinical Practice Guidelines and Computer Interpretable Guidelines

Clinical guidelines are the documents issued by authored organizations which encapsulate the best medical practices. They are defined as systematically developed statements to assist physician and patient decisions about appropriate health care for specific circumstances [35]. They contain intentions of the guideline, medical background, patient eligibility criteria, procedural statements such as clinical algorithms and drug recommendations, evidence for the advisories, treatment cost-benefit analysis, and references [36]. They are usually published in a textual format.

Using clinical guidelines might be beneficial for physicians, as they provide a better overview of available treatments with recommendations mostly for the physicians who are not completely confident about how to proceed with their patient's treatment. They also might help to change the attitude of older

physicians who already have developed their own methods which might be outdated.

The quality of guidelines can vary. The early ones were based on consensus among experts, later they were based on more evidence-based recommendations. To recognize the quality, classifications have been developed dividing the guidelines to various levels of excellence. Usually three levels are used: A (highest), B, and C. Level A concerns data derived from multiple randomized clinical trials, level B concerns data derived from a single randomized trial or non-randomized studies, and level C concerns a consensus opinion of experts [35].

Another document type used in clinical practice is clinical protocols. They provide a comprehensive set of rigid criteria outlining the management steps for a single clinical condition or other aspects of the organization [35]. They are much more algorithmic than clinical guidelines.

Clinical guidelines are guides rather than rules. They cannot be explicit as physicians are always those who make decisions. The praxis is that clinicians don't use the guidelines much. It is because they find paper-based guidelines difficult to use in clinical practice because of inconsistencies in guidelines or they are not convicted that application of guidelines will lead to a better care; there might be organizational barriers and the guidelines are too general, not patient-specific. A guideline should not require clinicians' judgments. However, as they are not patient-specific, their judgment is needed. Solution to this problem is standardization and computerization of the clinical guidelines into computer interpretable guidelines – CIGs. They are explicit, detailed and patient data-driven, can simultaneously achieve standardization of clinical decision making and individualization of patient care [35].

The clinical guidelines development consists of four basic steps. First, the paper-bused guideline is issued by authoring institutions, e.g. American Society for Preventive Cardiology (ASPC). Then the transformation is made from paper-based form into a computer-based language. The example of such language used for modeling clinical practice is GLIF developed by InterMed Collaboratory. Its formal representation defines ontology for representing guidelines, medical data and concepts. Third step is the implementation into information system of healthcare institution. The last step is interpretation of guidelines by clinicians [35].

There are numerous advantages of *computer interpretable guidelines* (CIGs). Implementation of CIGs in Decision Support Systems (DSSs) helps users to follow the guideline. DSS retrieves patient data from an electronic patient record (EPR) and generates patient-specific advice based on the data and guideline content. However, standardization of clinical guidelines and EPRs is still needed to avoid inconsistencies. The system can generate alarms and

reminders and the guidelines are automatically checked to discover inconsistencies. As CIGs are implemented in a formalized way, they can be checked also using formal methods, e.g. Petri nets, which enables finding of conceptual errors. The computerization provides an automated method of delivering guidelines to clinicians when the guideline is the most relevant to the care of patient at the time when patient is seen by a clinician.

To sum up the advantages of CIGs – they make the guidelines patient-specific, they are easier to use, they don't have any inconsistencies as they were checked using formal methods, they are the base for DSS which provides the right information in the right time without the need to additionally search for it.

An important question is, who should be involved into the translation of the guidelines from paper-based form into CIGs. As people with different knowledge and background would create different models from the same guideline, it is obvious, that cooperation between medical personnel and clinical software developers is needed. This cooperation will ensure required combination of clinical knowledge with technical skills. The cooperation is needed not only during the computerization of clinical guidelines, but also during creation of the guidelines. The collaborators have to take into consideration that too much or too little text would lead to errors in interpretation. During the translation IT specialists help the clinicians to define the guidelines in a standardized way convenient for the computerization. The cooperation during the computerization - clinician works with an IT expert - helps to create better CIGs. The clinician defines the medical knowledge and the IT expert transforms it into CIG. During this transformation they can consult possible ambiguities in guidelines that occur during the translation process. Thus, the advantages of the cooperation are elimination of vague, erroneous and incomplete recommendations at the beginning. It improves the logical consistency and completeness of clinical algorithms [35].

During clinical guidelines computerization templates called design patterns can be used. They are computer-interpretable templates for representing guideline knowledge using clinical abstractions that are appropriate for particular guideline sub-domains [35].

Two main approaches for integration of textual guidelines with formal knowledge bases of DSS are document-centric and knowledge-base-centric approach.

Document-centric approach uses markup methodologies like HTML or XML, which makes the use of guidelines as easy as browsing internet web pages. Comparing to paper-based guidelines, markup methods are a great improvement. Marked-up documents allow easy browsing and better linking concepts with text. While creating CIGs, first step is marking up a document with relevant tags. Then it is translated into computable statements. The

greatest advantages of this approach are that it can be marked-up to any level of granularity. Thus the system can be designed in stages, which enables creation of prototypes. Moreover, it also provides different displaying for different roles present during medical treatment process, e.g. physicians, nurses, patients, hospital administrators. Using appropriate ontology, all the relationships between the roles can be preserved. The disadvantage is, that mark-up methodology does not lend itself to representing complex decision logic. We need people to read the texts and apply a guideline manually. Besides technical skills the IT specialists also need medical knowledge. There are also problems when trying to combine information from multiple documents.

Examples of these systems are ActiveGuidelines by Epic Research institute or GEM system developed at Yale University. ActiveGuidelines is a system, which uses only minimal structure to mark-up guideline documents. The biggest advantage is that it is integrated with electronic patient record (EPR) and when the guideline criteria are met for the specific patient, the system provides the clinician with suitable recommendations. Recommended orders are marked-up and contain needed information. The clinician can browse through them and choose which to accept. In the end, the orders are automatically transferred to EPR. GEM is a system with very different approach as it is meant for health service research as well as for clinical DSSs. Its goal is to create a comprehensive mark-up system. The problem is that guideline documents are meant for human audience and may not be suitable for DSSs. GEM's solution is, that its ontology allows marking up the documents suitable for different views (physicians, nurses, patients) [36].

The *knowledge-base-centric approach* uses knowledge base and provides a complex knowledge model with recommendations adjusted to each patient. In process of knowledge base creation, the domain expert first gathers information from a guideline text, interprets it and encodes in a computable form in formalism suitable for knowledge base [36]. The computer system uses patient data, combines it with knowledge base and generates recommendations. Well structured recommendations can provide also access to corresponding part of guideline in textual form, for the purposes of generating explanations related to inferred recommendations. Another advantage of this approach is that system can combine information from different guideline documents. The problem arise when guideline document changes. Then the links between knowledge base and the document have to be updated and verified which is not always an easy task to perform.

Computable formalizations suitable for creation of knowledge base are EON and GLIF3. The EON system uses a framework based knowledge-engineering environment to model declarative domain knowledge. The system links textual supporting materials to knowledge elements, which are used to

generate patient specific recommendations [36]. GLIF3 is a specification for structured representation of guidelines that aims to facilitate sharing of clinical guidelines. Its objective is to provide a precise, human-readable, computable and multiplatform guideline representation [37].

As both approaches (document centric vs. knowledge-base-centric) have their pros and cons, there have been efforts to combine their good characteristics and create so called bridge approach. As both approaches have difficulties integrating guideline text with DSS, there are efforts to integrate structured knowledge base with marked-up text. Desirable characteristic of such DSS are: flexibility of document markups, preciseness of knowledge base elements, facilitation of easy mapping between the two and maintenance of links between knowledge base and guideline document.

The bridge approach uses an information markup search and retrieval technology to bridge the orthogonal divide between the knowledge base and the guideline text [36]. It associates clinical queries with knowledge base elements and uses information retrieval technology to find relevant parts of the guideline text. Thus there are no absolute references to the guideline text and the link between the text and knowledge base becomes more dynamic. When the marked-up source guideline document changes, it is no more needed to review and update the knowledge base. The system updates it using the existing queries in the knowledge base for a new guideline document. Another advantage taken from document-centric approach is, that viewing data in any level of detail is possible.

5 Logical and Fuzzy Petri Nets

As mentioned in Section 2 decision support systems consist of knowledge base and inference engine. While knowledge base encodes the expert knowledge with application of various knowledge representation formalisms (Section 2.2), inference engine derives the recommendations over knowledge and different types of input data (values of physiological signals, laboratory results etc.). Typical characteristic of medical reasoning is that clinicians are often forced to make decision under uncertainties resulting from vague answers from patients, unavailability of required patient data at the point of care, lack of available quality guidelines etc.

Mapping all those forms of inexactness onto a structured parallel distributed architecture may result in increasing of the reasoning efficiency. Logical and fuzzy Petri nets as directed bipartite graphs with a significant degree of structural parallelism and pipelining seems to be a good choice for knowledge representation and reasoning in DSS. There are several dialects of

logical and fuzzy Petri nets in the literature [38, 39, 40, 41]. For our purposes the approach presented in [41] and further developed in [19] is the most suitable.

For knowledge representation we use a set of propositions which can have the values true or false in the case of a logical knowledge base. For knowledge propagation we consider a set of production rules. Production rule describes the relation between two sets of propositions. Set A of propositions represents the antecedent of the production rule and a set B of propositions represent the consequent of the production rule. The knowledge is propagated by firing of a production rule. The interpretation of firing a production rule is following:

IF all propositions in the antecedent A have value true THEN the propositions in the consequent B are true.

We consider a simple knowledge base given by a set of propositions and a set of production rules of the following form: logical product of the propositions in the antecedent A implies the logical product of the propositions in the consequent B. Generally, the knowledge is propagated by firing a sequence of these rules, where the consequent of one rule is used as the antecedent of the next rule.

In many cases the validity of a proposition is not always certain. For such cases, it is suitable to use fuzzy values for the propositions. In the case of a fuzzy knowledge base fuzzy values from the closed interval of real values *<0,1>* can be used, where value 0 represents the case in which the proposition is not true and the value 1 represents the case in which the proposition is true. Values between 0 and 1 represent the measure of validity for the proposition. For example consider the following proposition *"The temperature of a patient is high"*. Obviously the validity of this proposition is uncertain. We know that this proposition is more valid if a patient has temperature 40°C than if he has temperature 38°C. In the cases when the validity of propositions is expressed by fuzzy values also the relation between propositions of the antecedent and propositions of the consequent is fuzzy. A production rule with fuzzy relation is called fuzzy production rule. The mechanism of firing the fuzzy production rule and the knowledge propagation in a fuzzy knowledge base will be explained in the section 5.2 devoted to fuzzy Petri nets.

5.1 Logical Petri Nets

A logical Petri net (*LPN*) is defined as 4-tuple:

$$LPN = (P,T,F, m), \quad (1)$$

where $P = \{p_1,p_2, \ldots p_n\}$ is a finite set of places, $T = \{t_1,t_2, \ldots,t_m\}$ is a finite set of transitions, F is a finite set of ordered pairs (p_i,t_j) defining input places, and (t_j,p_i) defining output places. $m: P \rightarrow \{0, 1\}$ is an association function, a mapping from places to real value 0 and 1 called marking. In case of *LPN* value 0 represents logical false and value 1 represents logical truth.

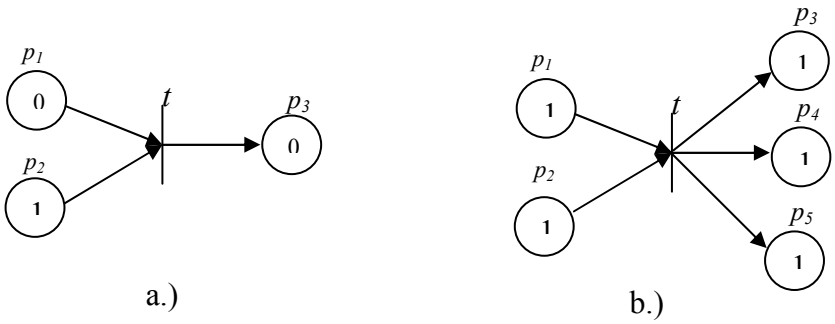


Fig. 3. Firing of transition t in *LPN*
a.) not enabled, b.) enabled

Definition of enabled transition: Transition $t \in T$ is enabled if the marking of all input places $(p_i,t) \in F$ is equal to 1. Formally we can express the enabled transition t as a logical *AND*:

$$AND(m(p_1),.....,m(p_i)) \begin{cases} = 1 \text{ transition } t \text{ is enabled} \\ = 0 \text{ transition } t \text{ is not enabled} \end{cases} \quad (2)$$

Based on (2) we can define the firing transition as a change of the marking of *LPN*. This occurs when the enabled transition t fires so that all output places $(t,p_i) \in F$ of a transition t have value equal to 1. Markings from the input places of the transitions are not removed after firing as in classical Petri net [25]. In case that transition t is not enabled it cannot fire, so the output place receives value 0. We can say that transition that is not enabled fires with value 0 to output places. If a transition has many output places all of them receive the same marking (Fig. 3). In case that one place p is an output place of several transitions $(t_i,p) \in F$, the marking of such place is calculated applying logical function *OR* (Fig. 4).

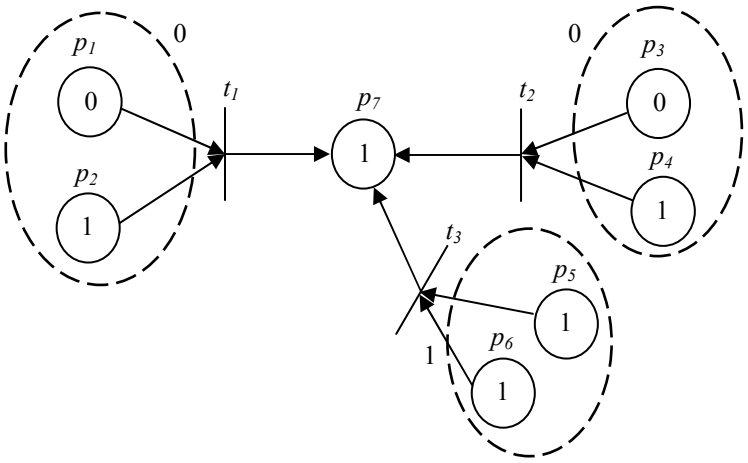


Fig. 4. Marking of output position p_7

Calculating marking of output place:

$$m(p^*) = OR\ (AND\ (m(p_1), ...,m(p_i))^t{}_1,....., AND(m(p_j), ...,m(p_k))^t{}_l), \quad (3)$$

where $m(p^*)$ represents the marking of output place, k represents the number of all considered places and l represents the number of all considered transitions.
To calculate the marking of place p_7 shown on Fig. 4 we use *OR* on contributions from transitions t_1, t_2, t_3 :

$$m(p_7) = OR\ (AND\ (m(p_1),m(p_2))^t{}_1,\ AND\ (m(p_3),m(p_4))^t{}_2\ ,AND\ (m(p_5),m(p_6))^t{}_3) =$$
$$= OR(AND\ (0,1),\ AND\ (0,1),\ AND\ (1,1)) = OR(0,0,1) = 1. \quad (4)$$

5.2 Fuzzy Petri Nets

Because the logical Petri net cannot deal with vague or fuzzy information such as „very good“ and “healthy” fuzzy Petri nets have been introduced [38]. They are used for fuzzy knowledge representation and reasoning. A fuzzy Petri net (*FPN*) is defined as 4-tuple:

$$FPN = (P, T, F, m) \quad (5)$$

P, T, F is finite sets defined as in previous example. $m: P \rightarrow [0, 1]$ is an association function, which assigns a real value between zero to one to each place called marking.

Definition of enabled transition: transition $t \in T$ is enabled if the marking of all input places $(p_i,t) \in F$ is larger than 0 (Fig. 5).
Formally we can express the enabled transition t as a fuzzy norm T_n [42]. In our case we use a simple function T_3.

$$T_n(m(p_1), ...,m(p_i)) = T_3(m(p_1), ...,m(p_i)) = min(m(p_1), ...,m(p_i)) \quad (6)$$

$$min(m(p_1), ...,hmp_i)) \begin{cases} > 0 \text{ transition } t \text{ is enabled} \\ = 0 \text{ transition } t \text{ is not enabled} \end{cases} \quad (7)$$

Based on (6) and (7) we can define the firing transition as a change of the marking of *FPN*. This occurs when the enabled transition t fires, so that all output places $(t,p_i) \in F$ have value calculated as in (6). Same as in *LPN*, markings in the input places of the transitions are not removed after firing. In case that transition t is not enabled it cannot fire, so the output place receives value 0. Same as in *LPN* not enabled transition fires with value 0. If one transition has many output places all of them receive the same marking (Fig. 5). In case that one place p is an output place of several transitions $(t_i,p) \in F$, the marking of such place is calculated applying fuzzy norm S_n [42]. In our example we use a simple function S_3. Marking of the output place $m(p^*)$ is as follows:

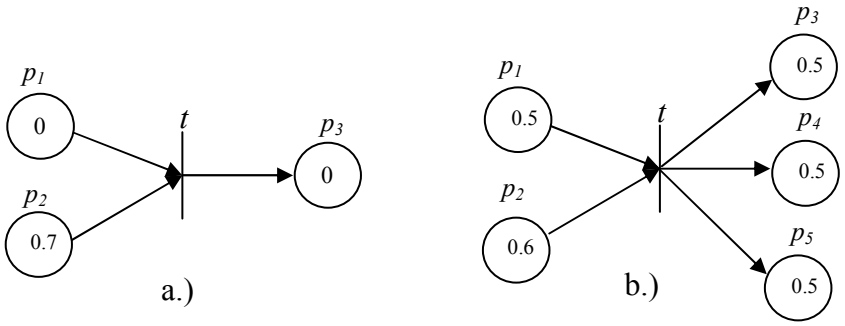


Fig. 5. Firing of transition t in *FPN*
a.) not enabled, b.) enabled

$$\begin{aligned} m(p^*) &= S_n\ [min(m(p_1),.....,m(p_i)^t{}_1,.....,min(m(p_j),.....,m(p_k))^t{}_l] \\ &= S_3\ [min(m(p_1),.....,m(p_i))^t{}_1,.....,min(m(p_j),.....,m(p_k))^t{}_l] \\ &= max[min(m(p_1),.....,m(p_i))^t{}_1,.....,min(m(p_j),.....,m(p_k))^t{}_l] \end{aligned} \quad (8)$$

where k represents the number of all considered places and l represents the number of all considered transitions.

To estimate the marking of place p_7 shown on Fig. 6 we use *max* on contributions from transitions t_1,t_2, t_3.

$$m(p_7) = max\ (min\ (m(p_1),m(p_2))^t{}_1,\ min\ (m(p_3),m(p_4))^t{}_2\ ,min\ (m(p_5),m(p_6))^t{}_3) =$$
$$= max\ (min\ (0.2,0.7),\ min\ (0.5,0.8),\ min\ (0.6,0.9)) = max(0.2,0.5,0.6) = 0.6 \quad (9)$$

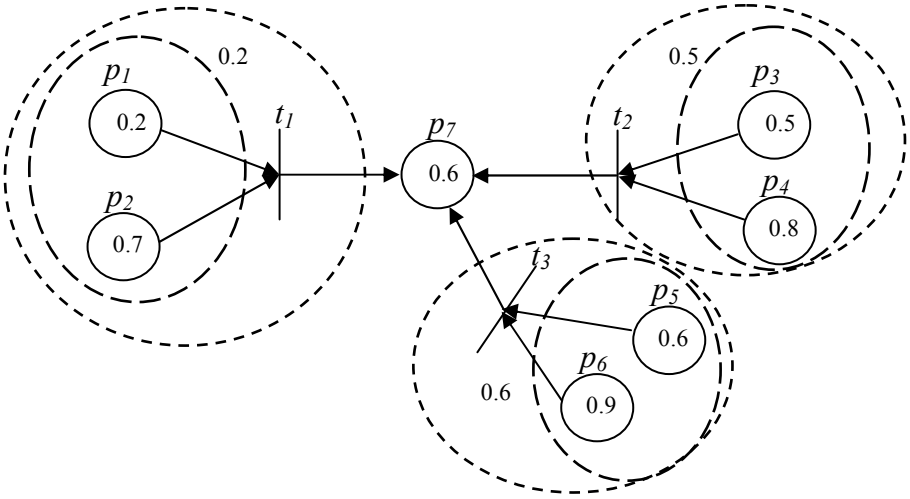


Fig. 6. Marking of the position p_7 in *FPN*

5.3 Knowledge representation and propagation with Petri Nets

In previous sections we have shown definitions of logical and fuzzy Petri nets. In introduction to the Section 5 we described a simple knowledge base in a form of logical and fuzzy production rules which describes the relation between two propositions e_i, e_j representing the antecedent – consequent pairs of the rule *pr: IF* e_i *THEN* e_j.

Considering Petri nets (both logical and fuzzy) as knowledge representation formalism each transition with its input and output places represents a production rule. Input and output places are represented by a propositions e_i, e_j. Markings m of the input and output places represent weights w (or degree of truth) of the respective input and output propositions e. In logical Petri net shown in Fig. 3 the input place p_2 may be represented by the following input proposition e_2 = "The patient's body temperature is 39 degrees". Because this proposition describes the true health state of the patient its weight is $w(e_2) = 1$ i.e. $m(p_2)=1$.

In the case of fuzzy Petri nets we deal with a different kind of information than in logical Petri net. This type of net is used to describe vague or fuzzy information such as "very high" or "good". Let's consider the net shown in Fig. 5. The input place p_2 may be represented by the input proposition e_2 = "The patient's body temperature is high". Based on the fuzzy set theory [42] we can say that the weight of the proposition e_2 is $w(e_2) = 0.7$ i.e.

$m(p_2)=0.7$. It is important to stress that we still didn't say anything about the weights or thresholds of the rules. This will be mentioned in detail in further text related to knowledge propagation.

Benefits of knowledge representation and propagation formalism based on logical and fuzzy Petri nets that will be presented in detail in further text is that it fulfils two important requirements for DSS:

- provision of unique recommendation for fixed values of inputs
- generation of recommendation in finite time for any values of input parameters.

Both of these statements are proved in [19].

For the purposes of the knowledge propagation we introduce the following definitions. For vectors $\boldsymbol{a}$, $\boldsymbol{b}$ defined as

$\boldsymbol{a}= (a_1 , a_2, \ldots, a_n); \boldsymbol{b}=(b_1 ,b_2, \ldots, b_n)$ (10)

we have defined ($\boldsymbol{r}$ x $\boldsymbol{n}$)-matrix $\boldsymbol{Y}$ with rows $\boldsymbol{y_1},\ldots,\boldsymbol{y_r}$.

Definitions of operations *or* , *and*, *neg* for logical Petri nets are shown below.

$\boldsymbol{a}$ *or* $\boldsymbol{b}$ = $(a_1 \vee b_1, a_2 \vee b_2, \ldots, a_n \vee b_n)$ *is an n-dimensional vector.* (11)

$\boldsymbol{a}$ *and* $\boldsymbol{b}$ = $(a_1 \wedge b_1) \vee (a_2 \wedge b_2) \ldots \vee (a_n \wedge b_n)$ *is a scalar.* (12)

neg $\boldsymbol{a}$ = $(\neg a_1, \ldots, \neg a_n)$ *is an n-dimensional vector.* (13a)

$\boldsymbol{Y}$ *and* $\boldsymbol{a}$ = ($\boldsymbol{y_1}$ *and* $\boldsymbol{a}$,..., $\boldsymbol{y_r}$ *and* $\boldsymbol{a}$). (13b)

Here the symbols $\vee$ $\wedge$ and $\neg$ denote the usual Boolean operators.

For fuzzy Petri nets the definition is based on fuzzy set theory:

$\boldsymbol{a}$ *or* $\boldsymbol{b}$ = $(max(a_1, b_1),max(a_2,b_2),\ldots,max(a_n,b_n))$. (14)

$\boldsymbol{a}$ *and* $\boldsymbol{b}$ = $max(min(a_1, b_1),min (a_2,b_2),\ldots, min (a_n,, b_n))$ (15)

neg $\boldsymbol{a}$ = $(1-a_1, \ldots, 1-a_n)$ (15a)

$\boldsymbol{Y}$ *and* $\boldsymbol{a}$ = ($\boldsymbol{y_1}$ *and* $\boldsymbol{a}$,..., $\boldsymbol{y_r}$ *and* $\boldsymbol{a}$). (15b)

$\boldsymbol{I}$, $\boldsymbol{O}$ are $\boldsymbol{r}$ x $\boldsymbol{n}$ dimensional matrices, where r represents propositions and n represents rules. Input matrix $\boldsymbol{I}$ describes the relation between input propositions and the rules. For each i_{ab} we place 1, if the input proposition e_a belongs to rule pr_b. Otherwise the value of i_{ab} equals 0. Output matrix $\boldsymbol{O}$ describes the relation between output proposition and the rules. For each o_{ab} we place 1, if the output proposition e_a belongs to rule pr_b. Otherwise the value of o_{ab} equals 0. These matrices describe clearly the situation in which a proposition may be output proposition of one rule but in the same time input proposition of another rule (for example relationship between proposition e_5 and rules pr_1 and pr_4 as shown on Fig.7).

Initial marking of Petri net (both *LPN* and *FPN*) is represented by vector $\boldsymbol{W_0}=(w(e_1),\ldots., w(e_n))$ with the weights of propositions $w(e)$. In case of $w(e)=0$ we say that the proposition is not valid.

$\boldsymbol{W_k}$ represents the marking of Petri net in iteration k. $\boldsymbol{A}^T$ represents the transposed matrix (or vector) $\boldsymbol{A}$. Knowledge propagation algorithm consists of the following steps:

Step 1: Arrange matrices $\boldsymbol{I}$; $\boldsymbol{O}$

Step 2: Calculate vector $\boldsymbol{V_k}$

Step 3: Calculate vector $\boldsymbol{U_k}$

Step 4: Calculate vector $\boldsymbol{W_{k+1}}$

Step 5: Go to *Step 2* until $\boldsymbol{W_{k+1}} = \boldsymbol{W_k}$

Vector $\boldsymbol{V_k}$ represents which rules in the marking $\boldsymbol{M_k}$ are not enabled. Vector $\boldsymbol{X_k}$ represents which rules for the marking $\boldsymbol{W_k}$ are enabled. Marking $\boldsymbol{W_{k+1}}$ represents the next marking of the net.

5.4 Knowledge propagation in logical Petri net

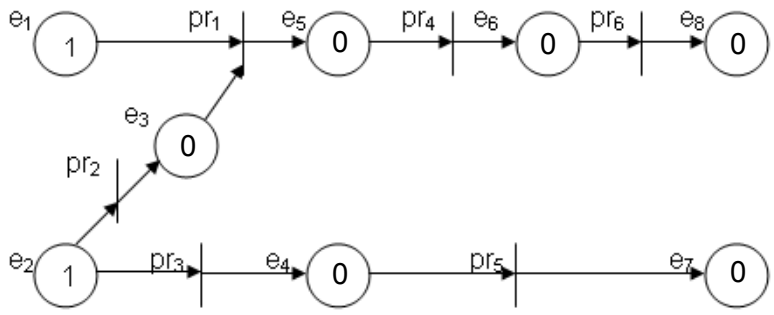


Fig. 7. Initial marking $\boldsymbol{W_0}$ of logical Petri net

For the purposes of algorithm illustration we use a net on Fig.7. The net contains six rules pr_1 to pr_6 with appropriate propositions in its antecedents and consequents. The initial weights of the propositions are represented by vector $\boldsymbol{W_0} = (w(e_1), w(e_2), w(e_3), w(e_4), w(e_5), w(e_6), w(e_7), w(e_8)) = (1,1,0,0,0,0,0,0)$.

Step 1 Arrangement of *I* an *O* matrices

$$I = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad O = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Step 2 Calculate vector V_k

$$V_0 = I^T \ \underline{and} \ (\underline{neg}\ W_0) \tag{16}$$

$$V_0 = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \underline{and}\ (0\ 0\ 1\ 1\ 1\ 1\ 1\ 1)$$

If we consider a first row of the matrix I^T the value of the element v_{01} in vector V_0 is:

v_{01} = *(1 AND 0) OR (0 AND 0) OR (1 AND 1) OR (0 AND 1) OR (0 AND 1) OR (0 AND 1) OR (0 AND 1) = 1*

$v_{02} = 0$; $v_{03} = 0$; $v_{04} = 1$; $v_{05} = 1$; $v_{06} = 1$.

Resulting vector $V_0 = (1,0,0,1,1,1)$ represents which rules based on the initial marking W_0 are not enabled. In this case not enabled rules are pr_1, pr_4, pr_5, pr_6. If we look closer on the creation of v_{01} we can say that first row of matrix I^T represents which input propositions are connected with rule pr_1. Vector $(\underline{neg}\ W_0)$ tells us which propositions are not marked. We can say that propositions e_1 and e_3 are connected to the rule pr_1 but only proposition e_1 is marked. Therefore, based on the definition of enabled rule in logical Petri net (2) we can say that this rule is not enabled. The same approach may be applied for the other elements of vector V_0.

Step 3 Calculate vector U_k

$$U_0 = \underline{neg}\ V_0 = \underline{neg}\ (I^T \ \underline{and}\ (\underline{neg}\ W_0)) = (0,1,1,0,0,0) \tag{17}$$

Negation of vector V_0 resulting in information about enabled rules (pr_2, pr_3).

Step 4 Calculate vector W_{k+1}

$$W_1 = W_0 \underline{or} (O \underline{and} U_0) \tag{18}$$

$$W_1 = (1,1,0,0,0,0,0,0\ ) \underline{or} \left(\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \underline{and}\ (0,1,1,0,0,0) \right)$$

Let's look closer on the meaning of vector $(O \underline{and} U_0)$. The output matrix O describes connection between the rules and their output propositions. U_0 says which rules are enabled. If we consider the third row of matrix O we can say that proposition e_3 is related to rule pr_2. Vector U_0 states that rule pr_2 is enabled. Therefore we can say that rule pr_2 can fire and propagates the input information further. In our case it propagates the weight of the proposition e_2 to the proposition e_3. After completing the *and* operation for each row, vector $(O \underline{and} U_0)$ says that next marking of logical Petri net is: $(O \underline{and} U_0) = (0,0,1,1,0,0,0,0)$.

Because markings from the input propositions of the rules are not removed after firing we use the initial marking W_0 to preserve this information in the new marking W_1. After four iterations ($k = 4$) we reach the "stable state" of our logical Petri net $W_{k+1} = W_k$. Final marking (Fig. 8) of the net is $W_5 = (1,1,1,1,1,1,1,1)$.

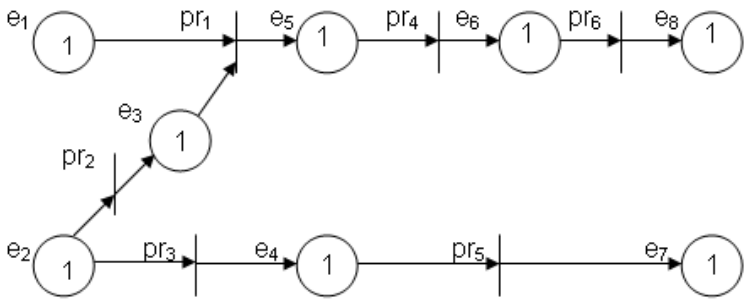


Fig. 8. Final marking W_5 of *LPN*

5.5 Knowledge propagation in Fuzzy Petri net

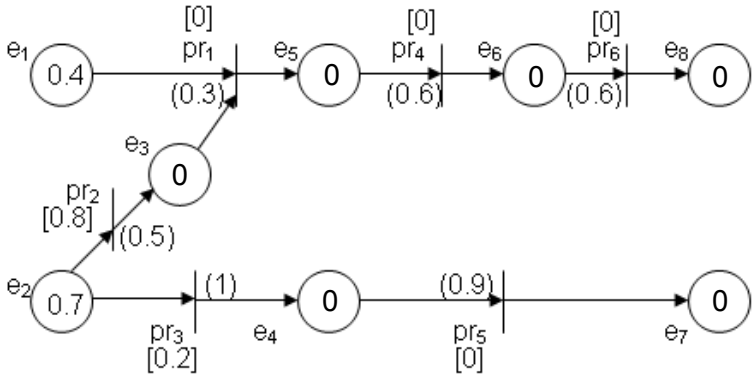


Fig. 9. Initial marking W_0 of fuzzy Petri net

We already said that using the fuzzy Petri net it is possible to model vague or imprecise information such as "high", "healthy" etc. To model the knowledge propagation more precisely we introduce the weights (or certainty factors) and thresholds of the rules [19] [41]. Threshold represents a lower bound on the degree of rule with respect to the weight of the rule as its higher bound. Vector ***WR*** represents the rule weights and vector ***Th*** represents the thresholds of the rules. For the fuzzy Petri net shown on Fig. 9 we can define the following vectors:

$\boldsymbol{WR} = (wr(pr_1), wr(pr_2), wr(pr_3), wr(pr_4), wr(pr_5), wr(pr_6)) = (0.3, 0.5, 1, 0.6, 0.9, 0.6)$

$\boldsymbol{Th} = (th(pr_1), th(pr_2), th(pr_3), th(pr_4), th(pr_5), th(pr_6)) = (0, 0.8, 0.2, 0, 0, 0)$

Initial marking $\boldsymbol{W_0} = (0.4, 0.7, 0, 0, 0, 0, 0, 0)$

Let's consider the weight of the proposition e_2, $w(e_2) = 0.7$. This is an input proposition of rules pr_2 and pr_3 with weights $w(pr_2)=0.5$ and $w(pr_3)=1$. Taking the minimum of the proposition weight and the rule weight we have $min(w(e_2), w(pr_2)) = (0.7, 0.5)=0.5$; $min(w(e_2), w(pr_3)) = (0.7, 1)=0.7$. If these values are bigger than a corresponding rule threshold the rule is enabled and may fire. In our case we see that only $min(w(e_2), w(pr_3)) > p(pr_3)$ then only rule pr_3 is enabled and may fire. Minimum of the values $w(e_2), w(pr_3)$ determines the value that propagates further. In our case the value that propagates is 0.7: $\min(w(e_2), w(pr_3)) = min(0.7, 1)) = 0.7$. Application of knowledge propagation algorithm is shown below.

Step 1 Arrangement of *I* an *O* matrices

The structure of fuzzy Petri net on Fig. 7 is the same as logical Petri net on Fig. 5 so the matrix ***I*** doesn't change. Taking into consideration the rule weights, we define the output matrix $\underline{\boldsymbol{O}}$:

$$\underline{\boldsymbol{O}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0.3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6 \end{pmatrix}$$

Step 2 Calculate vector V_k

$\boldsymbol{V_0} = \boldsymbol{I}^T$ *and* *(neg* $\boldsymbol{W_0}$*) = (1,0.3,0.3,1,1,1)*

Vector $\boldsymbol{V_0}$ represents which rules based on the initial marking $\boldsymbol{W_0}$ are not enabled. In this case not enabled rules are same as in previous example pr_1, pr_4, pr_5, pr_6. Not enabled rules are marked by 1.

Step 3 Calculate vector U_k

Negation of vector $\boldsymbol{V_0}$ results in information about enabled rules (pr_2, pr_3). In this case we need to calculate $\boldsymbol{V_0}$ with respect to the rule thresholds *th*. For this purpose we introduce new operation *top* defined over two vectors $\boldsymbol{a}=(a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b}=(b_1, b_2, \ldots, b_n)$, $\boldsymbol{a}$ *top* $\boldsymbol{b} = (c_1, c_2, \ldots, c_n)$, where $c_i = a_i$ if $a_i \geq b_i$, else $c_i = 0$.

Vector $\boldsymbol{U}_k$ is calculated as:

$\boldsymbol{U}_k$ *= (neg* $\boldsymbol{V}_k$*) top* ***Th*** *= (0,0.7,0.7,0,0,0)*

For $\boldsymbol{V_0}$: *neg* $\boldsymbol{V_0}$ *= (0,0.7,0.7,0,0,0)*

Th*= (0,0.8,0.2,0,0,0)*

$\underline{\boldsymbol{U_0}}$ *=(neg* $\boldsymbol{V_0}$ *) top* ***Th*** *= (0,0.7,0.7,0,0,0) top (0,0.8,0.2,0,0,0) = (0,0,0.7,0,0,0)*

Enabled rules are those for which value in vector $\underline{\boldsymbol{U_0}}$ is different than zero. In our case only the rule pr_3 is enabled.

Step 4 Calculate vector W_{k+1}

$\boldsymbol{W_1} = \boldsymbol{W_0}$ *or (*$\underline{\boldsymbol{O}}$ *and* $\underline{\boldsymbol{U}}_0$*)* (19)

The meaning of the vector *(**O** and **U**$_0$)* is the same as in example with logical Petri net, only now it takes into consideration the rule weights and thresholds. Following marking in respect of preserving the initial marking W_0 is:

$W_1 = (0.4,0.7,0,0.7,0,0,0,0)$

After three iterations ($k = 3$) we reach the "stable state" of fuzzy Petri net $\boldsymbol{W_{k+1} = W_k}$. Final marking (Fig. 10) of the net is $\boldsymbol{W_3} = (0.4,0.7,0,0.7,0,0,0.7,0)$.

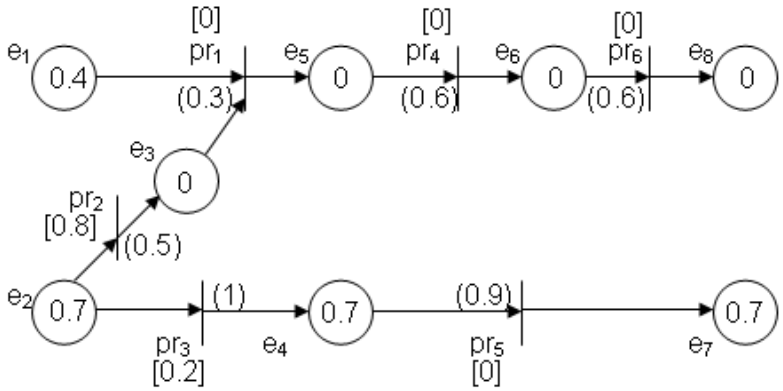


Fig. 10. Final marking $\boldsymbol{W_3}$ of fuzzy Petri net

6 Discussion

The chapter provided an overview of various approaches and aspect of developments in clinical decision support systems. Despite many identified challenges on factors leading to successful DSS implementation and respective methodologies that address them we believe that whole approach cannot be only about technology. Based on our expertise from development of clinical information systems one of the most important issues is communication with clinicians about their needs and perspectives on improvements of existing clinical processes. Even though they are not expected to be technology "geeks" often they have a very inspirational point of view about IT systems and roles that they should fulfill. One of these views is that clinicians work cannot be determined or in any way constrained by ICT. It is not true that clinicians are against the technology. ECG, EEG is in usage for a long time and a modern operation room is full of various technologies. However all these provide better access to information or in some other way assist the clinician in provision of better healthcare while maintaining his full autonomy in clinical setting.

Standardization efforts on international level (HL7, CEN) and national aspirations (like NCZI in Slovakia, www.nczisk.sk) will play more and more important role in the future. Emergence of large quantities of medical knowledge and patient data is one of the major drivers in application of decision support systems. Benefits from the information technology will

however not be significant if they are not droved by standardizations for example in knowledge representation, integration and EHR systems. Reduction of errors in clinical practice will be important issue in order to improve the healthcare services but also to effectively manage the increasing costs. All this posts an exciting challenge to researchers in domain of artificial intelligence and medical informatics related to knowledge management, representation and reasoning methods.

Acknowledgment: The authors would like to acknowledge the support for research under following projects: Measuring, Communication and Information Systems for Monitoring of Cardiovascular Risk in Hypertension Patients (APVV-0513-10); Analytics Services for SMARTer Healthcare (IBM SUR project); Research & Development Operational Programme for the project Support of Center of Excellence for Smart Technologies, Systems and Services I and II (ITMS 26240120005, ITMS 26240120029), Competence Centre of Intelligent Technologies for Electronisation and Informatisation of Systems and Services (ITMS 26240220072) co-funded by the ERDF.

References

[1] United Nations, Department of Economic and Social Affairs, Population division. World Population Ageing 2009 http://www.un.org/esa/population/publications/WPA2009/WPA2009_WorkingPaper.pdf.
[2] International Telecommunication Union, National e-Strategies for development, 2010 http://www.itu.int/ITU-D/cyb/estrat/estrat2010.html.
[3] Healy, J.C.: The WHO eHealth Resolution – eHealth for all by 2015, Methods Inf. Med. 46 (2007), pp. 2-4.
[4] Gartner Hype Cycle of Telemedicine, 2011 http://www.gartner.com/id=1754914
[5] Valky, G, Lehocki, F: Modern approach in multiple patients ECG monitoring, BHI 2012, Hong-Kong, January 2012.
[6] Jovanov, E. et. al.: Guest Editorial Body Sensor Networks: From Theory to Emerging Applications,IEEE Transactions on Information Technology in Biomedicine, Vol.13, Issue 6, November 2009, pp. 859 - 864.
[7] Haux, R: Medical informatics: Past, present, future, International Journal of Medical Informatics 79 (2010), pp. 599-610.
[8] Osheroff, JA. et. al.: Improving outcomes with clinical decision support: an implementer's guide, Health Information Management and Systems

Society; 2005.
[9] Garg, AX et. al.: Effects of computerised clinical decision support systems on practitioner performance and patient outcomes: a systematic review, JAMA 2005: 293 (10), pp. 1223-38.
[10] Berg, M: Patient care information systems and healthcare work: A sociotechnical approach, Int J Med Inf 1999;55(2), pp. 87-101.
[11] Peleg, M., Tu, S.: Decision support, knowledge representation and management in medicine. IMIA Yearbook of Medical Informatics, 2006, pp. 72-80.
[12] Aalst, W, Stahl C: Modelling Business Processes, MIT Press, 2011.
[13] Weske, M: Business Process Management, Springer, 2007.
[14] Quaglini, S et. al.: Guideline based careflow systems, Artif Intell Med 2000; 5(22), pp. 5-22.
[15] Osheroff, JA. et. al.: Clinical Decision Support Implementer's Workbook, HIMSS, 2004.
[16] Harrington, P: Machine Learning in Action, Manning Publications, 2012,
[17] Gruber, T. R.: A translation approach to portable ontologies. Knowledge Acquisition, 5(2):199-220, 1993.
[18] Gruber, T. R.: Toward Prinicples for the Design of Ontologies Used for Knowledge Sharing, International Journal Human-Computer Studies 43, p.907-928, 1993.
[19] Lehocki, F., Juhás, G., Lorenz, R., Szczerbicka, H., Drozda, M.: Decision support with logical and fuzzy Petri nets. Cybernetics and Systems, vol. 39, 2008, no. 6, pp. 617-640.
[20] Sittig, D: Grand challenges in clinical decision support, Journal of Biomedical Informatics 41 (2008), pp. 387-392.
[21] Fox, J et al.: Delivering clinical decision support services: There is nothing as practical as a good theory, Journal of Biomedical Informatics 43 (2010), pp. 831-843.
[22] Sutton, Dr, Fox, J: The syntax and semantics of the PROforma guideline modelling language, J am Med Inform Assoc 2003; 10(5), pp. 433-43.
[23] Osheroff, JA. et al.: A roadmap for national action on clinical decision suppor, J Am Med Inform Assoc 2007, 14(2), pp. 141-5.
[24] Peleg, M, Tu, S, et al.: Comparing computer-interpretable guideline models: a case study approach, J Am Med Inform Assoc 2002, 10(1), pp. 52-68.
[25] Murata, T: Petri Nets: Properties, Analysis and Applications, Proceedings of the IEEE, vol. 77, no. 4, April 1989.
[26] Kawamoto, K.: Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success, BMJ, doi:10.1136/bmj.38398.500764.8f (published March 2005).

[27] Lehocki, F., Píš, P., Kukučka, M.: Estimating the efficiency of medical diagnostic
systems. IEEE EMBC 2004, San Francisco, USA, September 2004, pp. 1028 – 1031.
[28] Garg, Ax et al.: Effects of Computerized Clinical Decission Support systems on Practitioner Performance and Patient Outcomes: A Systematic Review, JAMA 2005, 293(10), pp. 1223-1238.
[29] Friedman, CP: Evaluation Methods in Medical Informatics, Handbook of Medical Informatics, Springer, 1997.
[30] Heathfield, H et al.: Evaluating information technology in healthcare: barriers and challenges, BMJ 1998, 316(7149), pp. 1959-1961.
[31] Gooch, P, Roudsari, A: Computerization of workflows, guidelines, and care pathways: a review of implementation challenges for process-oriented health information systems, JAMIA 2011 (18), pp. 738-748.
[32] Reichert, M et al.: Flexibility in process-aware information systems, LNCS ToPNoC, 2009 (2), pp. 115-135.
[33] Reichert, M: What BPM technology can do for healthcare process support, AIME 2011, Bled, Slovenia.
[34] Quaglini , S et al.: Flexible guideline-based patient careflow systems, Artificial Intelligence in Medicie, 22(2001), pp. 65-80.
[35] Berendsen, A et al.: From Clinical Practice Guidelines to Computer-interpretable Guidelines, Methods Inf Med, 6/2010, pp. 550-570.
[36] Shankar, RD et al.: Integration of Textual Guideline Documents with Formal Guideline Knowledge bases, AMIA Symp. 2011, pp. 617-621.
[37] Khoumbati, K et al.: Handbook of research on Advances in Health Informatics and Electronic Healthcare Applications, IGI Global, 2009.
[38] Li, X. and Lara – Rosano, F. 2000. Adaptive Fuzzy Petri Nets for Dynamic Knowledge Representation and Inference. Expert Systems with Applications, vol. 19, pp. 235-241.
[39] Girault, F., Pradin-Chézalviel, B. and Valette, R. 1997. A logic for Petri Nets, RAIRO-APII-JESA Journal, vol 31,n. 3, pp. 525-542.
[40] Cardoso, J., Valette, R., and Pradin-Chezalviel, B. 1993. Fuzzy Petri nets and linear logic. In Conference proceedings of Systems, Man and Cybernetics, vol. 2, pp. 258-263.
[41] Chen, S. M., Ke, J., Chang, J. 1990. Knowledge Representation Using Fuzzy Petri Nets. IEEE Trans. on Knowledge and Data Engineering, vol. 2, no. 3.
[42] Zimmermann, H. J. 1990. Fuzzy Set Theory and its Applications. Kluwer Academic Publishers, pp. 23-39.

Ethology-Inspired Design of Autonomous Creatures in Domain of Artificial Life

Pavel NAHODIL, Jaroslav VÍTKŮ[1]

Abstract. This chapter describes novel methods of designing of autonomous agents. We inspire ourselves in fields as AI, Ethology and Biology, while designing our agents. Typical course of agent's life is similar to newly born animal, which continuously learns itself: consequently from basic information about its environment towards the ability to solve complex problems. Our latest architecture integrates several learning and action-selection mechanisms into one much more complex system. The main advantages of such an agent are in its total autonomy, the ability to gain all information from a surrounding environment. Also, the ability to efficiently decompose potentially huge decision space into a hierarchy of smaller spaces enables the agent to successfully learn and "live" also in very complex domains. Unsupervised learning is triggered mainly by agent's predefined physiology and intentions which are autonomously created during his life. Not only theoretical background to creation of our agents is presented. We describe our latest architectures of autonomous creatures, too. Several experiments which were concluded in order to validate the expected abilities of our agents are also presented here. One of main contributions of our research is in proposing an original hybrid domain independent hierarchical planner. This planner combines classical planning system with hierarchical reinforcement learning. The ability to accommodate changing ideas about causality allows the creature to exist in and adapt to a dynamic world.

1 Introduction

It has been shown on many examples that human can inspire himself in the nature, while trying to find out how to automatically solve some complex problems. Because of the fact that the nature had hundreds of thousand years to "invent" these approaches, the resulting solutions to given problem can be

[1] Czech Technical University in Prague, Faculty of Electrical Engineering, Technická 2, 166 27 Praha 6 – Dejvice, E-mails: nahodil@fel.cvut.cz, vitkujar@fel.cvut.cz

surprisingly simple and efficient together. We can mention the ant colonies as an example. This approach utilizes simple interactions between many small agents in order to solve complex problems. Systems inspired by this concept are now used to solve complex tasks, as is *Travelling Salesman Problem* (TSP) or some optimization tasks [1,2]. But mans strive to go further, to copy the Nature in her ability to create intelligent beings with similar qualities to the mankind reaches far into our history. Despite the many attempts, this old aim still has not been reached. The autonomous functioning of a robot in real environment meets many challenges. The control architecture must give the robot ability to react timely with respect to the local disturbances and uncertainties, while adapting to more persistent changes in environmental conditions and task requirements [3]. This adaptation occurs in such a way that the robot optimizes its behavior so as to minimize required effort and maximize its profit (i.e. gather maximum environment resources while consume minimum energy). The result is often called as rational behavior. Learning and adaptation should occur without outside intervention – unsupervised learning, which means that the agent itself must decide what, is good and what is not. The inherent problem in this area of research is that considerable work effort is required to equip robots with adequate means for sensing (sensors) and actuation (effectors). Recognition and transformation of data in noisy and voluminous environment poses an obstacle in the robot design. Thus, to study control architecture the research moved from real environments to virtual ones. The research of behavior no longer needs a physical robot; the virtual representation of a robot can provide the same level of embolisms as real one. For these virtual robots in analogy with the *Multi-Agent Systems* (MAS), the term "agent" started to be used [4]. These two fundamentally different approaches merge by the selection of common name "agent". MAS originally used the top-down approach, focused on planning, problem solving which we can consider as a high level function of some animals and also humans. On the contrary the bottom-up approach used in robotics and also by nature in the simple organisms is focused on reactions to the stimuli. This approach uses emergence as a tool for creating more complex and complicated behavior by chaining the most basic reactions together. By joining these two approaches together with a meaningful trade-off between theirs pros and cons proved to be a very interesting option. This option is called hybrid architecture.

Also, according to our belief, the single kind of problem representation or approach to solving the problem is almost never sufficient. Each action made by living animals is a consequence of superposition of many different motivations, needs, emotions, intentions etc. We simulate this by connecting several of the decision and control blocks. Each of these blocks consists of one

of the well-known and widely used systems, such as reinforcement learning or a planner. As a result, instead of attempting to implement all of the decision making and learning by e.g. neural networks and expecting the emergence of high-level behaviors, we connect the different blocks in such a way that the resulting system suppresses the weaknesses of particular subsystems and exploits benefits of each more efficiently. Unfortunately, the interconnection of these approaches together creates considerably big gap between their main basic characteristics: their functionalities, representation of a problem etc. We use hierarchical concepts in order to blur this gap in our research.

We build on the concept of *Reinforcement Learning* (RL) in our work. It provides the agent with capabilities to learn and adapt its behavior in unsupervised manner. Compared to classical RL, our architecture is able to define the rewards itself. This means that the agent is able to distinguish what is good and what is not, based on active changes in his physiological system. Traditional RL methods suffer from the course of dimensionality. Present attempts to beat this course lay especially in a hierarchical decomposition of decision space. But these hierarchies of actions have to be predefined a priory. Our architecture is able to discover and build this hierarchy of actions by itself, based on consequences of its behavior in some unknown environment. Agent basically learns how to act in order to maintain his physiological variables in the desired (safe) area. This means that agent's physiology is a source of motivation to execute particular actions. It forms a feedback loop which stabilizes the agent's physiology. This concept is similar to homeostasis in biology.

2 Theoretical Foundation for Designed Agent Hybrid Architecture

The autonomous operation of intelligent robots – artificial creatures in real environment poses many challenges to their control architecture. The control architecture must give the resulting robot ability to be reactive with respect to local disturbances and uncertainties and adapt to more persistent changes in environmental conditions and task requirements. This adaptation occurs in such a way that the robot optimizes its behavior so as to minimize required effort and maximize its profit (i.e. gather maximum environment resources while consume minimum energy). Learning and adaptation shall occur without outside intervention – unsupervised learning, which means that the agent itself must decide what, is "good" and what is not. Unfortunately, inherent problem of this research is that considerable work effort is required to equip robots with

adequate means for sensing, actuation and transformation of noisy and voluminous data to internal representations. Therefore, to study control architecture, conceptually the most complex and interesting part, the research moves from real environments to virtual environments. The term "agent" replaces the term "robot" or "artificial creature".

As it can be seen in the Nature on almost all types of organisms: successful life in our highly complex and dynamic environment requires fusion of more than just one selected approach. This is one of the main reasons, why we have focused on hybrid agent architectures, where a several number of different methods of problem solving and learning are connected together. The most important theory about the main principles will be described here.

2.1 Biological Reinforcements

This section describes biological reinforcements and their correspondence to reinforcements (or drives) that are used for unsupervised learning. The concept of reinforcements used in this work is inspired by work of J.E. Mazur [Mazur, 2006]. The following reinforcement types exist:

- **Positive reinforcement:** Presentation of a stimulus ("pleasant") that increases the probability of behavior. (e.g., praise for task completed).
- **Positive punishment:** Presentation of a stimulus ("aversive") that decreases the probability of behavior. (e.g., blister on finger for raking leaves without gloves).
- **Negative reinforcement:** Removal of a stimulus ("aversive") that increases the probability of behavior. (e.g., removal of loud party next door after complaining to neighbor.).
- **Negative punishment:** Removal of a stimulus ("pleasant") that decreases the probability of behavior. (e.g., Loss of phone privileges after staying out late).

The continued effectiveness depends on the continued presentation of both types of stimuli (pleasant or aversive) and in both cases a Response-Consequence contingency is important for maximal effectiveness.

Selected factors influencing the effectiveness of (positive) punishment [Mazur, 2006]:

- *Immediacy of Punishment:* To be effect, punishment must be delivered as quickly as possible following the response to be punished.
- *Schedule of Punishment:* Intermittent schedules of punishment are less effective than continuous (regular) schedules.

- *Motivation to Respond:* In general, the stronger the motivation (for whatever reason), the less effective a given level of punishment will be.
- *Make Alternate Behaviors Available:* Punishment is more effective if there are alternative responses that maintain their current level of positive reinforcement. This includes the case, where they can escape from the punishment situation entirely to obtain reinforcement elsewhere.

		Direction of Change in Behavior	
		Increase	**Decrease**
Method of Applying Stimulus after Behavior Occurs	**Present It**	Positive Reinforcement Behavior: UP	Positive Punishment Behavior: DOWN
	Remove It	Negative Reinforcement Behavior: UP (Avoidance or Escape)	Negative Punishment Behavior: DOWN (Omission or Time-out)
		Type of Stimulus	
		Pleasant	**Aversive**
Method of Applying Stimulus after Behavior Occurs	**Present It**	Positive Reinforcement Behavior: UP	Positive Punishment Behavior: DOWN
	Remove It	Negative Punishment Behavior: DOWN (Omission or Time-out)	Negative Reinforcement Behavior: UP (Avoidance or Escape)

Tab. 1: James Mazur's Matrix [10]

Disadvantages of Using (positive) punishment: It can lead to undesirable "side effects":

- *Emotional side effects:* Fear and anger may disrupt learning.
- *Suppression:* It may generalize to other behaviors besides the one being punished.
- *Partial punishment:* It is not effective, thus, continual monitoring is required by the behavior modifier.
- *Transfer:* Punishment in one situation may lead to aggression in that others.

When Punishment Usually Fails:

- The fact that it is delivered on an intermittent or "partial" schedule
- Punishment in the outside the laboratory in the *real world,* is often delayed.

- Initial punishment attempts are often mild and only escalate with subsequent attempts.
- Impending punishment is often signaled and, therefore, can be effectively avoided.

Note: Temporal contiguity (i.e., closeness in time) between the cause and the response on it is important to the punishment effect.

2.2 Interaction of Internal and External Stimulation

According to the world famous Lorenz's theory [19] the type of Fixed Action Patterns (FAPs) exhibited by an animal is a function of:

- The amount of accumulated action specific energy (internal stimulation) and
- The sign stimuli (external stimulation) to which the animal is exposed.

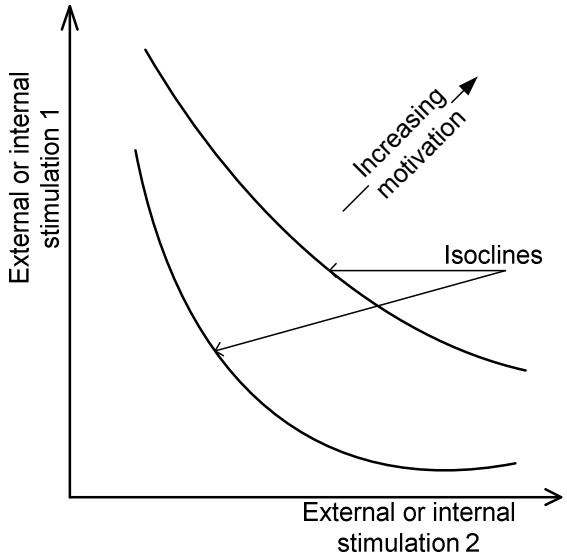


Figure 1. This graph shows interaction of two stimulations – external or internal. Motivation isoclines determine level of the same behavior motivation

Baerends and his colleagues [18] have provided an elegant demonstration of this principle. Male guppies exhibit several Fixed Action Patterns in their courtship behavior:

- sigmoid posture — a high intensity behavior
- sigmoid intention — a medium intensity behavior
- posturing — a low intensity behavior

The external markings of a male guppy vary with its readiness to show courtship. In terms of Lorenz's model [19], the external markings are an indication of the level of action specific energy for courtship. The stimulus value (of the female) increases with its size. Baerends conducted experiments in which males with different external markings were exposed to females of various sizes. The results of these tests are shown below and indicate that for each pattern of male courtship behavior, the size of the female needed to elicit the pattern was less the greater the readiness of the male to court.

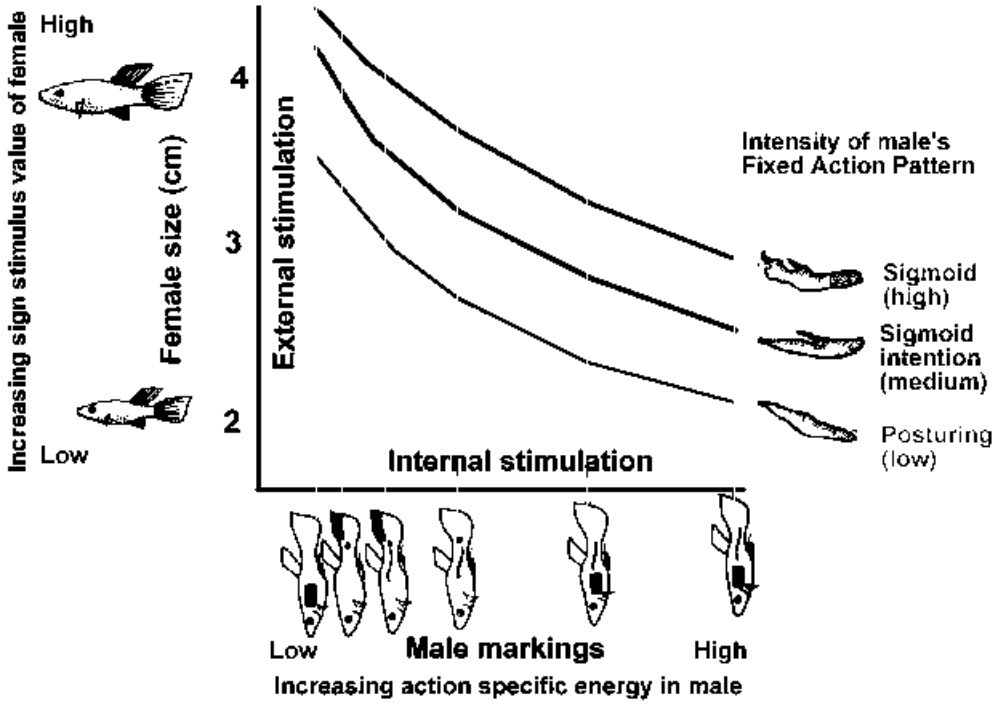


Figure 2. Courtship behavior of male guppies. The strength of external stimulation (measured by the size of the female) and of the male's internal state (measured by the color pattern of the male) jointly determine the strength of the courtship tendency (measured by the typical courtship postures S, *Si,* and *P*). Each isocline joins points of equal courtship tendency. The hyperbolic isoclines suggest a multiplicative relationship between internal and external stimulation. (After Baerends [18])

The diagram shows the influence of the strength of external stimulation (measured by the size of the female) and the internal state (measured by the color pattern of the male) in determining the courtship behavior of male guppies. Each curve represents the combination of external stimulus and internal state that produces the sigmoid courtship patterns of increasing intensity (After Baerends[18]).

Identifying relevant stimuli: A response is said to be under the control of a particular stimulus if the response is altered by changes in the (intensity, duration, frequency, quality) of that stimulus. Thus, differential responding to various stimulus features reflects stimulus control by those features.

Identifying relevant stimulus features: Stimuli vary along a number of dimensions, such as color, location, and size. These features may independently or in combination, control behavior. To determine what dimensions, and even what particular segments of stimulus dimension control behavior, we also test for "differential responding" as the dimension is varied systematically.

Measurement of the degree of stimulus control: Generalization of stimulus control occurs as dimensions of the stimulus are systematically varied. The segment or element of the dimension with the greatest stimulus control will produce the strongest learned response. Elements that vary from this stimulus (along any number of dimensions) will produce less responding. The variations in responding as the dimension is varied around the element with the greatest stimulus control, yields a function or curve, known as the *stimulus generalization gradient*. Gradients that are relatively more "steep" (e.g., when the same elements are being tested with animals given two different training experiences) indicate relatively greater stimulus control. The shape of the generalization gradient is determined by the differential reinforcement used in the S+/S- type of discrimination training. Without such training, responses to stimuli similar to the training CS or S^D will generalize more broadly than if such training is carried out.

Stimulus generalization and stimulus discrimination: Generalization and discrimination may be complementary phenomena, as Domjan [20] claims; however, they are not opposites. Generalization is an "innate" response of organisms. Discrimination – that is responding differentially to different stimulus features – is a result of specific learning experience.

Stimulus Intensity or Salience: There really appears to be two different influences here. Intensity is clearly important in determining stimulus control or relative stimulus control as in overshadowing experiments. Salience, on the other hand, is not determined solely by intensity. The question of what makes some stimuli more "salient" than others, when one is not obviously more intense than the other is unclear. Species differences certainly would seem to influence the relative salience of different types of stimuli. In addition, it seems likely that salience may depend on prior experience.

Interdimensional versus Intradimensional Discriminations: The distinction being made in this section is between discriminations that involve stimuli that vary only along one dimension (intra = within dimension variation) and stimuli

that vary along more than one dimension, and thus, might be discriminated based on these various dimensions (inter – between dimension variation).

Example: Humans and other animals use their ability to discriminate and to generalize in order to respond appropriately within particular environments or situations. "Perceptual concepts" thus represent groups of stimuli that are similar in some ways (thus we generalize among them), such as "flowers," while at the same time we are able to discriminate among these groups and others (say, birds or vegetables), or we can discriminate among individuals within these groups (lilies vs. roses).

2.3 Behavior Structure

Behavior is heterogeneous; it has structure, which limits its shape-ability. Understanding how learning occurs (i.e., what types of changes are possible and what types are not) requires the trainer to be aware of the unlearned behavior of the subject of the training because unlearned and learned behavior interact in complex ways.

Appetite and consummatory behavior:

- *Appetitive behaviors* are those parts of the *Fixed-Action Patterns* (FAP) that bring the animal into contact with the object of its motivation. The object of the motivation is often called the *goal object* or simply the *goal*.
- *Consummatory behaviors* are those parts of the FAP that allow the animal to "consume" the goal object.

Behaviors that do not require learning are distinguished from learned behaviors because:

- they appear fully functional the first time they are performed
- once initiated, they run to completion
- they are not modified by experience
- they are highly stereotyped (show little variation among individuals)

The distinction between learned and innate behaviors is not always clear, even when it is:

- the behavioral repertoire of most animals includes both innate and learned behaviors
- both require complex neural networks that still develop via complex interactions of genes and environment

- innate (involuntary) behaviors have no motive while voluntary behaviors are motivated
- typical metrics are duration, frequency, and intensity

Reflex:

- Reflex is the simplest type of *involuntary*, unlearned behavior.
- Reflex is a response to an eliciting stimulus; thus the response is said to be *elicited behavior* as opposed to emitted behavior.
- Reflexes are short and cannot be interrupted.

Fixed-action patterns (FAPs):

- FAPs are complex and systematic strategies (elicited *response patterns*) that animals use to reach important species-typical goals.
- FAPs are also known as *species-typical behaviors***.** Nest-building is a good example, since various unique behavioral methods of building are typical of various bird species.
- FAPs are discrete and recognizable
- FAPs are fewer stimuli bound than reflexes. Their intensity is not a simple function of the stimulus and their timing is not so closely determined by the stimulus.
- FAPs can interrupt each other, and in the normal course of events they do so.

  Example: A Siamese fighting fish that is eating will stop eating and display if a rival appears. This property has obvious adaptive value, and a fish would be very inflexible if behaviors switched on for a fixed time regardless of changes in the stimulus situation.
- FAPs are released by "sign stimuli", usually particular components of the stimulus provided by a rival, a potential mate or others.

After Discharge:

- After discharge is a typical phenomenon of fixed action patterns.
- The threat displays of Betta persist after the stimulus disappears. If a displaying fish suddenly has no opponent, perhaps it has been removed by the experimenter, the fish's display will persist, a phenomena called *after discharge*. After discharge occurs on two timescales .The discrete fixed-action patterns after discharge lasts for a second or two, but there is a much longer aggressive after discharge with an alteration of threats.

Sign stimulus (releaser):

- Sign stimuli are the critical components of the entire stimulus complex that are necessary and sufficient for elicitation of the FAP.
- Sign stimuli are activating the FAPs
- Sign stimulus consists of a few simple cues reliably associated with conditions/situations in which the FAP will be adaptive — e.g., high-contrast, moving red spots are, for herring gull chicks, reliably associated with their parents' beaks
- Sign stimuli are the result of another evolutionary „stimulus filtering" mechanism — only this one less flexible than associative learning

Innate Releasing Mechanism (IRM):

- IRM is internal mechanism which — when the sign stimulus is perceived — triggers the FAP to run to completion, even if stimulus removed.
- IRM can range from simple — e.g., inter-neurons in spinal reflex arcs — to complex term really represent, for plenty of behaviors, a "black box".

Supernormal Releaser:

- Supernormal Releaser is an exaggerated sign stimuli leading to exaggerated responses.

FAP examples:

The existence of FAP's can be/has been exploited evolutionarily by "code breakers"- organisms that mimic sign stimuli to produce FAP's to their own advantage

- nest parasites like brown-headed cowbirds
    - sign stimulus for feeding = gape, often with yellow mouth, and calls
    - because cowbirds are generally larger than hosts, chick's gape and call constitutes a supernormal releaser of feeding behavior by host parents
- rove beetle lays eggs in ant nests
    - larvae mimic pheromone that releases FAP in ants causing them to move larvae into brood chamber, where they eat ant eggs and larvae

- mimic food-begging behavior: tap worker ant's mandibles, releasing food regurgitation behavior by ants
- mimics of cleaner wrasse (also demonstrates complexity of interactions)
  - cleaner wrasse are fish that clean ectoparasites off other fish
  - set up "cleaning stations" on coral reefs; other fish learn where these are
  - when fish approaches, wrasse performs stereotyped swimming display – sign stimulus
  - in response, fish adopts head-down or head-up posture with mouths and gills open = FAP
  - that FAP, in turn, releases cleaning behavior by the wrasse
  - mimics mimic the swimming display; when fish open mouths and gills, mimics bite of chunks of gill

2.4 Reinforcement Learning

The key and basic principle is the RL, learning method inspired in behaviorist psychology, where an agent learns, which actions should take in the given state in order to maximize its future reward from his environment. The basic idea is the same with a dynamic programming, it is very general approach and the only main disadvantage is fact that an environment formulated as a *Markov Decision Process* (MDP) is required [5]. Interaction with the MDP environment means that each discrete time step t an agent perceives the finite set of states $|S|$ and is able to execute finite set of actions $|A|$. After executing the selected action $u_t$in the state x_t, the environment responds with a reward or punishment $r(x_t,u_t)$ and a new state$x_{t+1}=T(x_t,u_t)$ is generated. The next-state function T and the reinforcement r function are not known to the agent. The important property of MDP is that the transition function T is based only on the actual state and executed action.

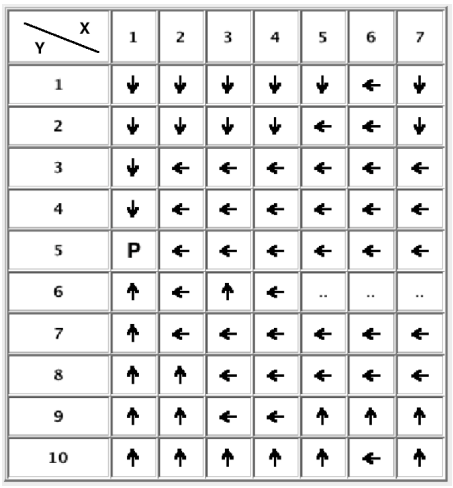


Figure 3. An example of learned behavior which controls the lights is shown here. The table represents primitive actions with the highest utility based on the agent's position in the map. The agent approaches towards the switch and executes action press (denoted by Phere) on the correct position. The successful execution of this behavior switches the value of variable lights-state between two possible states: on/off.

The goal of RL is to choose actions in response to states so that the reinforcement is maximized, this means that an agent is learning policy: a mapping from states to actions. There are several possible ways to implement a learning process; here was chosen a Q-learning. In this form of RL an agent learns to assign values to state-action pairs a Q-value function, the value of this function is sum of all future events. While immediate rewards are more important, hare is used discounted cumulative reinforcement, where future reinforcements are weighted by value $\gamma \in <0,1>$. The equation (1) represents the optimal Q-value function.

$$Q^*(x_t, u_t) = r(x_t, u_t) + \gamma \max_{u_{t+1}} Q^*(x_{t+1}, u_{t+1}) \quad (1)$$

At each step, the agent executes one action (selected based on the discounted Q-value function) receives reinforcement and updates Q-value of a given state-action pair in the table according to the off policy *Temporal Difference* (TD) control - equation (2), where $\alpha \in <0,1>$ is the learning rate.

$$Q(x_t, u_t) \leftarrow Q(x_t, u_t) + \alpha[r(x_t, u_t) + \gamma Q(x_{t+1}, u_{t+1}) - Q(x_t, u_t)] \quad (2)$$

In order to get a good trade-off between exploration and exploitation, an action selection mechanism uses some kind of randomization, instead of pure greedy

method. A system that implements this entire mechanism will be called a *return predictor*.

2.5 Hierarchical Reinforcement Learning

The classical RL approach has one disadvantage: size of look-up table (matrix) for storing Q-values grows very fast with an environment complexity. This means that in slightly more complex environment the Q-value matrix can have too many dimensions and the learning convergence can be very slow. In order to beat the course of dimensionality, *Hierarchical RL* (HRL) was introduced. We can define *Decision space* (D) as some defined subset of all possible actions and environment states, over this decision space can operate one return predictor. This decision space can be then seen as an *abstract action*. The main idea of hierarchical RL is very simple: in case of the classical "flat" Q-learning algorithm an agent selects among primitive (one-step) actions. Compared to this, in the hierarchical RL the return predictor can select among primitive and abstract actions (decision spaces). The HRL uses *Semi Markov Decision Process* (SMDP), where a waiting time for the next time step *t+1* is random variable.

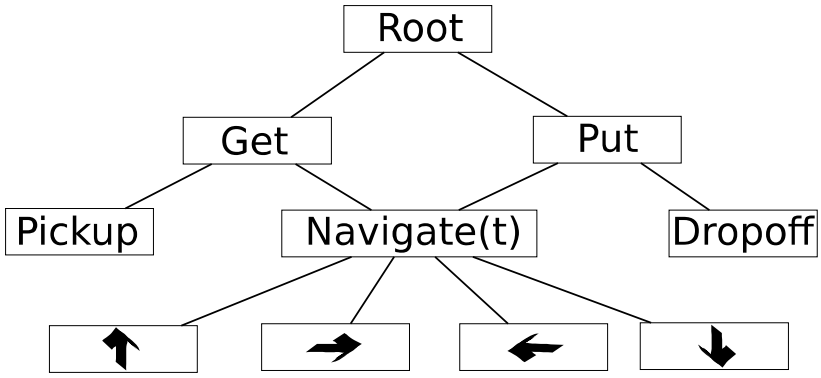


Figure 4. Example of hierarchical task decomposition for well-known taxi problem (6). Hierarchy of abstract actions builds on consecutively more and more primitive actions.

In our approach is the MAXQ value function decomposition used [6], where the received reward can be distributed into a hierarchy of decision spaces D_i using factorization function $\rho(\bar{r}, D_i)$, where $\bar{r}$ is the reward generally from the composite behavior. The parameter τ represents positive duration of action, then the Q-learning update formula for decision space D_i is in equation (\ref{eq:3}).

$$Q_i(x_t, u_t) \leftarrow Q_i(x_t, u_t) + \alpha[\rho(\bar{r}, D_i) + \gamma^\tau Q_i(x_{t+1}, u_{t+1}) - Q_i(x_t, u_t)] \quad (3)$$

Our proposed approach is based on an architecture called "*Hierarchy, Abstraction, Reinforcements, and Motivations Agent Architecture"* (HARM) [7]. Therefore is used motivation $m(D_i)$ which defines "how much" the agent wants to execute particular action (corresponding to a decision space D_i). The resulting *utility* of action (for a decision space on the top of the hierarchy) is defined in the following equation:

$$\varphi_{D_i}(s, a) = m(D_i) Q_i(s, a). \quad (4)$$

Utilities for the rest of decision spaces in a hierarchy are composed from its own utility and utilities of all parent decision spaces through connection of strength $c_{D_i}^{D_j}(s, a)$ as seen in the equation (5).

$$\varphi_{D_i}(s, a) = m(D_i) Q_i(s, a) \sum_{j \in pars(D_i)} c_{D_i}^{D_j}(s, a)\, \varphi_{D_j}(s, a) \quad (5)$$

Because of this approach, the motivation to execute a particular behavior (action) can spread through the connection function $c_{D_i}^{D_j}(s, a)$ from the top of a hierarchy towards primitive actions. This means that a selection of concrete primitive action to be executed emerges from various motivations, conditions and dependencies in a whole hierarchy. When a reinforcement/punishment is obtained, this information travels in the opposite direction, from the primitive actions towards the more complex decision spaces on the top of the hierarchy through the factorization function $\rho(\bar{r}, D_i)$.

2.6 Planning System

For implementation of planning system it was used the world-wide known language, called *Stanford Research Institute Problem Solver* (STRIPS). It is formally represented as a quadruple. The *P* is the set of conditions expressed by propositional variables describing the world state, *I* denote the description of initial state and *G* is description of properties which are fulfilled in a goal state(s). *O* is the set of operators - actions; each operator consists of the quadruple. The elements α_s and β_s describe the constraints when the action can be applied, that is: describe which conditions must be true and which false in

the given situation. The elements γ_s and δ_s describe action effects after its application, that is: which propositional variables will become true and which false. Roughly speaking, the current state of the world is described by a binary vector, where operators change values of bits on a specified position in a specified manner. The plan is a sequence of applicable operators that consecutively transform the description of initial state towards the state which fulfils the goal conditions.

As a typical planner, STRIPS requires on its input three main things: description of the current state, description of a goal state and a set of possible actions. Our latest architecture, presented in [9] is able to automatically infer this information from the HARM action hierarchy. This process will be described later in more details.

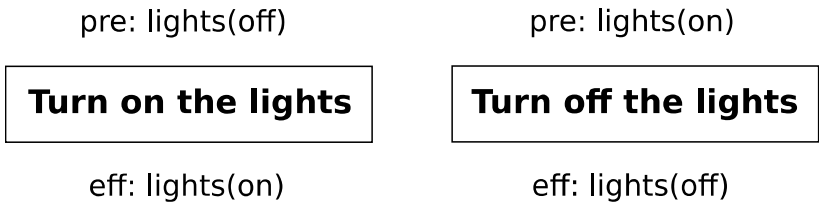


Figure 5. Simplified example of two primitive actions represented in the STRIPS language. From the preconditions it is obvious that lights cannot returned on if they are already turned on and vice versa.

3 Main Concepts Used in our Novel Approach

Dr. David Kadleček, in his Dissertation Thesis [7], presented HARM system, which is capable of creating such hierarchy of decision spaces autonomously, based on received reinforcements of various types [8]. Several of concepts are taken from this work and similar older research, while some of our novel methods were originally presented in [9]. The main principles, discovered by our research, will be briefly described in this chapter.

3.1 Physiology and Intentions - The Sources of Agent's Behavior

HARM architecture is inspired mainly in the fields of biology, ethology and control engineering. Such an agent has its own predefined physiology here. Agent's physiological state-space, represented by a dynamical system, contains set of agent's internal variables (see Fig.6). The physiological state-space

contains two important areas: limbo and a purgatory one. Limbo area represents the optimal conditions, if an agent is in this area, no motivation is generated. On the other hand, if an agent is in the purgatory area, an amount of produced motivation increases exponentially. If an agent actively moves some of his physiological variables towards the optimal conditions, reinforcement is received, if the movement is in another direction, towards the purgatory area, a punishment is received [8]. Later, this concept was augmented by the intentional state space. This space does not have the purgatory area and represents the set of agent's possible interests. The agent's physiology is predefined by the designer. Compared to this; the intentional state space is empty before the beginning of the simulation. The agent is able to autonomously add newly discovered variables in this state space. These intentions then motivates agent to learn corresponding newly discovered behavior - ability. During his life, agent continually learns how to connect his behavior in such manner, that he is able to respond to the motivations produced by these dynamical systems correctly. For example, an agent who contains the physiological variable called *water level* tries to learn drinking behavior. Because of this, the resulting system utilizes some kind of self-rewarding closed-loop mechanism and thus is able to learn autonomously.

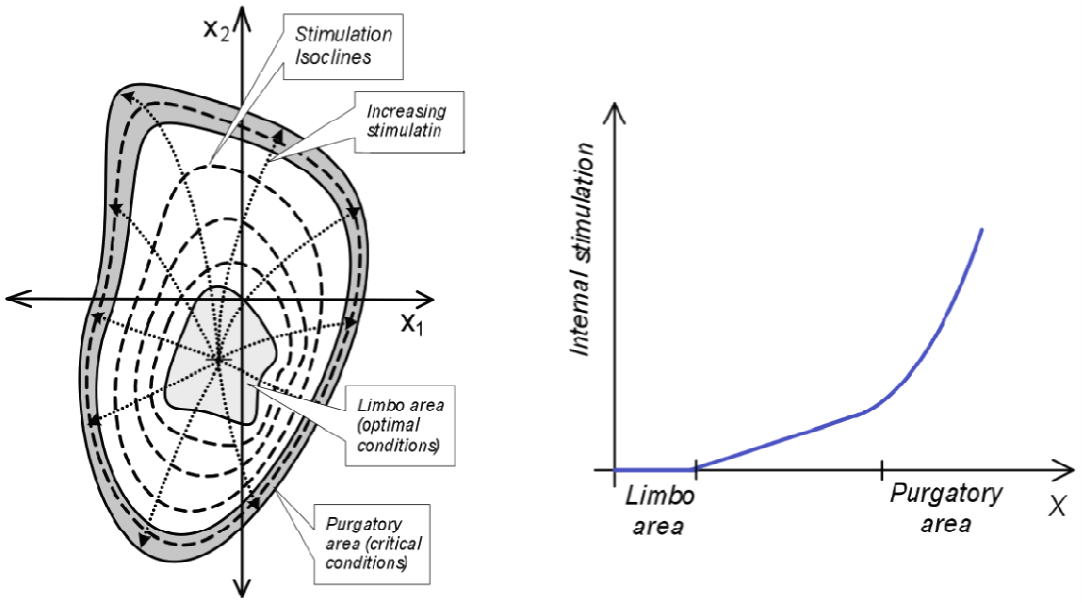


Figure 6. Physiological state space and amount of stimulation produced. Physiological space contains three regions: (i) limbo – animal's conditions are almost optimal, (ii) purgatory – animal's conditions are critical and (iii) area in between these two where the stimulations grow nearly linearly. Stimulations increase from the origin towards boundaries.

3.2 Autonomous Creation of Action Hierarchy

After receiving a reward or a punishment, new decision space D_i in a hierarchy of actions is created. This new decision space is then connected to a physiological (or intentional) variable through the motivation link $m(D_i)$. Because of this approach, an agent autonomously connects consequences of his behavior with own physiology and learns how to preserve homoeostasis. A set of variables and actions contained in particular decision spaces (and thus also the shape of action hierarchy) is maintained during the agent's life by using four main operations: sub-spacing, behavior associating, variable removing and variable promoting. This approach dramatically reduces the size of decision space which has to be searched by the learning algorithm. It speeds up the learning convergence and enables our agent to learn even in very complex domains.

3.3 From Reinforcement Learning towards the Planning

As a latest result of our research in the field of ALife, we have proposed a system that is capable of deliberative "thinking" over this autonomously created hierarchy of abstract actions (decision spaces). This gives the agent whole new dimension of abilities how to use this knowledge.
Compared to the RL, from our point of view, the planning is deliberative approach capable of solving complex tasks, but it requires accurate description of an environment. This requirement can cause problems even in relatively simple environments, where total number of possible states always grows too fast to be handled by a planner. This disadvantage is solved by hierarchical planners, for example *Hierarchical Task Network* (HTN), but these planners are domain dependent, or at least domain configurable. The main advantage of our approach is that our hierarchical planner can beat the course of dimensionality a well as other hierarchical planners, but moreover maintains its domain independence. In other words our planned is domain self-configurable: by using the autonomous creation of action hierarchy, it can adapt itself to a given domain.
In the Fig. 3 it can be seen an example of learned decision space represented by a 2D matrix, where each tale corresponds to a position of an agent in the map. Each primitive action (depicted on each tale) represents the learned action, which is the action with the highest Q-value. The agent discovered that by pressing the switch on the left side of the map the light can be switched on/off. It was identified as agent's ability to change some environment property and new intention to learn this behavior was created. The picture represents behavior for turning on/off the lights, which was learned through this

motivation. The decision space D_i (matrix of Q-values) contains agent's actual position and the variable causing the reinforcement is *lights-state*.

The basic idea is that in order to use this decision space as a primitive action, we need to consider only the "main" variable of a decision space, the variable that **change during the reinforcement**. The following example is depicted in the Fig. 12. In this case, where the decision space consists of three variables: agent's *X* and *Y* position and the *lights-state*, the "main" variable of the decision space is *lights-state*, to the planner will take into account only this variable. Now follows the description of how primitive actions in the STRIPS language are generated: the decision space was created in order to learn the behavior turn on/off the lights. Exactly this does the primitive action in the STRIPS language. In case of a binary variable, this decision space can be represented as two primitive actions in the STRIPS language. The vector describing the problem has one bit **Turn on the lights**, in this simple case only. The action contains precondition: *lights off*, and effect: *lights on*. The action **Turn off the lights** contains precondition: *lights on*, and effect: *lights off*.

The description of entire environment can be automatically generated in form of STRIPS language by use of this principle. The main advantage here (besides the domain independence) is the fact, that only those interesting and potentially important information are passed to the planner. The state description was reduced from 3 variables to one, in the previous example. A hierarchy of RL actions serves here as some kind of filter. This autonomous pre-processor filters information for the deliberative planner, which works over the hierarchy of actions.

4 Selected Simulations

We have concluded many simulations in order to verify anticipated capabilities our autonomous agents. The simulations tested various aspects of agents' behavior, for example learning how to solve complex tasks, ability to reuse the autonomously gained knowledge, ability to act in dynamical environment while using simulated sensors etc. In this section we would like to focus on two sample experiments which test agent's ability to autonomously create the hierarchy of actions and to learn behaviors in this hierarchy. The second sample experiment will focus on testing of the deliberative subsystem. This means that agent's knowledge will not be used only on the layer of RL, but the agent will be ordered to reach given goal, that is to autonomously create and execute plan. This plan will be based solely on the knowledge which was learned by the agent and stored in the hierarchy of RL actions.

4.1 The Treasure Problem - Autonomous Creation of Hierarchy

In this selected experiment, the agent's task is to get to the treasure locked behind the door. In order to open the door the agent has to put the stones onto the buttons in a specific order. Besides learning the task, the agent needs to drink and eat in order to survive. Before the commencement of a simulation, the agent's physiological state space is equipped with variables: water, food and special obligation variables motivating the agent to pick and drop the stones, to open the door etc. Also, the agent has the capability of the following primitive actions: *move* in four directions, *pick, drop, eat* and *drink*.

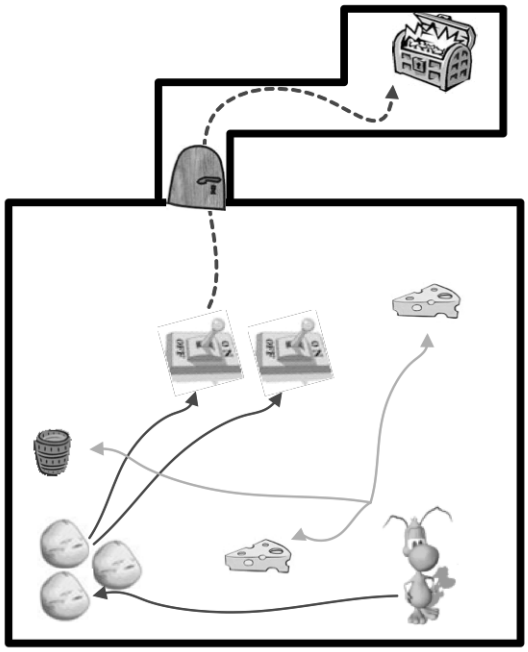


Figure 7. The treasure problem: agent has to reach the treasure. In order to open the door he must put the stones onto the switches in the specified order. Agent has to follow two physiological needs: hunger and thirst.

In the Fig. 7 there is a problem description, map of the simulation environment. The Fig. 8 depicts the resulting action hierarchy with connections of decision spaces to its own sources of motivations. We can see that an abstract action "*reaching the treasure*" can be decomposed to more primitive behaviors "*open the door*" and "*go to treasure*". The action "*open the door*" can be further decomposed into two subtasks "*pick stone*" and "*drop stone*". This autonomously created action hierarchy efficiently represents the problem

structure and the agent is able to reach the treasure and simultaneously eat or drink if necessary.

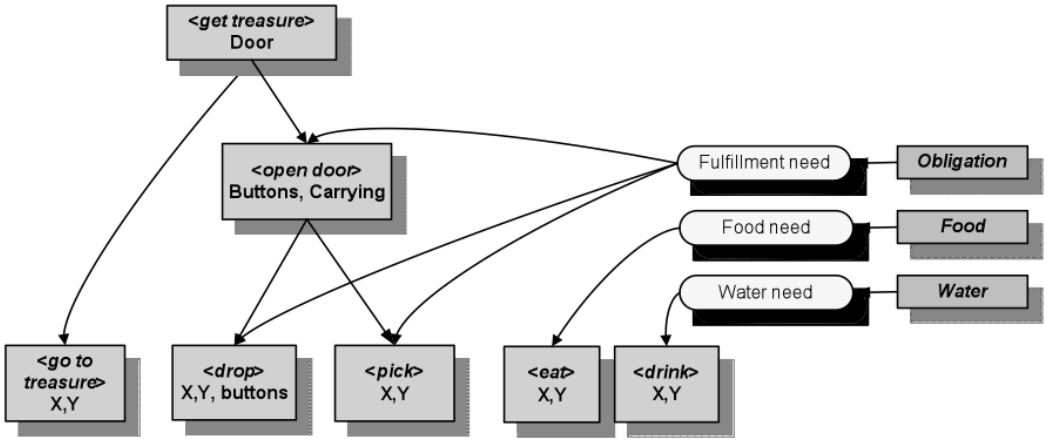


Figure 8. Autonomously created hierarchy of abstractions. These actions are connected through the motivation link m(Di) to the sources of motivation - agent's physiological variables.

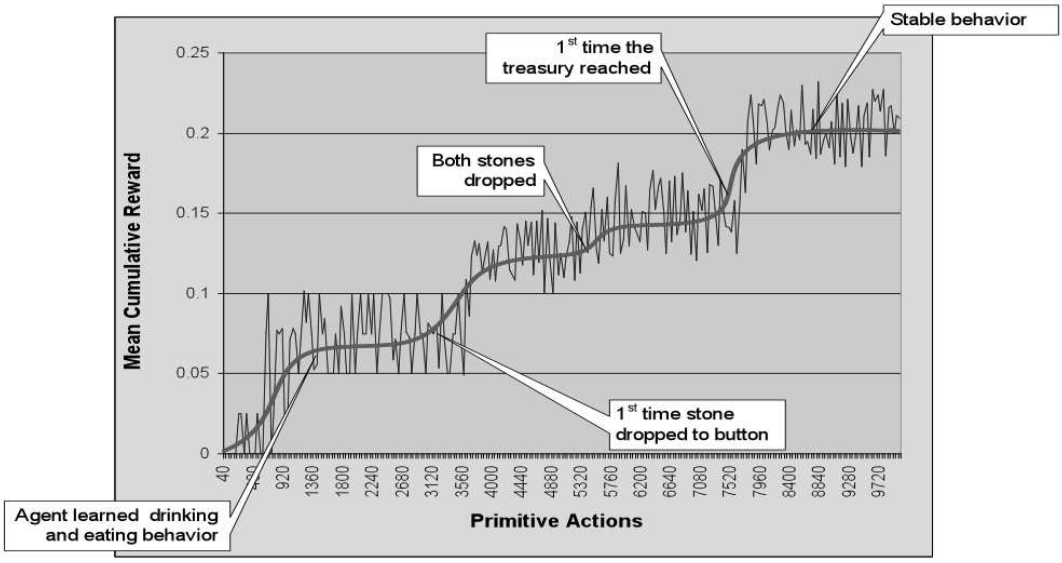


Figure 9. Example of a conversion of the RL decision space into set of primitive actions in the STRIPS language. Based on the "main variable" (there in force done), the agent was able to autonomously generate two primitive actions with their preconditions and effects. The same process was executed on the rest of actins, which are motivated from the agent's intentional state-space.

The Fig. 9 shows the course of agent's continuous learning by interaction with the environment. On the Y axis there is value of mean cumulative reward obtained by the agent from the environment, on the X axis is time in discrete steps. From the graph are clearly visible moments when the agent managed to discover new behavior. We can see that the hierarchy of actions was built consequently by the bottom-up approach: from the simpler behaviors towards the more complex strategy. Finally the agent learnt the stable behavior. This means that was able to successfully drink and eat when necessary. In a "free time" the agent was training the fulfilling the goal of the "Treasure problem".

4.2 The Lights and Doors –Simplified Example of Designed Hybrid Planning

The second experiment was concluded in order to test the agents ability to reuse an autonomously knowledge by the planning system, that is to get the set of abstract actions in a form of RL and autonomously create the world description (model) and a set of primitive actions. These (from the planners point of view) primitive actions can be then used for planning.
In this experiment, the user needs to pass the hallway in order to reach the goal position. The requirements are that all doors on the path are opened and the lights are turned on. The only user's a priori knowledge is that these systems can be controlled only from unknown and unreachable part of the map. So our agent is sent to find out how these systems can be controlled. The agent learns how to survive simultaneously, so after some time, the agent is physically there and able to fulfill relatively complex tasks, as, for example, "*enable passing through the hallway*", which is composed of subtasks: "*turn on the lights*", "*open the door1*" and "*open the door2*". The desired state of the map is depicted in the Fig. 10, however, before the task is specified to the agent, the both doors are closed and lights are turned off. The user does not have any prior knowledge about the problem, his only knowledge is that the controls to the necessary properties are somewhere in the unknown sector of the map (left part). Our agent is sent to autonomously learn these principles, while he has only two physiological needs predefined and a set of the following primitive actions: *move* in four directions, *eat, drink* and *press buttons*.

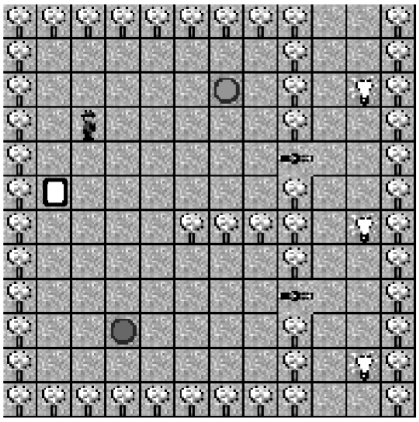


Figure 10. Map of the environment containing one agent, one lights-switch (on the left), source of water (down), source of food (up) and two buttons controlling doors (things in the wall). The hallway on the right contains lights and two doors between them that can be opened/closed.

After some time into the simulation, the agent was able to successfully identify and learn all five possibilities how to interact with the environment. The Fig.11 shows the resulting action hierarchy.

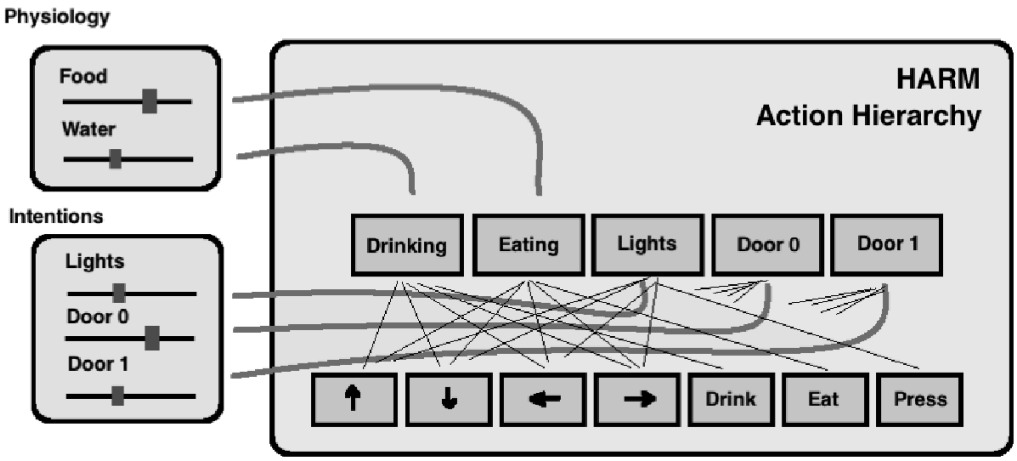


Figure 11. The resulting hierarchy of abstract actions. Drinking and eating behaviors are motivated by the agent's physiology; remaining actions are motivated by the agent's autonomously created intentions. The maximum level of action abstraction is one; therefore each abstract action is composed only of selected subset of primitive actions.

It is apparent that during his existence, the agent autonomously creates three intentions motivating him to learn how to control lights and both doors. The agent is now able to successfully drink, to eat, to switch the lights and to open/close both doors. We can see that the eating and drinking behaviors are

motivated from the agents predefined physiology, while the rest of actions agent discovered by himself. For each of these actions, new intentional variable was added to the intentional state space. This means that after discovering his new ability, the agent started to intend to try this new behavior again and again.

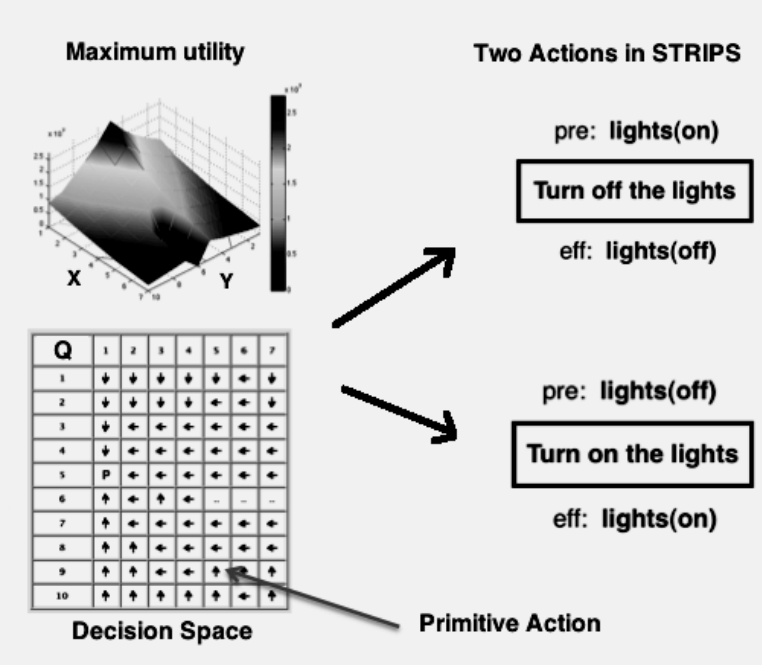


Figure 12. Example of a conversion of the RL decision space into set of primitive actions in the STRIPS language. Based on the "main variable" (there in force done), the agent was able to autonomously generate two primitive actions with their preconditions and effects. The same process was executed on the rest of actins, which are motivated from the agent's intentional state-space.

The Fig. 12 depicts the principle of how the abstract action (composite behavior) can be automatically translated into a set of primitive actions in the STRIPS language. So far we have focused only on the actions motivated by the agent's intentional state space, where a corresponding variable has only two states. The behavior "*switch the lights*" is represented as two primitive actions: "*turn on the lights"* and "*turn of the lights"*, with the corresponding preconditions and effects.

At the later stage of the experiment, the agent has a sufficient knowledge to fulfill the given task for the user. The user describes the goal state (lights on, doors opened), the agent creates the plan and executes it. The resulting plan is composed of sequence of actions: "*open the door0*", "*open the door1*" and "*turn*

on the lights". The principle of plan execution is as follows: The planner sets intention to execute the given action to the maximum. The planner waits until the intention falls towards zero, which means that given action was successfully executed. This is executed consecutively for all actions in the plan until the goal state is reached.

4.3 Benefits of our System Compared to the other Approaches

In this part we would like to mention why we believe that this presented approach has some considerable advantages compared to the other similar architectures.

The planner requires simple and clear description of a given problem. The main benefit of this approach is in a combination of hierarchical reinforcement learning subsystem. The planning does not have to deal with all information about any surrounding environment here. The hierarchy of abstract actions serves as some kind of interface between outer complicated world and the planner. The augmented HARM system in our agent autonomously finds and stores just that important information from the environment. The information (e.g. variables) which is not interesting for our agent, likely they are also not important and therefore are ignored.

Pioneering hierarchical planning systems were domain dependent, this means that the user had to specify the hierarchy of actions (tasks) by hand based on the knowledge about particular problem [11, 12]. Nowadays hierarchical planners, as is e.g. S*imple Hierarchical Ordered Planner* (SHOP) [13], are domain configurable. It means that these planners need a domain description on its input. This main disadvantage has been also studied. There are some systems that are able to partially learn the knowledge about given domain needed by these hierarchical planners [14]. Several similar hybrid approaches combining planning system and RL were also found, but these are also at least partially domain dependent [15, 16]. Our approach can be said as a *domain self-configurable*, because no a-priory knowledge is needed before the beginning of the simulation. Our agent discovers everything necessary itself by interaction with the environment.

The other main benefit of this concept is in the fact that intention to execute particular action increases the action importance *in context* of the other actions. The resulting of primitive action executed by the agent is still a result of various preferences which affect the action selection mechanism formed by the hierarchy of return predictors. This means that even during the execution of plan, the agent successfully fuses more reactive and more deliberative control into one decision making system. In other words, the agent is able to take into

account both, the differential (immediate) and integral (future long-term) part of reward [17]. It means that even during the plan execution the agent is still able to prefer much more important things at the same moment, e.g. run away from the predator. In the second experiment, it was observed that the agent autonomously interrupted execution of plan in order to drink or eat. This feature is very important while operating in real, dynamically changing environment: to be able to solve complex tasks, but still be able to take into the account different needs.

5 Conclusion

Inspired by recent work in ethology and animal training, we integrate representations for time and rate into a behavior-based architecture for autonomous virtual creatures. The resulting computational model of affect and action selection allows creatures to discover and refine their understanding of apparent temporal causality relationships which may or may not involve self-action. The fundamental action selection choice that a creature must make in order to satisfy its internal needs is whether to explore, react or exploit. In this architecture, that choice is informed by an understanding of apparent temporal causality, the representation for which is integrated into the representation for action.

The ability to accommodate changing ideas about causality allows the creature to exist in and adapt to a dynamic world. Not only is such a model suitable for computational systems, but its derivation from biological models suggests that it may also be useful for gaining a new perspective on learning in biological systems. The implementation of a complete character built using this architecture is able to reproduce a variety of conditioning phenomena, as well as learn in real-time using a training technique used with live animals.

As a future work can be seen the creation of more sophisticated algorithm which will be able to infer the precondition and effects for general problems, that includes considering variables with more than two states and also domains where effects of particular actions in the hierarchy could interfere.

Also, we would like to conclude some experiments in the real environment with some more complicated sensory data. The most of similar architectures which use some hierarchical structures (e.g. *Belief-Desire Architecture* (BDI) [12]) are domain dependent; this means that some form of domain description need to be specified before the start of the simulation.

We believe that the main advantage of the architecture, described by us here, is in the fact that it can adapt itself to a previously unknown environment, learn completely from the scratch and reuse this knowledge for deliberative problem solving.

Acknowledgement: This research has been funded by the Department. of Cybernetics, Faculty of Electrical Engineering, CTU - Czech Technical University in Prague under Project MSM6840770038.

Literature

[1] Dorigo, M., Gambardella, L.M., Ant Colony System: A cooperative learning approach to the travelling salesman problem. In: *Evolutionary Computation, IEEE,* 1997, pp.53-66.

[2] Dorigo, M., Maniezzo, V., Colorni, A., The Ant System: Optimization by a colony of cooperating agents, in *IEEE Transactions on Systems, Man, and Cybernetics-part B*, vol. 26, 1996, pp.29-41.

[3] Steels, L. and Brooks, R. *The Artificial Life Route to Artificial Intelligence: Building Situated Embodied Agents.* New Haven: Lawrence Erlbaum Assoc, 1994.

[4] Wooldridge, M. R., *An Introduction to Multi-Agent Systems*. John Wiley \& Sons, New York,2002.

[5] Bellman, R. *A*, *Markovian Decision Process*. IndianaUniv. Math. J. 6: 1957, pp. 679–684.

[6] Dietterich, T.G., Hierarchical Reinforcement Learning with the Max-Q Value Function Decomposition, in: *Journal of Artificial Intelligence Research 13*, 1998, pp. 227–303.

[7] Kadleček, D., *Motivation Driven Reinforcement Learning and Automatic Creation of Behavior Hierarchies*, PhD thesis supervised by Nahodil, P., CTU in Prague, FEE, Department of Cybernetics, 2008, Prague, pp. 134.

[8] Kadleček, D., Nahodil, P., New Hybrid Architecture in Artificial Life Simulation, In: *Proc. of 6th European Conf. of Artificial Life: Advances in Artificial Life*, (eds. Kelemen, Sosík) ECAL 2001, Prague, LNAI Nr. 2159, vol. 1, Springer Verlag, Berlin, 2001, pp. 143-146.

[9] Vítků, J., *An Artificial Creature Capable of Learning from Experience in Order to Fulfil More Complex Tasks,* Diploma Thesis supervised by Nahodil, P., CTU in Prague, FEE, Dept. of Cybernetics, Prague, 2011, pp. 123.

[10] Mazur, J. E. *Learning and Behavior* (Sixth Edition). Upper Saddle River, Prentice Hall, New York: 2006.

[11] Wilkins, D. E., *Hierarchical Planning: Definition and Implementation*, Technical Note 370. AI Centre, SRI Internat., 333 Ravenswood Ave, Menlo Park, CA 94025, 1985.

[12] Sardina, S., de Silva, L., Padgham, L. Hierarchical Planning in BDI Agent Programming Language: a Formal Approach, *AAMAS 06 Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, 2006.

[13] Nau, D., Cao, Y., Lotem, A., and Muñoz-Avila H., SHOP: Simple Hierarchical Ordered Planner. *In IJCAI-99*, pp. 968-973, 1999.

[14] Ilghami, O., Nau, D., Muñoz-Avila H., and AHA D. W., *CaMeL: Learning Methods for HTN Planning.* AIPS-02, 2002.

[15] Ryan M. and Pendrith, M., RL-TOPs: An Architecture for Modularity and Re-Use in Reinforcement Learning, *In Proceedings of the Fifteenth International Conference on Machine Learning*, San Francisco, CA, USA, 1998, pp. 481-487.
[16] ROSS, M., *Hierarchical Reinforcement Learning: A Hybrid Approach*, PhD Thesis, The University of New South Wales, School of Computer Science and Engineering, 2002.
[17] Svatoš, V., *Behavioral control of robots with evaluation of individual credit for a collective goal. (*in Czech *Behaviorální řízení robotů s hodnocením přínosu jednotlivce pro kolektivní cíl*). PhD Thesis supervised by Nahodil, P., CTU in Prague, FEE, Department of Cybernetics, 2008, Prague, pp. 123.
[18] Baerends GP, Brouwer R, Waterbolk HT, Ethological studies on Lebistes reticulatus (Peters). I. An analysis of the male courtship pattern. *Behavior 8, 1955,* pp.249–334.
[19] Lorenz, K. *The Foundations of Ethology* (in Czech *Základy etologie*). Academia, Praha, 1993.
[20] Domjan, M. (2005). Pavlovian conditioning: A Functional Perspective. *Annual Review of Psychology*, 56, 179-206.

Multiple Time Scales Recurrent Neural Network for Complex Action Acquisition: Model Enhancement with GPU-CUDA Processing

Martin PENIAK[1] and Angelo CANGELOSI[2]

Abstract. This chapter presents novel results of complex action learning experiments based on the use of extended multiple time-scales recurrent neural networks (MTRNN). The experiments were carried out with the iCub humanoid robot, as a model of the developmental learning of motor primitives as the basis of sensorimotor and linguistic compositionality. The model was implemented through the Aquila cognitive robotics toolkit, which supports the CUDA architecture and makes use of massively parallel GPUs (graphics processing units). The results show that the model was able to learn and successfully reproduce multiple behavioral sequences of actions in an object manipulation task scenario using large-scale MTRNNs. This forms the basis on ongoing experiments on action and language compositionality.

1 Introduction

Building artificial cognitive systems not only advances the current state of the art in the field of artificial intelligence but also provides us with insights into many different aspects of human capabilities, reveals problems and loopholes in our current models of the brain and cognition. These models often involve dealing with inputs from multiple sensory modalities, solving complex, highly parallel calculations and controlling the behavior of embodied systems. We have developed a complex neural network model that utilizes hundreds of NVIDIA GPU processors, which paved the way for novel experiments on

[1] The University of Plymouth, United Kingdom, E-mail: martin.peniak@plymouth.ac.uk
[2] The University of Plymouth, United Kingdom, E-mail: a.cangelosi@plymouth.ac.uk

action learning.

Humans are able to acquire much skilled behavior during their life-times. The learning of complex behaviors is achieved through a constant repetition of the same movements over and over, with certain components segmented into reusable elements known as motor primitives. These motor primitives are then flexibly reused and dynamically integrated into novel sequences of actions. Arbib proposed a schema theory that provides the theoretical foundations underlying this process [1]. The schema theory has been adopted in many studies, for example in [2]–[4].

For example, the action of lifting an object can be broken down into a combination of multiple motor primitives. Some motor primitives would be responsible for reaching the object, some for grasping it and some for lifting it. These primitives are represented in a general manner and should therefore be applicable to objects with different properties. This capacity is known as generalization, which also refers to the ability to acquire motor tasks by different ways. This means that the learning of new motor tasks can be done by using anybody effector, or simply by imagining the actual task itself (see for example [5]). In addition, one might want to reach for the object and throw it away, instead of lifting it up. Therefore these motor primitives need to be flexible in terms of their order within a particular action sequence. The amount of combinations of motor primitives grows exponentially with their number and the ability to exploit this repertoire of possible combinations of multiple motor primitives is known as compositionality. The hierarchically organized human motor control system is known to have the motor primitives implemented as low as at the spinal cord level whereas high-level planning and execution of motor actions takes place in the primary motor cortex (area M1). The human brain implements this hierarchy by exploitation of muscle synergies and parallel controllers. These have various degrees of complexity and sophistication that are able to address both the global aspects of the motor tasks as well as fine-tune control necessary for the tool use [6].

The existence of motor primitives and their recombination into sequences of actions is supported by the biological observations of both humans and animals. Sakai et al. conducted experiments in visuomotor sequential learning and demonstrated that his subjects spontaneously segmented motor sequences into elementary movements [7]. Thoroughman and Shadmehr showed that the complex dynamics of reaching motion is achieved by flexibly combining motor primitives [8]. dAvella et al. analyzed the data recorded from electromyographic activity from 19 shoulder and arm muscles and concluded that: *”the complex spatiotemporal characteristics of the muscles patterns for reaching were captured by the combinations of a small number of components, suggesting that the mechanisms involved in the generation of the muscle*

patterns exploit this low dimensionality to simplify control" ([9], p. 7791). Experiments conducted on animals are also consistent with these findings. For example, it has been shown that the electrical stimulation of primary motor and premotor cortex in monkeys triggers coordinated movements such as reaching and grasping [10]. Giszter et al. found that a frog's leg contains a finite number of modules organized as linearly combinable muscle synergies [11].

Several action learning models have been proposed that implement functional hierarchies via explicit hierarchical structure, as with the MOSAIC model [12] or the mixture of multiple Recurrent Neural Networks (RNN) expert systems [13]. In these models the motor primitives are represented through local low-level modules, whereas higher-level modules are in charge of recombining these primitives using extra mechanisms such as gate selection systems. These systems carry great potential benefits. For example, the learning of one module does not interfere with the learning of other modules. Moreover, with the adding of extra low-level modules, the number of acquirable motor primitives can increase as well. However, it has been demonstrated that the similarities between various sensorimotor sequences result in competition between the modules that represent them. This leads to a conflict between generalization and segmentation, since generalization requires the representation of motor primitives through many similar patterns present in the same module whereas different primitives need to be represented in different modules to achieve a good segmentation of sensorimotor patterns. Because of the conflict that arises when there is an overlap between different sensorimotor sequences, it is not possible to increase the number of motor primitives by simply adding extra low-level modules [14]. The learning of motor primitives (low-level modules) and sequences of these primitives (hi-level modules) need to be explicitly separated through subgoals [13], [15].

Yamashita and Tani [2] were inspired by the latest biological observations of the brain to develop a completely new model of action sequence learning known as Multiple Timescales Recurrent Neural Network (MTRNN). The MTRNN attempts to overcome the generalization-segmentation problem through the realization of functional hierarchy that is neither based on the separate modules nor on a structural hierarchy. Hierarchies are rather based on multiple time-scales of neural activities that are responsible for the process of motor skills acquisition and adaptation, as well as perceptual auditory differences between formant transition and syllable level [16]–[20].

This chapter presents novel results of complex action learning based on an MTRNN model embodied in the iCub humanoid robot. The model was implemented as part of Aquila cognitive robotics toolkit [21]. This allows accelerated MTRNN learning experiments through the CUDA architecture which makes use of massively parallel GPU devices that significantly

outperform standard CPU processors on parallel tasks. The new experiments use extended MTRNN models to train the iCub to acquire reusable motor primitives. These will be subsequently used in experiments on the simultaneous acquisition of motor and linguistic skills, and the exploitation of compositional structure in both sensorimotor and linguistic representations [22]. Specifically, the experiment was designed to test the capability of the MTRNN system to learn multiple sensorimotor sequences in an object manipulation scenario. There are three semantically different classes of actions that are expected to exhibit similar sensorimotor patterns (e.g. push or pull the block). The choice of these semantically similar behaviors was influenced by our experimental plan and will facilitate the investigation of the verb island hypothesis.

2 Method

2.1 iCub Humanoid Robot Platform

The iCub (www.icub.org) [23] is a small humanoid robot that is approximately 105cm high, weights around 20.3kg and its design was inspired by the embodied cognition hypothesis. This unique robotic platform with 53 degrees of freedom (12 for the legs, 3 for the torso, 32 for the arms and six for the head) was designed by the RobotCub Consortium [24], which involves several European universities and it is now widely used by the iTalk project and few others.

The iCub platform design is strictly following open-source philosophy and therefore its hardware design, software as well as documentation are released under general public license (GPL). Tikhanoff et al. have developed an open-source simulated model of the iCub platform [25], [26]. This simulator has been widely adopted as a functional tool within the developmental robotics community, as it allows researchers to develop, test and evaluate their models and theories without requiring access to a physical robot. The iCub was used in the current study with the MTRNN system controlling four joints of each arm. Each of these joints has different degrees of freedom constrained by the actual design of the iCub's body, and partly by the software for security reasons.

Table 1. Maximum range of joint movements

Joint	Range of movement
arm pitch	185º
arm roll	161º
arm yaw	137º
elbow pitch	112º

The sensorimotor states of the iCub were sampled at 100ms rate and were used for training a set of Self-Organising Maps (SOM). These sequences were further down-sampled to 500ms to simplify the learning process of the backpropagation through-time (BPTT) algorithm and to examine the precision of the learned sensorimotor patterns.

2.2 Self Organising Maps for Input Sparse Encoding

The MTRNN system used Self-Organising Maps (SOMs) as means of preserving the topological relations in the multidimensional input space to reduce the possible overlap between various sensorimotor sequences and to aid the learning process (see Figure 1 and 3). The self-organizing map was trained prior to the MTRNN's BPTT training using a slight variation of the standard SOM unsupervised learning algorithm [27]. The data set consisted of all the sequences used to for the MTRNN training as well as additional sequences, which involved variations to achieve smoother representation of the input space and minimize data loss incurred during the process of vector transformation. (1) shows the description of these vectors where $l(i)$ defines their dimensions.

$$v_i = \{v_{i,1}, v_{i,2}, v_{i,3}, \dots, v_{l(i)}\} \tag{1}$$

The transformation of a vector to a self-organizing map (SOM) is given by (2) where $v^{sample} = l(i)$, σ defines the distribution shape of $p_{i,t}$ and N represents the overall size of the selforganizing map.

$$p_{i,t} = \frac{exp\left\{-\frac{\left\|v_i - v^{sample}\right\|^2}{\sigma}\right\}}{\sum_{j \in N} exp\left\{-\frac{\left\|v_i - v^{sample}\right\|^2}{\sigma}\right\}} \tag{2}$$

correspond to an activation probability distribution of the self-organizing map whose inverse transformation generates multidimensional vector that directly sets the target joint angles of the iCub. (3) describes this transformation where v_i represents the target position for the i^{th} joint index, $y_{j,t}$ is the MTRNN's j^{th} output activity, $s_{i,j}$ is the i^{th} index of the vector corresponding to the SOM's node j.

$$v_i = \sum_{j \in N} y_{j,t}\, s_{ij} \tag{3}$$

2.3 Online control

The MTRNN's core is based on a continuous time recurrent neural network characterized by the ability to preserve its internal state and hence exhibit complex dynamics. The system receives sparsely encoded proprioceptive input from the robot, which is used to predict next sensorimotor states and therefore acts as a forward kinematics model (e.g. [28]).

The neural activities were calculated following the classical firing rate model where each neuron's activity is given by the average firing rate of the connected neurons. In addition to this, the MTRNN model implements a leaky integrator and therefore the state of every neuron is not only defined by the current synaptic inputs but also considers its previous activations. The differential equation (4) describes the calculation of neural activities over time where $u_{i,t}$ is the membrane potential, $x_{j,t}$ is the activity of j^{th} neuron, w_{ij} correspond to synaptic connections from the j^{th} to the i^{th} neuron and finally the τ parameter that defines the decay rate of ith neuron.

$$\tau_i u_{i,t} = -u_{i,t} + \sum_j w_{ij} x_{j,t} \tag{4}$$

The decay rate parameter τ modifies the extent to which the previous activities of the neuron affect its current state. Therefore, when the neurons are set with large τ values their activities will be changing more slowly over time as compared to those neurons set with smaller τ values.

In this experiment, 256 input-output neurons were set to $\tau = 2$ while the hidden neurons consisted of two different categories where each had different time integration constant. The first category comprise of 60 fast neurons with $\tau = 5$ and the second of 20 slow neurons set to $\tau = 70$. These two categories are attempting to capture the dynamics of complex behavioral patterns by flexible recombination of motor primitives into novel sequences of actions. As described in the introduction, multiple timescale networks have been suggested as the underlying system that facilitates this behavioral compositionally.

The network is fully connected and hence every neuron is connected to every other neuron including itself. There is one exception where the slow neurons are not directly connected to the input-output layer but rather indirectly via the fast neurons.

The continuous time integration model of the MTRNN's neurons were defined by the differential equation 4 while the actual membrane potentials are calculated by its numerical approximation defined by (8).

$$u_{i,t+1} = \left(1 - \frac{1}{\tau_i}\right) u_{i,t} + \frac{1}{\tau_i}\left[\sum_{j \in N} w_{ij} x_{j,t}\right] \tag{5}$$

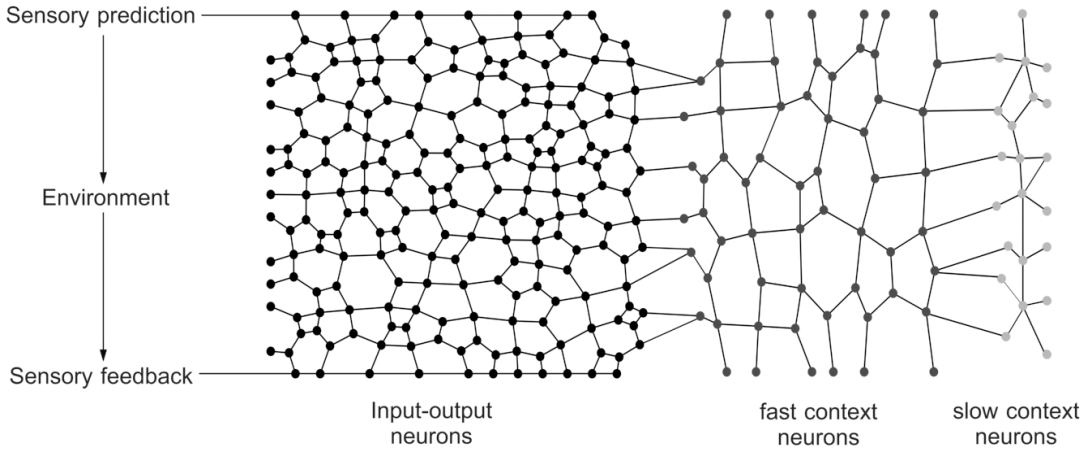


Figure 1. The system receives proprioceptive information as a multidimensional vector m_t subsequently activating a self-organizing map, the activity of which is associated to the network's input. The neural network then predicts the next sensorimotor state m_{t+1} based on its current state and input. At this stage, the neural activations on the output layer are assumed to correspond to the activity of the self-organizing map whose inverse transformation generates multidimensional vector that directly sets the target joint angles of the iCub.

The activity of neuron is calculated in two different ways (see (6)) depending on whether a neuron belongs to the input-ouput ($i \in Z$) or the hidden layer.

$$y_{i,t} = \begin{cases} \frac{exp(u_{i,t})}{\sum_{j \in Z} exp(u_{j,t})} & \text{if } i \in Z \\ f(u_{i,t}) & \text{otherwise} \end{cases} \tag{6}$$

Therefore, the input-output neuron activations are calculated using the Softmax function (the top part of (6)) while the hidden neurons use conventional Sigmoid function (see (7)).

$$f(x) = \frac{1}{1+e^{-x}} \tag{7}$$

The Softmax function was used to achieve an activation distribution that is consistent with that of the self-organizing map. The system receives proprioceptive information as a multidimensional vector m_t subsequently activating a self-organizing map, the activity of which is associated to the network's input. The neural network then predicts the next sensorimotor state m_{t+1} based on its current state and input. At this stage, the neural activations on the output layer are assumed to correspond to the activity of the self-organising map whose inverse transformation generates multidimensional vector that directly sets the target joint angles of the iCub. The iCub then updates the

positions of its joints, which are again fed back through the SOM into the MTRNN system as $x_{i,t+1}$. Hidden neurons are simply copied as the recurrent states for the next time step, see (8).

$$x_{i,t+1} = \begin{cases} p_{i,t+1} & \text{if } i \in O \\ y_{i,t} & \text{otherwise} \end{cases} \tag{8}$$

2.4 Back Propagation Through Time

The MTRNN needs to be trained via an algorithm that considers its complex dynamics changing through time and for this reason we used the BPTT algorithm as it has been previously demonstrated to be effective with this recursive neural architecture [2].

This learning process is defined by finding the suitable values for the synaptic connections minimizing the global error parameter E, which represents the error between the training sequences and those generated by the MTRNN. The error E is calculated using the Kullback-Leibler divergence as described in (9) where $y^*_{i,t}$ is the desired activation value of the ith output neuron at the time t and $y_{i,t}$ is its actual output.

$$E = \sum_t \sum_{i \in O} y^*_{i,t} log\left(\frac{y^*_{i,t}}{y_{i,t}}\right) \tag{9}$$

The synaptic connection values are updated according to (10) where their optimal levels are approached through minimizing their values with respect to $\partial E/\partial w$ that defines the gradient. The learning rate is given by parameter α and n represents the learning iteration step.

$$w_{ij}(n+1) = w_{ij}(n) - \alpha \frac{\partial E}{\partial w_{ij}} \tag{10}$$

The already mentioned gradient $\partial E/\partial w$ is defined by (11) while the recurrence equation 12 is used to recursively calculate $\partial E \partial u_{i,t}$.

$$\frac{\partial E}{\partial w_{ij}} = \sum_t \frac{1}{\tau_i} \frac{\partial E}{\partial u_{i,t}} x_{j,t-1} \tag{11}$$

The $f'()$ is the derivative of the sigmoid function defined by (7). The $\delta_{i,k}$ is Kronecker's delta, which is set to 1 when $i = k$ otherwise it is 0.

$$\frac{\partial E}{\partial u_{k,t}} = \begin{cases} y_{i,t+1} - y^*_{i,t+1} + \left(1 - \frac{1}{\tau_i}\right) & \text{if } i \in O \\ \sum_{k \in N} \frac{\partial E}{\partial u_{i,t+1}} \left[\delta_{i,k}\left(1 - \frac{1}{\tau_i}\right) + \frac{1}{\tau_k} w_{ki} f'(u_{i,t})\right] & \text{otherwise} \end{cases} \tag{12}$$

The initial values of the synaptic connections were randomly generated between -0.025 and 0.025. The first five slow neurons were set to different values for different behavioral sequences to allow their learning, which exploits the initial sensitivity characteristics of the continuous time recurrent neural networks [29].

3 Experiments and Results

This section presents results of the testing of the MTRNN and BPTT systems on the iCub robot. The experimental task required the MTRNN system to learn 8 different behavioral patterns (slide box left/right, swing box, lift box up/left/right, push/pull box).

The Sequence Recorder module of Aquila was used to record these sensorimotor patterns while the experimenter was guiding the robot by holding its arms and performing the above mentioned actions. This requires the activation of the response
of the robots actuators.

Figure 2. Tutoring the iCub robot while recording the sensorimotor sequences.

Every behavior was recorded three times with slight variations that involved 5cm offsets with respect to the center of the object. This was done to achieve smooth representations of the input space and reduce the errors incurred during the SOM transformations. This generated thousands of

sensorimotor sequences all of which were used to train the SOM prior to the MTRNN training that only used the original sequence (without offsets) for each behavior.

The self-organizing map consisted of 256 nodes and was trained (see Figure 3) using the Aquila's SOM module and all the data collected during the tutoring session, sampled at 100ms. In order to achieve a good precision of the SOM, it was necessary to run its training for 160,000 iterations using the initial learning rate of 0.05.

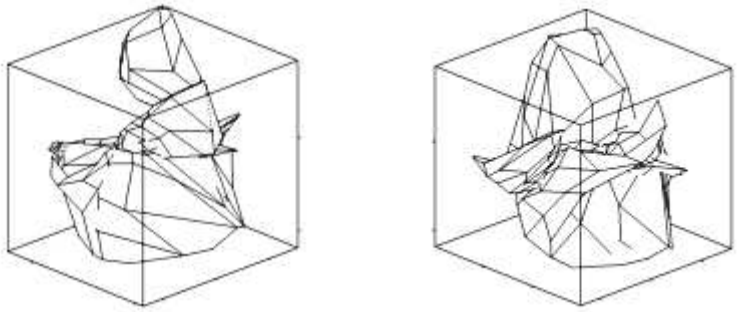


Figure 3. 3D visualization of the trained self organizing map. The left image shows the visualization of the left arm's input space and the right image is the visualization of the right arm's input space. The input space visualization of each arm was done via Aquila where the second, the third and the fourth joints were assigned x,y,z dimensions respectively

Five different learning trials were conducted, where each trial was initialized with a different seed used to generate random numbers for synaptic connections. The BPTT algorithm was set to run for one million iterations with the learning rate set to 0.015 and sigma parameter set to 0.0045. This computationally intensive training was possible through the utilization of a cluster of NVIDIA Tesla and Fermi GPU cards as well as the Aquila CUDA compliant module. Aquila implements both GPU and CPU versions of the MTRNN system. Our preliminary benchmark tests showed 12x speedup of the training algorithm and 75x speed-up of the neural network forward pass when using GPUs and MTRNN with 336 neurons. The more neurons the MTRNN system uses the greater the benefit of the GPU over CPU implementations.

At the end of the training, the learned neural network was tested on the iCub in the same setup as that during the tutoring part. The results from the first three trials showed that the MTRNN system was able to replicate all the eight sequences while successfully manipulating the object. The last two trials were not equally successful. While the fourth trial produced MTRNN capable of performing the first five behaviors, the last trial showed only hints of learning and was not able to replicate any action satisfactory. This can be seen from the

error, which was significantly higher than in the rest of the other runs (see Table 2).

These experiments revealed some interesting dynamics on the sensorimotor training the system. For example, the behavior of pushing the block involved a complex sensorimotor flow that is naturally constrained by the actual interaction with the object. This means that often the interaction with the object would be significantly different from the learned interaction and thus, in several cases, the MTRNN dynamics was very different from the original one. Interestingly, when this was the case, the iCub would spend a bit more time correcting its positions and only then it would push the block forward.

Table 2. Errors at the end of each trial

Run	**Error**
1	1.0026
2	1.2226
3	1.4509
4	1.7519
5	2.8045

The results presented herein demonstrated that MTRNN system is able to be extended to learn eight different behavioral sequences. This was a significant improvement from previous use of MTRNN. The number of learned behaviours in our case already exceeded the learning performance in Yamashita and Tani experiments [2] where the computational power required for the training and processing of SOMs was saved by using small input sizes, which might have consequently limited the number of learnable sensorimotor patters: *"If the sizes of the TPMs (SOMs) are set to larger value, representations in the TPMs become smoother and data loss in the vector transformation decreases. For the current experiment, however, in order to reduce time spent on computation, sizes of the TPMs were selected such that they were the minimum value large enough to allow the TPMs to reproduce, in real time, sensorimotor sequences through the process of vector transformation, the teaching sequences and output sequences"* ([2], p. 15-16).

This was not the limitation in our case since both the SOM and the MTRNN are massively parallelized and processed on the GPU devices, which allowed us to experiment with larger network sizes. In fact, it was found that the 64 neurons used to represent the proprioceptive input space were not enough in our experimental scenario. There seem to be three primary reasons for this. The first is the fact that the number of sequences was higher in our case, and therefore more nodes were needed to smoothly represent the input

space. The second is due to higher complexity of the learned sensorimotor sequences, which is particularly true for pushing and pulling behaviors. And finally, the iCub's joint angle ranges are significantly higher than those of the Sony QUIRO used in Yamashita and Tani experiments [2].

4 Conclusions and Future Work

We have showed that the MTRNN model was able to learn eight different behavioral sequences. These constitute the motor primitive for ongoing experiments for the learning of action and language compositionality.

Current developments involve the use of three additional self-organizing maps, linked to the MTRNN, and trained to represent simple linguistic inputs, as well as the object shapes and color features, obtained from images fed through logpolar transform inspired by human visual processing. This extension will facilitate our investigation of language learning.

In particular, our next experiments will be addressing a specific linguistic hypothesis first proposed by the cognitive psychologist and linguist Michael Tomasello. The hypothesis, which is also known as the verb island theory, predicts that verbal argument structures are learned on a purely itemspecific basis [30]. In other words, children do not learn that adult-like verb constructions can be combined with certain types of nominals and clauses, e.g. to get a transitive directobject structure. Rather, children acquire verb concepts by developing this knowledge on an item-by-item basis where the understanding of verbs is at first limited to the context where these verbs appeared. Consequently, the general notion of transitive construction and direct object is an abstraction that occurs only during later developmental stages when a critical mass of these verb islands have been attained and thus recognized as the instances of the same general underlying (sensorimotor) structure through the process of semantic analogy [31]. During the later developmental stages children are exposed to more and more construction types where different semantic roles are linked in a similar way and as a result the involved syntactic categories will be abstracted into subjects and objects [32].

The first planned experiment will investigate the role of semantic similarities between different words during early language acquisition. In particular, the hypothesis addressed by this experiment is whether a generalization to unheard sentences is easier in condition where all learned events are of the same semantic type. Though conceptually simple, this experiment will constitute the first viable extension of the already conducted research within the iTalk Project. In addition, these problems are also discussed in child development research and therefore this work could provide useful insights.

The extension of the experiment will investigate the effects of using different learning techniques such as holistic, scaffolded and parallel learning. There are several other possibilities for farther experiments on which we are yet to agree, however, the experiments outlined in this section present an important step towards expanding our current knowledge of action-language integration as well as the acquisition of more complex grammatical constructions.

Acknowledgement: This work was supported by the EU Integrating Project - ITALK (214886) within the FP7 ICT programme – Cognitive Systems, Interaction and Robotics.

References

[1] Arbib, M., *Neural Organization: Structure, Function, and Dynamics*. Cambridge: MIT Press, 1998.

[2] Yamashita, Y. and Tani, J., "Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment," *PLoS Computational Biology*, vol. 4, no. 11, 2008.

[3] Kuniyoshi Y. and Sangawa S., "Early motor development from partially ordered neural-body dynamics – experiments with a cortico-spinalmusculoskeletal model," *Biological Cybernetics*, vol. 95, no. 6, pp. 589– 605, 2006.

[4] Mussa-Ivaldi, F. and Bizzi, E., "Motor learning through the combination of primitives," *Philos Trans R Soc Lond*, vol. 355, pp. 1755–1769, 2000.

[5] Jeannerod, M., The *Cognitive Neuroscience of Action*. Cambrige, MA and Oxford UK, Blackwell Publishers Inc, 1997.

[6] Rizzolatti G. and Luppino G., "The cortical motor system," *Neuron*, vol. 31, pp. 889–901, 2001.

[7] Sakai, K., Kitaguchi, K. and Hikosaka, O., "Chunking during human visuomotor sequence learning," *Experimental Brain Research*, vol. 152, pp. 229–242, 2003.

[8] Thoroughman, K. A. and Shadmehr, R., "Learning of action through adaptive combination of motor primitives," *Science*, vol. 407, pp. 742–747, 2000.

[9] d'Avella, A., Portone, A., Fernandez, L. and Lacquaniti, F., "Control of fast-reaching movements by muscle synergy combinations," *Neuroscience*, vol. 26, no. 30, pp. 7791–7810, 2006.

[10] Graziano, M. S., Taylor, C. S., Moore, T. and Cooke, D. F., "The cortical control of movement revisited," *Neuron*, vol. 36, pp. 349–362, 2002.

[11] Giszter, S. F., Mussa-Ivaldi, F. A. and Bizzi, E., "Convergent force fields organized in the frog's spinal cord," *Neuroscience*, vol. 13, pp. 467–491, 1993.

[12] Wolpert D. M. and Kawato, M., "Multiple paired forward and inverse models for motor control," *Neural Networks*, pp. 1317–1329, 1998.

[13] Tani, J. and Nolfi, S., "Learning to perceive the world as articulated: an approach for hierarchical learning in sensory–motor systems," *Neural Networks*, pp. 1131–1141, 1999.

[14] Tani, J., Nishimoto, R., Namikawa, J. and Ito, M., "Codevelopmental learning between human and humanoid robot using a dynamic neuralnetwork model," IEEE *Transactions on Systems, Man, and Cybernetics - Part B*, vol. 38, pp. 43–59, 2008.

[15] Tani, J., Nishimoto, R. and Paine, R., "Achieving "organic compositionality" through self-organization: reviews on brain-inspired robotics experiments," *Neural Networks*, vol. 21, pp. 584–603, 2008.

[16] Newell, K., Liu, Y. and Mayer-Kress, G., "Time scales in motor learning and development," *Physical Review*, vol. 108, pp. 57–82, 2001.

[17] Huys, R., Daffertshofer, A. and Beek, P. J., "Multiple time scales and multiform dynamics in learning to juggle," *Motor Control*, vol. 8, pp. 188–212, 2004.

[18] Varela, F., Lachaux, J. P., Rodriguez, E. and Martinerie, J., "The brainweb: phase synchronization and large-scale integration," *Nature Reviews Neuroscience*, vol. 2, pp. 229–239, 2001.

[19] Honey, C. J., Kotter, R., Breakspear, M. and Sporns, O., "Network structure of cerebral cortex shapes functional connectivity on multiple time scales," *Proceedings of the National Academy of Sciences*, vol. 368, pp. 10 140–10 245, 2007.

[20] Poeppel, D., Idsardi, W. J. and Wassenhove, V., "Speech perception at the interface of neurobiology and linguistics," Philosophical *Transactions of the Royal Society B: Biological Sciences*, vol. 368, pp. 1071–1086, 2008.

[21] Peniak, M., Morse, A., Larcombe, C., Ramirez-Contla, S. and Cangelosi, A., "Aquila: An open-source gpu-accelerated toolkit for cognitive robotics research," *International Joint Conference on Neural Networks*, San Jose, California, 2011.

[22] Cangelosi, A., Metta, G., Sagerer, G., Nolfi, S., Nehaniv, C., Fischer, K., Tani, J., Belpaeme, T., Sandini, G., Fadiga, L., Wrede, B., Rohlfing, K.,

Tuci, E., Dautenhahn, K., Saunders, J. and Zeschel, A., "Integration of action and language knowledge: A roadmap for developmental robotics," *IEEE Transactions on Autonomous Mental Development*, 2010.

[23] Metta, G., Natale, L., Nori, F., Sandini, G., Vernon, D., Fadiga, L., Hofsten, C., Rosander, K., Lopes, M., Santos-Victor, J., Bernardino, A. and Montesano, L., "The icub humanoid robot: An open-systems platform for research in cognitive development," *Neural Networks*, vol. 23, no.8-9, pp. 1125–1134, 2010.

[24] Metta, G., Vernon, D., Natale, L., Nori, F. and Sandini, G., "The icub humanoid robot: an open platform for research in embodied cognition," *IEEE Workshop on Performance Metrics for Intelligent Systems*, 2008.

[25] Tikhanoff, V., Fitzpatrick, P., Nori, F., Natale, L., Metta, G. and Cangelosi, A., "The icub humanoid robot simulator," *International Conference on Intelligent Robots and Systems IROS*, Nice, France, 2008.

[26] Tikhanoff, V., Cangelosi, A. and Metta, G., "Integration of speech and action in humanoid robots: icub simulation experiments," *IEEE Transactions on Autonomous Mental Development*, vol. 3, no. 1, pp.17–29, 2011.

[27] Kohonen, T., *Self-Organizing Maps*. Springer, 1996.

[28] Wolpert, D. M., Ghahramani, Z. and Jordan, M. I., "An internal model for sensorimotor integration," *Science*, vol. 269, pp. 1880–1882, 1995.

[29] Nishimoto, R. and Tani, J., "Learning to generate combinatorial action sequences utilizing the initial sensitivity of deterministic dynamical systems," *Neural Networks*, vol. 17, pp. 925–933, 2004.

[30] Tomasello, M., *Constructing a language*. Cambridge, Massachusetts: Harvard University Press, 2003.

[31] Goldberg, A., "The emergence of the semantics of the argument structure constructions," in *The emergence of language*, B. MacWhinney, Ed. Lawrence Erlbaum, 1999.

[32] Dittmar, M., Abbot-Smith, K., Lieven, E. and Tomasello, M., "Young german children's early syntactic competence: a preferential looking study," *Developmental Science*, vol. 11, no. 4, pp. 575–582, 2008.

What we mean when we talk about the mind

Karel PSTRUŽINA[1]

Abstract. In the paper we try to point out the relationship of mind and brain. The dominant solved in this traditional relationship is missing third member, without which any solution is always one-sided. For this we consider the missing third member of human thought, which characterize a continuous movement of the intentional contents of mind, with tones, with numbers, etc. but human thought itself is only the initiator of this movement.

1 Human thinking

At the beginning we try to define the mind. We believe that this question can be answered only by such way that we ask two questions, namely:

- what is happening in the whole mind;
- and how the mind could be included in the whole of being.

These questions are linked together

Our preliminary answer is: the mind is being. This is the ontological status of mind. However, the mind is such a being, which is the medium of movement of human thought and also a necessary condition for occur of human thought. The mind is something that allows the movement of human thought. It is something which enables to human thinking saved stimuli that are entranced from the beings. Human thinking them picks up that and operates with them. The mind is not defined to the brain, but the human thinking. Therefore, we also seem that the traditional Mind - Body problem is a difficult to solve. The ontologically-oriented point of view means that the mind is being, and everything else is already resulting from there.

There are two beings (mind and brain) intended thought in our opinion. They are symbiotic to the human thinking. Without both (i.e., without mind and

[1] The University of Economy, Department of Philosophy, W. Churchill Sq. 3, 130 00 Prague. E-mail: Pstružin@vse.cz

brain) the perpetual course of human thinking was not possible. But it is also possible to say that the human thinking is constantly moving and by its perpetual moving completes concrete form how of the brain (therefore we speak about plasticity of brain) and mind. Every movement of human thought is reflected in brain as changes in the plasticity of the brain, especially the creation of new synapses and their changes; and the human mind is changing by fillings with new intentional contents (the mentals); and in both (the brain and mind) because that the human thinking uses the specifics of their structural arrangement for its movements or operations. Mind and brain therefore are not correlated at the causation relation or priorities, but both (mind and body) are in relation to the movement (it means movement of human thinking). These beings are therefore dependent on the human thinking as well as human thought is dependent on them. Human thought is the third member of the equation on the mind - body problem.

Mind and brain are separate beings, even if they are interconnected being, or are intertwined. What will happen in one of that has resulted in the second expression of being too. In the mind-body problem such a view is known as supervenience. The supervenience represents such opinion: what happens in the mind immediately is reflected in the brain and vice versa, even if not everything, what happens in the brain and mind we are aware (parts of the mind are unconscious structures too). But in supervenience initiation and causation are either on the side of the brain or the mind.

Our concept of the classical solution to the mind-body problem differs mainly in such point that the initiative is on the side perpetual movement of human thought. It is the human thinking, which moves and causes changes in the brain and also in mind. Changes in brain and mind are dependent on the movements (or operations) of human thought, but not all the movements of human thought become aware. Causation is clearly on the side of human thought.

Mind - Body problem, as is usually called in the current literature is multidimensional and therefore allows a variety of approaches. Just for these approaches are controversy waged, let alone the content of individual arguments. Supporters of physicalism and mentalism are not in the dualistic approach of reconciliation, but rather the dualism as one of the warring parties seeking to explain the Mind - Body problem. Dualism also set aside "Double aspect theory." Gradually, then even begin to specify the school of functionalism and behaviorism or phenomenology, one of the parts to the problem highlighted.

I have no aim of this paper to give an overview of different approaches, but speak to the merits of Mind – Body problem. My opinion consists at the view that the mind is a separate quality.[2] That does not mean that the manifestations of mind are not determined by specific processes in the central nervous system, but it is also important to note that the mind can determine not only how the stimulus go of neuronal pathways, but also the very existence of the central nervous system.

Ontologically status of the mind can not be primarily explained by the process of its emergence from the brain processes, it means by the processes that are founded on the quantity and frequency of stimuli in the brain, but only by such way so that we will from the beginning to focus on very own process of mind. At it is human thinking and its perpetual movement that create main part of mind processes.

The most constant stream of thought activity can be identified with the inner speech in everyday life, or as a generating idea process during day dreaming. This is taken as the ever vigilant cogitationes flow, which is an internal commentary and which present essential elements our self-identity. It can also be taken in connection with logical operations performed to solve problems, or furnishing of memory records. In all these cases, the thinking is closely tied to language or images and thinking is overlapping by them. It is the cause why thinking is often identified with the language.

From our previous considerations it follows that by this way thinking can not be identifying in the ontological sense. Thinking is much more process itself, which can operate with anything (concepts, ideas, notes, symbols, numbers ...), always with something, but thinking itself is only this pure stream that concepts, ideas, notes, symbols, numbers, ... carries, mixes them, but thinking is behind them as something separate and independent.

So if you characterize thinking as perpetual discourse, then thinking can not be based on the existence of which would be an expression, but its spontaneity. So we consider thinking as autonomous thinking in terms of sufficient reason, and non-transferability on the irreducibility on anything else. That's because thinking is simple stream. Thinking is a simple movement, which in its permanency articulates the fact that it is always thinking "about". It may seem that this form of revelation is his reliance on its being, which manifests itself as one of its modes, performing other differences, but our response is the opposite.

[2] It is many books that deals with Mind – Body Problems, for example [1], [2], [3]

A simple movement, which is thinking, always creating reflection of being and by this way thinking constitutes things and the world.

Constitutional activity of thinking is usually linked to speech, to “logos”, but the thinking is movement, not language itself. Thinking is something what enables speech. However, it may break away from the speech and perform reflection being that exists at the level of perception, or on the other side at the level of certainty of the truth about essences that speech is not able to capture.

If we want to differ essences and their processes of constitution as follows movement of thought, it is necessary to distinguish a layer of thinking from the layer of beings. If we did not do so, it would be ontological conception in contradiction. That's because it does not allow it to accommodate the multiplicity as a totality and a differentiation of the thinking and what there is another form of existence and how it is possible. It should be borne in mind that even the initial exclusion of thinking can do nothing else but thinking itself. Thinking is not deducible from being, because it would first have to devote itself.

Much easier is to accept of independence of thought, which is in addition to being and which carried out the difference in essences.

How can we explain in detail the processes of differentiation of entities which is carried out by thinking.

Thinking in discourse performs comparisons between current percepts and endocepts, ie, thinking is saying "that's not it" or "that's it." Basically, these processes are therefore negation and identity. For example, when we want to characterize what is white, then our thinking is saying white is what is not red, what is not blue, what is not black, what is not yellow and so on. And when we are looking on something white then our thinking is saying it is like white. Percepts are constitutes by this way.

However, if we do not built own mental picture of the world's, it means endocepts that precedes perception (it is probably before puberty), or if it is a new sensation, the mind does not have its own basis for comparison, we create then the things on comparisons with other sensations. This process is based on negation, because we cannot determine the percept by otherwise. Let's say that in the prepubescent stage of ontogenetic development we determine all things on the level of sense. For example "salinity" as something what is "not bitter", "non-acidic," "not sweet".

Thinking has another domain, too. This domain is speech. Each term is a negation, and thus the constitutions of essences are accomplished. The concept in itself automatically includes possible essence over other essences and thus their constitution. The concept has already been undertaken and therefore the difference after a certain objectivity, appointment. Thinking there might not compare the concept with other concepts or percept to determine essence, but the concepts can be combined and thus our thinking works towards a context of a world.

The thought acts as an independent and irreducible to being, as something what is permanent and what is spontaneously at both levels, it means at level of perception and level of rational thinking. Thinking dispose by both of these characteristics, it is permanently movement and spontaneity are crucial for granting ontological independence, because they mean that initiation of thinking is coming from thinking itself and not in anything outside. Thinking is also unceasing stream, which manifests itself, i.e. binding with concepts, ideas, notes, symbols, numbers ... which articulates. Thinking does not occur due to concepts, ideas, etc., but because they exist due to thinking. They are tied to the thinking, not thinking of them.

2. Creation of concepts

We can demonstrate the role of thinking on constitution of concepts.

First, we consider the creation of concepts how to solve this problem I. Kant.

If we read Kant [4], then we learn from him, especially considering that the concept is in general at individual or at concrete. And further, that the formation of the concept is based on three operations: reflection, comparison and abstraction. The most important operation of human thinking is here reflection which includes the other two above-mentioned operations (i.e. comparison and abstraction). These three operations are not independent of each other, or perhaps successive, but it is the only one complex operation, which searches for something that is common in particulars while this compares with other items.

Furthermore, it is noted also that Kant, when considering the abstraction of the operation as one engaged in the development concept, tends to abstraction as abstraction from something, not as an abstraction of something. M. Heidegger notes in this context: Abstractions for Kant is not looking for unity, but leaving aside differences [5].

How, then, according to Kant, the concept arises:

Kant points out reflection as a main operation when we construct concepts. The Reflection is an operation of the human thinking that is able to reflect something (what constitutes the general feature in a particular case) in consciousness. During this operation is just something taken in as a concept. What follows is captured and it is a generality, but the generality of diversity. In operation of reflection is the spontaneity taking into account of something general, which then further compare with other specific things, and so distinguish what is specific for each concreteness and also to see what they have in common specificity.

Here we are at issue, which sets us apart in their approach to the clarification of the term from Kant and therefore we need to further clarify our position.

Our position is as follows.

In creation of the concept three operations play their role: comparison, generalization and abstraction. These three operations are logically connected and they complement each other. This is not just Kant´s term reflection is replacement by generalization, but mainly about the fact that human thought is always splits into two streams (primary and secondary streams of human thinking). The second stream reflects all processes automatically, i.e. It bends and reflects everything that is performed by the primary stream of thinking and thus it is conveyed to consciousness. Starting point of concepts creation, in our opinion, is finding of essence, which compares the essences of the phenomena and other things before finding something general in the particular.

The fact we get through abstraction, but abstraction is in our view, a different form than it is often considered. This is not a discarding. The abstraction is not an act, when we from the whole come to the particulars, it means to characteristics, properties, or varieties, but for our opinion the particulars, characteristics and so on are in mind in advance.

Here also I differ from Husserl and his method of phenomenological reduction.[6] According to Husserl the phenomenological reduction is the bracketing of any opinion on the objective world. What does it means that we give objective world into brackets? Such a concept is nothing other than abstraction of something insignificant and so we are still something is what makes the thing things. It is how Kant understands to abstraction. However, abstraction for Husserl not impoverish the world, but rather enriching it,

because we reduce the world on one side, but we also produce something that is not in the world. We produce the substance, we produce pure ideality. But come back to the reduction. However, we can only guess what we throw away when we realize phenomenological reduction? They could be two possibilities:

- Either we act so that it will try it and we will imagine that the thing can do without the other properties and therefore these properties are not essential for the úsiá (essence);

- Or we have some premonition of what is the nature of thong and on this basis we identify irrelevance properties of thing.

Husserl responds that *by making this epoché we obtained myself, with his own pure life of consciousness in which and through which all the objective world exists for me - in a way which is right for me. The world for me ... is not nothing but a world that is in such my cogito consciously being for me and having a validity for me. We have the whole world in itself in our consciousness. So the natural world - that world, I am talking about and I can speak about- precedes de facto existence the pure ego and his cogitationes as being on that is earlier.*[6 § 56]

Our approach is something else. We believe that the constitution of any thing as a whole is always the composition properties of the stimuli that were pulling from sensors of entities, which are already in advance as a singular. They are what the human mind recognizes through operations identity and negativity, and these properties are combined by human thinking (given together) so way that we realize constituted a thing (for example a cup) as a whole. Properties have their basis in existence. Human thinking is only recognized that as a specific feature in that they are assigned to something what was already defined before by operation of identity and of negativity in encapsulated structure of endocepts. Property is what is formed and what stands out in relationships. Without any relationship can not reveal what actually is, can not reveal their essence.

The stimulus comes to the brain even different neuronal pathways and they are also in different parts of the brain identified. For example the visual center of the brain is divided into several areas. One of them, for example, identify vertical lines and again from those stimuli that mark the horizontal, in other parts of the brain are different colors, is identified in another movement, etc.

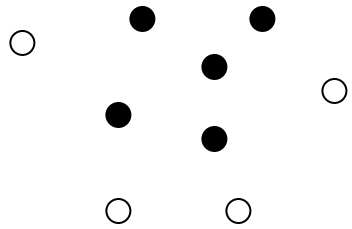
We know 32 visual cortical areas with 305 linked by mutual connections at makaka monkey and 7 areas in addition is processing visual stimuli and other stimuli. Neurons each area behaves differently and they have different areas and different inputs and outputs [7p.154].

Such stimuli are synthesized by human mind into the final version, which is referred as a things and constitution of mentals.

We acquire the impression that we have made abstraction that we cast out some features that do not specify the essence of thing, but actually we just got to singular stimuli as human thought get out from beings. What we call an abstraction it is for us only perpetual orientation movement of human thought in pre-constitutive stage in which individual properties are identified. Human thinking and its movement (their experienced as a time sequence), however, can sometimes run into other properties, other characteristics that must be identified and included under a new concept, it is necessary to define them as specific and as such include the relationship to something already others. The property itself is not the essence.

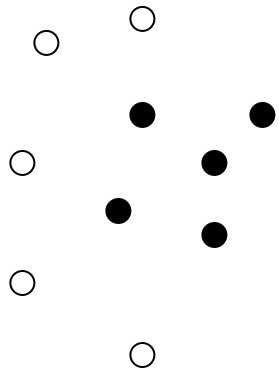
If it goes in the human mind pre-constitutive stage may occur simultaneously comparisons, it means to such movements of human thought, which are currently initiated by neuronal circuits compared to each other with neuronal circuits (memory traces) of entities previously pulled out of similar stimuli. Everything is on hold and the current setting of reflexive reverberations. The primary stream of human thought retains the current stimulus and the secondary stream is engaged by encapsicity structure of endocepts and looks for similarity. This leads to the identification of what is constituted for the same thing earlier, contained in the mind of sensations. For both of these neuronal circuits are eliminated those ones with some previous perceptions lacking and yet the human mind from these stimuli constituted a similar thing and it's mass compares the sameness of all listed below (under the same concept), to clearly distinguish what essence consists, therefore, what thing is, what is its essence, no matter what has not the particular cases. These processes are spontaneous and largely unaware. These are processes that already knows by the child, but we cannot define and which are almost not able to learn. This means that these automatisms, which can be Kantian a priori processes.

Here we want to illustrate our conception.



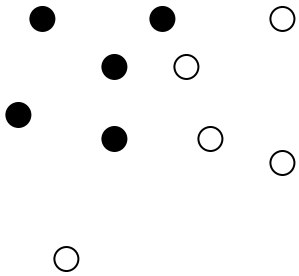
The picture 1.

We assume that the picture 1 represents the active neural circuits when we look on the dining table.



The picture 2

The picture 2 represents the active neural circuits when we look on writing table.



The picture 3

And the picture 3 represents the active neuronal circuits when we look on table tennis.

They are engaged different neuronal circuits but black ones are active all the time when we look on any table. Such neuronal circuits represent the essence of table. The white neural circuits are active as well, but they represent not essence of table, but some other attributes, or quality of different tables.

Abstractions are eliminative such operation of human thinking at the pre-constitutive stage. But the essence is only a single trait (represented by some neuronal circuit that could be at several neuronal modules). Sometimes it can be also a summary of several individual properties. The concepts are only names for essences and they make base for the very similar things with the same essence.

3. The Mind

Now we can come back to mind and its relation to brain and to human thinking.

If we accept theses that thinking is like being, that thinking is independent on the brain and thinking play main role when we constitute the world.

It seems an elegant solution would be adoption of the opinion that what happens in the brain and what happens in the mind are one and the same things. It is just different names the same processes that depend only on how we have a dictionary for these processes. With this view M. Warnock appeared at her work "Memory".[8] M. Warnock also says that there is only one objection that can be taken seriously, and it is a local determination of brain processes and inability to place an ideas (or mentals), because one and the same thing cannot be somewhere in the area of placement, and simultaneously does not be localized.

This aspect, however again drags us into the same problem. It has to be seen how we use the term localization. The concept of space is for M. Warnock superior categories to mind and body. The space is something that should make the transition between mind and body, or space is able to unify of both worlds. The space of thought processes may not be the same as the space of physical processes. The world of thought need not be in any space, or space of thinking may have completely different structure. And so it is with all the categories. If we consider thinking as ontologically separate, then it is necessary for this plane of existence of thinking to create also a separate categorical apparatus that allows an understanding of this world without constant comparison to the world of physical entities.

And also the question of authenticity, whether it is solved by deriving, or even independence based on emerge of thinking, is quite superfluous. When we accept that the thinking emerge from the quantity of neuronal stimulus, then we explain thinking as separate and irreducible in the ontological sense, but also as deducible from the base of being and its development.

The concept of emerge of thinking is trying to bridge the perspectives of physicalism and mentalism that prevail at certain times depending on how it transforms our understanding of the workings processes of the brain. Physicalism prefers physical processes, or brain processes standing against what are a mental processes and so called double-aspects theory is nothing more than the equivalent of psycho-physical parallelism.

Let's look more closely at the various positions so that we will try to clarify what is happening in us at a certain phenomenon - for example, if we drink water and how this phenomenon explains physicalists and mentalists.

If I am thirsty, then the idea or our feeling of thirsty arises by this way, that my body impulses to the brain and these are processed, resulting in feelings of

thirst. The body needs water and thus transmits impulses. The actual need is a basic stimulus that is transmitted to the brain where stimuli are processed. Our thinking reflects our needs of water and it focuses our behavior on searching source that could be satisfying our physical needs. There is no doubt that such processes occur at our heads, and that the explanation offered by physicalism, largely affects the process of awareness of the needs after that following human behavior. Mental processes are only subject to physical processes occurring in the body and brain. They are the result of allowing what their most promising and effective satisfaction.

But what happens when the body is sufficiently saturated with water and still feel the desire to drink, as for example when we are sitting with a friend in a convivial entertainment, or better yet, if the bet with a friend to drink 5 liters of water. Is it a stimulus that can also be explained within the framework of physicalism?

Certainly not. In this case, the body does not feel the need for fluids and body does not transmitting impulses to the brain, and even we are not aware of thirst, and despite of that we are drinking water. Here we do not speak, that we are very needed for fluids, but we are motivating for drinking a water by our mind. The motif for us could be a bet, or prestige, which is achieved when we win the bet, or it could be our will that proves our abilities. Does social prestige be explained in terms of physicalism? I suppose not. Social prestige or the will are a mentalistic aspects, it means that mental awareness determines the processes in the brain and subsequent behavior, including the activities of individual organs of the body?

To explain the prestige or the will from the point of view of physicalism terms, we would probably commit many inaccuracies and explanation would be very difficult and probably would be out of all aspects.

The explanation of mentalism is more acceptable in this case, because it is based on the specific mental processes, which take precedence position over programs which work only on base of physical aspects. The most common interpretation of these mental processes is the idea that mental processes emerge from brain processes similarly how for example property of fluid emerges from quantitative accumulation of water molecules. My opinion is that the mental is irreducible quality, that can be discover only as new entities and it cannot be based on structural accumulation of physical processes. It means in our particular case, that there are mental quality - the will to drink water, or

social prestige, they are not only accumulation of neurons and their bioelectric and biochemical processes.

Ontological terms an mental is the concept that can be expressed by words K.R. Popper "*there is a real novelty*." There are such a being, which are autonomous and irreducible to its base. Such conception is against the other concept (if I borrow again K.R. Popper's terminology) "*nothing new under the sun*", which means that everything is, all the diversity of existence has its original foundation in ones (perhaps like Parmenides spoke about) and this underlies the importance of its irreducible.[9 p. 14] In our case, the mental processes can not be reduced or explained by biochemical and bioelectric activity of neurons or neuronal modules, but one's mental processes are independent, they are superior to neuronal processes and mental obey brain processes. Which would mean specifically that the need for the prestige, or the will, as a mental process compels such an exchange of biochemical and bioelectric impulses to neural networks that the man will be drinks water, even though he is not thirsty, because he is subject who dictates to brain processes. Mental is cause to brain processes.

The will or mental processes determine which programs will be generated in the brain and which will be inhibited. Mentalism in all cases, prioritize the mental processes as a prior of physical. The soul controls everything including bioelectrical pathways and biochemical processes.

The processes of emerge new quality is trying to explain the how novelties emerge of separate and independent processes, or substances. The key issue for the granting of autonomy and irreducibility of mental processes is a form of their creation. These processes explain new quality by the way of accumulation of elements over a critical limit, when these elements become part of new structure where the whole is more then the parent elements.

Each element is in its relations with other elements, and in these relations shows what is reflected in their properties and their specifics. It is based on the continuing effort by the internal arrangement how they are in relations with other elements, along with which it forms a structured whole. The element is in the network of interactions with other elements, and its position is determined and by attractors, it means by such interactions, which allows it best to use its internal structure. The attractor is more authentic interaction. Attractor implies an element with favorable characteristics of other elements. It is the interaction allowing preserving the element and the element is attracted by the interaction.

The successor is subsequent interactions in which the element reproduces its relations with other elements of the structure.

Here is the beginning of a new quality by the processes of networks, because the element reinforces only those interactions that are its strengths and getting so dependent to the structure in which it operates. New quality is drawn into the interactions and actually no longer new quality is something separate, but it is only possible as part of a structure of the whole. If this structure is sufficiently complex in terms of its reproduction, it means if its complexity has exceeded a critical threshold, then a new quality is already a real novelty in the K.R. Popper sense.

Accumulation of elements above this critical threshold is mostly due to the fact that the structure of system must respond to their environment. Therefore, this structure binds with other structure, having a strong attractor. And so the structure comes into interaction with others and if this interaction is very strong then their internal interactions are more appropriate in terms of a new whole.

The whole disposes by internal dynamic in nature, not only because it constantly reacts to its surroundings, but also responds to its internal states. Some of its elements have a negative successors, other attractor with attracting more and more new elements, and thus transforming the original network interactions. The whole has external and internal stimuli, which are compensated, and it is actually a novelty that is irreducible to its elements, or subparts. It is necessary to explain this structure to separate from its behavior.

The idea that novelties emerge may thus explain the psychological needs and to obey all the bioelectric and biochemical stimuli in the brain under the dictation of these mental processes, but also manages to explain this principle of physical processes.

Literature:

[1] Priest, S.: *Theories of Mind*. Penguin Books, 1991.
[2] Armstrong, D., M.: *The Mind – Body Problem*. Colorado, Perseus Book Group 1999.
[3] Nosek, J.: *Mind and Body in Analytical Philosophy* (*Mysl a tělo v analytické filosofii*) Prague, Filosofia 1997.
[4] Kant, I.: *The Critic of Pure Reason* (*Kritika čistého rozumu*) Prague, Oikúmené 2001.

[5] Heidegger, M.: *Phenomenological interpretation of Kant´s Critic of pure Reason* (*Fenomenologická interpretace Kantovy Kritiky čistého rozumu*), Prague, Oikúmené 2004.
[6] Husserl, E.: *The Idea to pure phenomenology and phenomenological Philosophy* (*Ideje k čisté fenomenologii a fenomenologické filosofii*), Prague, Oikumeně 2004.
[7] Crick, F.: The Astonishing Hypothesis (Věda hledá duši), Prague, Mladá fronta 1997.
[8] Warnock, M.: *Memory*. London, Faber & Faber Ltd. 1987
[9] Popper, K., R., Eccless, J., C.: *The Self and its Brain*. Routledge & Keagan Paul Inc. 1977.

Logic and Cognitive Science

Igor SEDLÁR[1] and Ján ŠEFRÁNEK[2]

Abstract Our aim is to show that the logical point of view and methods of logic are indispensable for the understanding of human cognition. However, the results of some well known psychological experiments may be seen as denying the relevance of logic in studying human reasoning. Wrong design decisions and interpretations of these experiments are analyzed in this chapter. Arguments supporting an externalist position for a level of descriptions of cognition are presented. Finally, relations of logic and actual human reasoning are analyzed and illustrated on some examples.

1 Introduction

Our goal is to argue that logic is relevant for cognitive science and that the contribution of logic to the understanding of human cognition is fundamental.

Knowledge and reasoning are essential capabilities and results of human cognition. Our approach is based on an externalist viewpoint. According to this viewpoint, knowledge and reasoning can be studied as objective phenomena, independent on neural and mental processes.

We will analyze two psychological experiments that can be seen as implying that logic is not relevant for understanding of human reasoning, viz. the selection task and the suppression task. We shall argue that these experiments are based on misleading design decisions and the interpretation of their results aimed against the relevance of logic is not justified.

Subsequently, the 'logical point of view' is presented. We specify the relevant types of problems and the logical method for studying knowledge and reasoning.

After that, a variety of particular logical systems and ways of doing logic is described. An attempt to characterize cognitive tasks corresponding to different logical systems is presented.

[1] Department of Logic and Methodology of Science, Faculty of Arts, Comenius University, Bratislava, E-mail: sedlar@fphil.uniba.sk

[2] Department of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University, E-mail: sefranek@ii.fmph.uniba.sk

2 Cognition and Truth

The goal of this section is to argue that important features of cognition and cognitive abilities are connected to the external environment. Most importantly, contents of sound cognitions are crucially dependent on the state of the external world. Consequently, knowledge and reasoning can (and should) be studied as objective phenomena, independent on neural and mental processes.

Some features of cognition are recognizable even on low biological levels. Cognitive biology considers the ability of living agents to distinguish on the molecular level, cell level and the level of simple organisms as an exhibition of elementary cognitive capabilities. According to Kováč [15], biological evolution is a progressing process of knowledge acquisition.

The analysis of behavior of apes, dogs and other animals with an observable level of cognitive abilities leads to conclusions that the animals are able to reason and that they have knowledge about the external world.

Living agents (e.g., dogs, apes, and sometimes also people) observe results of their own actions or of actions of other agents. They distinguish success or failure of actions and learn on the basis of such observations etc.

Everyday behavior of living agents forces some kinds of reasoning and of knowledge acquisition. External conditions and criteria are crucial for the successful achieving of goals, for confirmation or supporting of the acquired knowledge. Correctness and usefulness of reasoning is tested with respect to external conditions.

The theoretical stance emphasizing the role of external conditions, when truth of a piece of knowledge and correctness of an act of reasoning is considered, shall be called *externalism* in this paper. Of course, knowledge and reasoning are supported by some mental and neural processes. The point of view, which abstracts from the role of these processes may be called *epistemological*. The basic ideas of our understanding of externalism and the epistemological point of view are discussed below.

2.1 Cognition: Belief, Confirmation and Falsification

The behavior of agents in some new, not completely known conditions and tasks is usually a process of trial and error. Agents observe responses of the external environment to actions, learn from the results of this process and fix the corresponding knowledge or beliefs (we do not distinguish here between knowledge and belief, even if it is possible and sometimes also necessary).

An epistemological translation of the paragraph above is that agents confirm or falsify their beliefs, while the confirmation or falsification takes the external environment into account.

2.2 Cognition and Reasoning

Let us discuss the role of reasoning within the tasks of confirmation and falsification. There are very simple forms of confirmation and falsification, sometimes based on the elementary level of reflexes. We are interested here only in the role of reasoning in those tasks. Importantly, confirmation and falsification may be represented as processes of formulating arguments and counterarguments. Obviously, reliable criteria enabling to decide if an argument is defeated by another argument are needed. Only external, intersubjective criteria are relevant as a tool of evaluation of defeats from the epistemological point of view.

To sum up: truth is a crucial attribute of contents of cognition (of beliefs); confirmation and falsification are used to evaluate truth of beliefs; reasoning, which respects some reliable and intersubjective criteria, is a tool of confirmation and/or falsification.

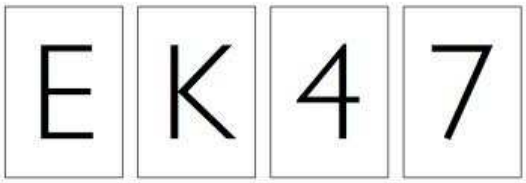
3. Reasoning: A Psychological Point of View

A natural outcome of our externalist and epistemological standpoint is the view that formal logic is rather important with respect to the understanding of human cognition. However, this stance has been challenged.

This section discusses two well-known psychological experiments, considered to be of great relevance to questions concerning the psychology of human reasoning. Both may be used to argue that formal logic does not account for the ways humans actually reason.

3.1 The Selection Task

The selection task, also known as Wason's task or Wason's selection task (see Wason [22],[23]), is constructed as follows. Subjects are shown four cards with numbers on one side and letters on the other. For example:



Subjects are then confronted with the following rule:

"If there is a vowel on one side, then there is an even number on the other side" (1)

Their task is to identify the cards which it is necessary to turn if one has to confirm or falsify the rule. In other words, they have to point to the cards (an only those cards) which have to be turned in order to settle the question if the rule applies to the displayed cards or not.

Surprisingly, the majority of subjects select E and 4 (usually around 45%) or E alone (35%). Only 5% of the subjects select what seems to be the correct answer, viz. E and 7 (data source: Stenning and van Lambalgen [21, p. 46]).[3]

It is easy to see the results as implying that actual human reasoning proceeds quite differently than by the rules of logic. More on this in section 3.3.

3.2 The Suppression Task

The suppression task (Byrne [8]) shows that additional premises may change subjects' inferences. For example, the inference

If she has an essay to write, she will study late in the library
She has an essay to write

Therefore: She will study late in the library

is made by 90% of the subjects: they use modus ponens correctly when A and $A \rightarrow B$ are the only premises. (Data source: [21, p. 181]) Now an additional premise is added, e.g.:

[3] One could reason as follows. In order to confirm the rule, it is necessary to rule out the possibility that it does not apply. The rule does not apply if there is a card with a vowel on one side and an odd number on the other. Therefore, it is necessary to check if there are cards of this sort. However, only E and 7 can be the possible falsifiers, therefore it is necessary to turn these.

If she has an essay to write then she will study late in the library
If the library stays open then she will study late in the library
She has an essay to write

This changes the situation dramatically. Only 60% of the subjects make the modus ponens inference and conclude "She will study late in the library". This means that many subjects do not make the valid inference when an additional premise is present: many subjects do not use modus ponens when the premises are $A \to B$, $C \to B$ and A). The experimental data has been used as a basis for denying the relevance of logic for human reasoning (See Oaksford and Chater [20] for example, where a probabilistic approach is advocated).

3.3 Comments

The experiments discussed in the previous section seem to suggest that even the simplest inference rules of propositional logic are not followed by a significant portion of subjects involved in reasoning tasks. Does this mean that logic is not relevant with respect to the understanding of actual human reasoning? (Or at least not as relevant as it was assumed to be?)

This subsection provides a preliminary answer. We put forward several straightforward remarks concerning the design decisions and the interpretations of results of both experiments.

First, the selection task is not a reasoning task, but a combinatorial task. We should explain the difference. Of course, some nontrivial reasoning is required for writing an essay or designing a hat or selecting an optimal option from a set of options. But logic is not and cannot be interested in a detailed description of such tasks (and many similar or substantially different tasks). Reasoning relevant from the logical point of view should be carefully described. Logic aims at *characterizing entailment*: if a set of premises is given, what is a correct conclusion?[4] A derivation of conclusions from premises is understood usually as a reasoning task.

Now, back to the selection task – it is a combinatorial task, but with insufficient, incomplete, unclear input. It is implicitly assumed that experimental subjects should know that the words "if .... then ...." are interpreted as material implication. Of course, this is an unjustified assumption.

[4] We will characterize some other domains of logic in Section 5.

As regards the suppression task, capability to apply modus ponens is tested. The essentially lower ratio of applications of modus ponens in the second case is interpreted as evidence that people do not use modus ponens automatically, in each situation, but human reasoning is dependent on the content and the context. This is true, but it is not an argument relevant with respect to logic and to applications of logic in human reasoning. Each logician or mathematician uses modus ponens only when true premises are given. In the second case of the suppression task there was a symptom that the premises may be not true. Consequently, the behavior of experimental subjects not applying modus ponens is quite rational (and, it could be said that the experimental subjects, who applied modus ponens blindly, did not reason carefully).

Note that the behavior of some experimental subjects was non-monotonic: they applied modus ponens to a subset of premises, but not to its superset. There are non-monotonic logical systems taking into account symptoms that some premises could be questioned and, therefore, some previous consequences cannot be derived anymore.

Let us return back to psychological experiments. Consider the following "experiment". Suppose that some experimental subjects should solve the following task. There are seven sheep on the meadow under the forest in the morning. Two sheep walked off the meadow later. How many sheep remained on the meadow?

97% of experimental subjects responded correctly. In the second round of the experiment, additional information is presented: Visibility is rather poor under the forest in the morning.

After receiving the additional information 34% of experimental subjects said that 5 sheep remained on the meadow, 12 % that 4 or 6 remained and the rest that the task does not have a solution.

The conclusion of the experimenter was as follows: Counting is viewed by arithmetic as a content-independent procedure applied impartially and uniformly to every problem regardless of the content involved. Hence, one could reason, it is not an appropriate tool for humans in real conditions.

Of course, nobody designed such an experiment. Everybody knows that truths of arithmetic do not depend on abilities of people to count or on conditions, where arithmetic could or could not be applied.

The same holds for logic. Validity of logic does not depend on abilities of people or on conditions of applicability of logic to reasoning tasks. Abilities of people, success or failure of human behavior are interesting from the psychological point of view. However, such aspects are not interesting from the logical point of view.

3.4. Logic and Actual Reasoning

The mainstream notion of logic sees its subject-matter as rather distinct from "actual human reasoning". It is not the task of logicians to study mental processes inside humans when they reason. They even do not have to rely on popular beliefs about the correctness of specific inferences.

According to the mainstream view (which has its roots in Frege's anti-psychologistic attitude towards logic and the foundations of mathematics), logic is seen as dealing with the *criteria of correctness* of inferences. Correctness is often specified as *truth-preservation*. Given a set of premises, which propositions cannot fail to be true in case all the premises are true? In other words, logic deals with *consequence*. Its task is to come up with an appropriate *definition* of consequence and with its matching formal *models*. These usually come in form of a formal language with appropriate semantics, together with a consequence relation. This is a relation between sets of sentences of the formal language and sentences, defined either syntactically (in terms of inference rules) or semantically (in terms of the semantic structures).

Philosophically speaking, this picture is sometimes summed up by saying that logic is *normative*. The task of logicians is to come up with formal models of consequence which in turn prescribe what is to count as correct inference. Consequently, any actual inference deviating somehow from the pattern prescribed by the formal model is deemed incorrect.

We think that the mainstream view is correct in its emphasis of formal models of inference. However, the idea that "Logic" is somehow superior to "actual reasoning" is too simplistic.

First, there is no such thing as "Logic". Some inferences that are correct from the viewpoint of classical logic are not correct from the viewpoint of intuitionistic logic, for example. (The view sketched here is sometimes called 'logical pluralism', see Beall and Restall [4])

Second, the factual evidence that comes from the history of modern logic (in the 20th century) is overwhelming. Simply said, most of the "non-classical" logics that emerged during the previous century (and that keep on emerging until this day), have their *raison d'être* deeply rooted in a felt discrepancy between the "predictions" of a formal model of inference (usually classical logic) and (intuitions about) actual inferences.

Modal logics are a good example. Almost every beginning logic student finds the properties of material implication somehow awkward. The inference from p to $q \rightarrow p$, deemed correct by classical propositional logic, is seen as suspicious. To be more specific, the "if …, then …" (or "… implies…") of natural language behaves differently than the "$\rightarrow$" of classical propositional logic. Modern modal logic emerged from the need to provide a more

appropriate formal model of "if…, then…". C. I. Lewis (see Lewis [16], Lewis and Langford [17]), the key figure of its early modern history, thought that adding necessity is sufficient: 'Necessarily ($p \rightarrow q$)' was seen by him as the correct formal rendering of "*p* implies *q*". However, this formal model has its own discrepancies and soon more refined models were suggested, viz. the various relevance logics (Anderson and Belnap [2],[3] and Mares [19]).

This ambition of modern logicians to "keep up" with the intuitions about actual reasoning and usage of the "logical words" has many examples, viz. conditional logics, non-monotonic logics, etc.

To sum up, the relation of logic to actual reasoning is not as simple as it may seem. First, logic is not to be thought of as a single set of 'correct' rules of inference. A more appropriate view of logic is to see it as a discipline aiming at providing formal models of inference and inference-related concepts. Second, these models are strikingly diverse and most of them were born of the need to model actual reasoning more flexibly and appropriately.

An important consequence of this viewpoint is that the relevance of logic to actual reasoning cannot be conclusively refuted by pointing out that a particular logical system does not fit in with intuitions or experimental data. There is always the possibility of providing a more appropriate system.

4. Logic Strikes Back

An important recent defense of the relevance of logic for cognitive science is Stenning and van Lambalgen [21]. Their strategy is to assess the importance of subjects' interpretations of the reasoning tasks, such as the Selection task or the Suppression task („reasoning *to* an interpretation").

Their claim is that after the interpretation has been settled, one may proceed to a formal model of the subjects' responses ("reasoning *from* an interpretation"). Stenning and van Lambalgen argue for the prominence of non-monotonic logics as a model of reasoning. Special attention is devoted to an interpretation and formalization of conditional sentences (rules) with exceptions.

5. Logic and Human Reasoning

This section outlines the reason why the logical point of view and methods of logic are indispensable for the understanding of reasoning, and hence for the understanding of human cognition. First, a possible way how logic as a

scientific field evolved from sophisticated human reasoning is sketched. After that some logical systems are discussed from the viewpoint of relevance for understanding of reasoning and cognition. The systems are presented in a somewhat sketchy manner, as a thorough exposition is not the aim of this chapter.

5.1. Human reasoning and logic

We shall argue that a logical representation of reasoning is a natural result of the cultivation of human cognitive capabilities and of the attempts to understand and describe our reasoning.

Our starting point is counter-argumentation via the search for counterexamples. If somebody wants to show that the arguments of his opponent are wrong, she may try to construct a similar flow of claims (sentences) which leads from true premises to a false conclusion. There is an analogy to attempts to falsify general statements. An obvious procedure is to find a special case for which the general statement is not true.

There is a nice example in the history of human thought. Socrates mastered the art of counter-argumentation and constructing of counterexamples as a tool of rational dialogue, most importantly as a tool of uncovering the falsity of someone's beliefs, as a tool of supporting our knowledge via arguing against unsupported claims. His influence leaded through Plato to an invention of a logical system by Aristotle.

The step from counter-argumentation to logic is simple. First, we emphasize that the construction of counterexamples entails a shift from the content of sentences to their form – counterexamples are of the same form as the attacked sentences. Second, the focus is shifted from the construction of counterarguments to finding ways of reasoning which are immune from counterarguments. In other words, the attention of (not only Aristotelian) logic was and is focused on *truth-preserving* schemes of reasoning. Remember the well known syllogism: if each A is B and each B is C, then each A is C. It is impossible to find a counterexample (a substitution of some notions, names of classes) such that each A is B, each B is C, but there are some A, which are not C. On the other hand, you can find a counterexample to the following form of reasoning: if some A are B, some B are C, then some A are C. The first scheme of reasoning preserves truth (it leads necessarily from true premises to a true conclusion). The second scheme is obviously not truth-preserving.

An important feature of correct human reasoning is an ability to preserve truth of basic postulates, facts and starting points in the flow of

reasoning to the truth of consequences. A fundamental relevance of logic for cognitive science is based on that observation.

We have to reflect the development of logic from Aristotelian times to the state, where a rich variety of logical systems and ways how to do logic (Makinson [18]) is available. Schemes of reasoning uncovered by a logical system preserve truth, if logical constants (each, some, if – then, possibly, etc.) are understood in the way specified by the logical system. Reasoning to an interpretation, as understood by (Stenning and van Lambalgen [21]) is a procedure leading to a selection of an appropriate understanding of logical constants for a given reasoning task. Moreover, a need to specify and to model in an abstract way new logical constants or some new meaning of a logical constant leads often to a new logical system and to a new option for a reasoning to an interpretation.

However, the characterization of the variety of reasoning procedures and styles is not exhausted by simple truth preservation. Different types of schemes of truth-preserving reasoning provide a characterization of different forms of *deduction*. We have to mention also *hypothetical* reasoning also.

Fortunately, (at least some of) people reason even if they do not have only true premises at their disposal. This kind (more precisely, a class of kinds) of reasoning is studied intensively in artificial intelligence. Non-monotonic reasoning and defeasible reasoning are the terms used in artificial intelligence. We will use the term 'hypothetical reasoning' and note that it can be (and is) described in many different ways. We sketch only a simple characterization here.

When people reason hypothetically, they consider a set of defeasible assumptions. In general there are some incompatible assumptions in the set. Some assumptions attack other assumptions (via their consequences). Usually, it is not possible to speak about the correctness of an isolated assumption. It is more productive to consider sets of assumptions and to check whether they are defended against the attacks of some (counter)assumptions. A conflict-free set of assumptions S is *admissible* if it counterattacks each attack against each member of S. This is the basic idea of Dung [10], where this notion was introduced precisely. In Bondarenko et al. [7] it was adapted for assumption-based frameworks and applied to a characterization of default reasoning in the frame of various non-monotonic formalisms.

Different kinds of non-monotonic logic, defeasible logic, argumentation frameworks, logic programming etc. study hypothetical reasoning and various types of sets of admissible assumptions. Some argumentation semantics, based on notions of conflict-freeness and admissibility are discussed later in this section.

We conclude this subsection as follows. There are two important features of human reasoning – preservation of truth and accepting of admissible sets of assumptions. The *logical point of view* can be characterized (at least for the aims of this chapter) as focused on truth-preservation or on admissibility of assumptions (arguments). *Methods* of logic consist in an abstraction from the content of pieces of knowledge or sets of sentences, in the construction of some symbolic, formal languages[5], which enable an abstract and general treatment of a kind of reasoning.

It is important to note that the method enables a highly detailed description of reasoning and that thanks to this level of details it is possible to construct computational models of reasoners and to implement them in real applications.

Epistemological point of view was characterized by an abstraction from mental processes and by an emphasis on reliable, intersubjective criteria. Logical point of view and methods of logic contribute to understanding and modeling correct reasoning in accordance with the principles mentioned above.

5.2 Basic logical systems

We are aiming to show how a cognitive stance may influence a construction of a logical system.

The classical two-valued logic is based on a platonic view of the world: individuals have or have not some properties, a situation or an event occurs or does not occur, a sentence is true or false. Tertium non datur (the law of excluded middle) is a logical expression of this basic attitude. Similarly, contradictions are not allowed. Thus, a proposition that there is an object which some property may be proved if it is demonstrated that an assumption about the non-existence of such an object leads to a contradiction. This stance is sometimes characterized by the slogan that logic is a set of features of the world and the (!) correct reasoning consist in discovering those features.

Constructivist logic evolved as an opposition to the kind of logic characterized in the previous paragraph. Existence of an object satisfying a property can be proved only if the object with that property is constructed. According to a branch of constructivism called intuitionism, methods of construction are based on natural human intuition. Thus, correct reasoning consist in following given capabilities of our mind. Of course, constructivist

[5] An objection against logic is that it constructs some strange artificial languages. We hope that it is only a marginal stance. Nobody criticizes physics or engineering because of their use of an artificial language. Similarly, logic discovers fundamental knowledge thanks to an art of abstraction and focusing on principles.

logical systems are sharply separable from the not too clear philosophical motivations and the difference with respect to classical logic may be characterized precisely.

Another stream of logical systems deviating from the classical logic enlarges the set of truth values. Truth and falsity are not the exclusive values; the third value was introduced first. Systems with infinite sets of truth values were constructed. Fuzzy logics, which emphasize that the borders between classes, properties etc. may not be sharp, are in fact multi-valued logics.

Finally, we mention the Kripkean semantics. This style of semantics provides a characterization of such propositions, where a direct assignment of a truth value is not appropriate. Epistemic logic, presented below, is one example.

5.3 Epistemic logic

One of the most prominent current approaches to modeling information and cognition is epistemic logic. Epistemic logic dates back to the seminal works of von Wright [24] and Hintikka [14]. This subsection offers a sketch of its basics.

The language of epistemic logic extends the language of classical propositional logic by a family of knowledge operators K_i, where i ranges over some set of agents G: $K_i\, p$ is read 'agent i knows that p'.

Epistemic models for a set of agents G are structures $M = (W, \{R_i\}_{i \in G}, V)$, where W is a non-empty set, every R_i is a binary relation on W and V is a valuation, i.e. a function from the set of propositional atoms to subsets of W. Informally, W is thought of as the set of epistemic alternatives or possible worlds. However, it is usual to refer to them in a more neutral manner as 'points'.

Next, R_i is an epistemic indistinguishability relation for the agent i: R_ixy iff i cannot distinguish between points x, y. To be more specific, i cannot distinguish between x and y if she does not have access to information that would render one of the points as obviously incorrect. For example, if I do not know whether it is sunny in London, then I cannot distinguish between any x containing the fact that it is sunny in London and any y containing the fact that it is not sunny there. In most applications, the indistinguishability relations are assumed to be equivalence relations, i.e. reflexive, symmetric and transitive.

Truth of formulas is relative to points: most importantly, K_iA is true at a point x iff A is true at every y such that R_ixy. Hence, i knows that A iff A holds at every epistemic alternative. This is in line with our intuitions – if I know that A, then points with non-A are obviously not sound epistemic alternatives.

Epistemic logic clarifies several somewhat involved scenarios such as the muddy children puzzle and is often used in computer science (see Fagin et

al. [11]). One of its positive features is the ability to represent knowledge of agents about the knowledge of other agents in a clear way.

However, epistemic logic also has a number of counterintuitive features. For example, if a formula A holds at every point in some model M then K_iA also holds in every point in M. (The reason is simple: the set of 'i-reachable' points from any given x is obviously a subset of W). Read informally, this means that every agent i knows every valid formula! Moreover, the formula $K_i (A \rightarrow B) \rightarrow (K_iA \rightarrow K_iB)$ is valid in every M. In other words, knowledge is closed under modus ponens. Together with the knowledge of valid formulas, this entails that knowledge is closed under valid implications. As a special case, knowledge is closed under classical consequence.

However, it is obvious that this is an idealized situation. In many situations agents do not know every consequence of their knowledge. For example, they might didn't perform the needed inference steps or they lack the computational resources to do so. This obvious discrepancy between our intuitions about the knowledge of real agents and the 'predictions' of epistemic logic is known as *the logical omniscience problem*.

The standard answer is to distinguish between *implicit* and *explicit* knowledge. Explicit knowledge is seen as a body of consciously accepted and confirmed information. On the other hand, implicit knowledge is the body of logical consequences of explicit knowledge. It is acknowledged that the K_i operators represent implicit knowledge.

However, the issue of modeling explicit knowledge remains interesting. The literature offers several approaches (see Fagin et al. [11, ch. 9]).

5.4 Dynamic Epistemic Logic

The kind of epistemic logic described in the previous section is often described as being static. It models information states of agents by the indistinguishability relation, but these are information states at a given time.

However, an important feature of knowledge and cognition is its *dynamics*. We often revise our beliefs in the face of new evidence, or supplement our information as a result of observation and communication.

Modern logic offers several formal models of information dynamics. An important contribution is the *belief revision theory*, see Alchourrón, Gärdenfors and Makinson [1]. It is a refined model of belief change with several distinguished modes of change. The first one is simple *expansion*: sometimes we add to our beliefs new information that does not conflict with our previous beliefs. The second one is *contraction*: sometimes we abandon our beliefs for various reasons. The third, and perhaps the most important one, is

revision: sometimes new beliefs have to be added which contradict some of the previous beliefs. Now the problem is to restore consistency. Which beliefs is it best to abandon? The belief revision theory offers an interesting answer, which is, however, beside the scope of this chapter.

A different model of information dynamics is the *public announcement logic* (see van Ditmarsch, van der Hoek and Kooi [9, ch. 4]. This is an extension of the basic epistemic logic of section 5.3. Importantly, the basic epistemic language is extended by a modality $[A]$ for any formula A. Now the formula $[A]B$ is read 'After a truthful public announcement of A, B is the case.' Semantically, $[A]B$ is true at a point x iff A is true at x and B is true at x with respect to a model where every point that does not make A true has been deleted.

Public announcement logic is a simple formal model of communication and public observation with many interesting applications. For more information on the formal models of epistemic dynamics, see van Benthem [5].

5.5 Admissibility semantics

This subsection provides a cursory view on a formalization of defeasible reasoning. The formalization is not our primary goal. More interesting is to show that a formal, logic-based approach to a description of hypothetical reasoning is possible and fruitful. We describe a very simple, but elegant abstract argumentation framework by Dung [10].

An abstract argumentation framework AF is a pair (A, R), where A is a set (of arguments) and R is a binary relation on A. If a pair of arguments (a, b) is in R, it is said that a attacks b. Notice that nothing is supposed about the structure of arguments. Similarly, no details about the attack relation are given.

Dung accepted an essential decision – a status of an argument may be specified reasonably only with respect to a set of arguments. Hence, an argument a is *acceptable* with respect to a set of arguments S, if and only if for each argument b attacking a there is an argument c in S attacking b. A set of arguments S is *conflict-free* iff there is no pair a, b in S such that a attacks b. We already know that a conflict-free set of arguments S is *admissible* iff each attack of an argument b against an argument a in S is counterattacked by S, i.e., there is an argument c in S such that c attacks b.

Some argumentation semantics (mappings from AF to sets of sets of arguments) are presented below. A conflict-free set of arguments S is a *preferred extension* of AF iff it is a maximal (with respect to the subset relation) admissible set of arguments from AF. S is a *stable extension* of AF iff each argument not in S is attacked by an argument in S. F, a *characteristic function*

of *AF,* assigns to a set of arguments *S* the set of all arguments acceptable with respect to *S*. The (only) *grounded extension* of *AF* is the least (with respect to the subset relation) set of arguments *G* such that *F(G) = G*. It is said that *G* is the least fixed point of *F*. The existence of the least fixed point of *F* follows from the monotony of *F*.

Other semantics of abstract argumentation frameworks were proposed since Dung's seminal paper and logical research in argumentation proceeded also from abstract frameworks to structured frameworks, where the structure of arguments is interesting and also other logical aspects of argumentation are considered.

It is important to notice that logical research of argumentation abstracts from such problems or claims as: persuade your partner in a dialogue that your opinion is his opinion etc. Attention of logics is focused only on arguments and their relations.

Other, yet more important remark is the following. Semantics of other non-monotonic formalisms can be expressed in terms of argumentation semantics. Let us consider logic programs under the stable models semantics (Gelfond and Lifschitz [12]). Nowadays, the term 'answer set semantics' is used more often and answer set programming (ASP) became a leading paradigm in implementation and theoretical research of knowledge representation and reasoning in artificial intelligence. It was shown already in [10] that logic programs may be represented as argumentation frameworks and some semantics of logic programs correspond to argumentation semantics. Most importantly, from the viewpoint of ASP, stable models correspond to stable extensions. An early extensive work studying relations of non-monotonic formalisms and argumentation semantics is Bondarenko et al. [7].

In order to close this subsection: it was shown that hypothetical (defeasible, non-monotonic) reasoning may be viewed as an argumentation framework, where some assumptions play a role of arguments, conflicts between sentences are considered as attacks between arguments and solutions of conflicts are specified by some semantics. However, we did not choose the argumentation frameworks with an intention to present it as a leading paradigm in the research of hypothetical reasoning. A final remark - hypothetical reasoning is closely connected to a representation of dynamic aspects of knowledge and, consequently, to belief change (updates and revisions).

6. Conclusions

An activity highly relevant for cognitive science is, e.g., to present a field of knowledge, which is assessed as unquestionable and, moreover, to develop a

language and a method supporting that goal. An unprecedented intellectual boom connected to the mathematical logic research in the beginning of 20th century was oriented towards that kind of goals – to provide firm and unquestionable foundations of mathematical knowledge.

It is well known that the deep results by Goedel, Turing, Tarski and others rendered this ambition hopeless. However, a more modest characterization given by Barwise and Etchemendy [6], who claim that mathematical logic is an idealized presentation and communication of mathematical results seems to be interesting from the cognitive science point of view, too.

It was emphasized repeatedly in this chapter that real-world stimuli contributed to the development of new systems and kinds of logic. A role of artificial intelligence was noticed in this context, too. We shall conclude by an interpretation of the role of logic in computer science (Halpern et al. [13]) from the viewpoint of cognitive science. Logic is proven to be an appropriate and effective tool for development of theories and constructions in computer science (program specification and verification, programming languages research, databases research, complexity theory, multi-agent systems, automated design verification, knowledge representation etc.). This fact is contrasted with respect to the essentially less important role of mathematical logic in contemporary mathematics. An application of languages and systems intended for idealized modeling of some aspects of the world and some kinds of entailment relation to a successful description of domains interesting for computer science and for reasoning about those domains could be interpreted as a success story from the cognitive science point of view; certainly, cognitive capabilities are needed for such applications.

Acknowledgments: Work of the first author was supported by the grant 'Semantic models, their explanatory powers and applications' (VEGA 1/0046/11). The work of the second author was supported by the grant VEGA 1/1333/12.

References

[1] Alchourrón, C.E., Gärdenfors, P. and Makinson, D.: On the Logic of Theory Change. *Journal of Symbolic Logic* **50** (1985), pp. 510 – 530.

[2] Anderson, A. R., Belnap, N. D.: *Entailment*, volume 1. Princeton University Press, Princeton, 1975.

[3] Anderson, A. R., Belnap, N. D.: *Entailment*, volume 2. Princeton University Press, Princeton, 1992.

[4] Beall, J. C., Restall, G.: *Logical Pluralism*. Clarendon Press, Oxford, 2006.

[5] van Benthem, J.: *Logical Dynamics of Information and Interaction.* Cambridge University Press, Cambridge, 2011.

[6] Barwise, J., Etchemendy, J.: Computers, visualization, and the nature of reasoning. In T.W. Bynum and J. H. Moor (Eds.) *The Digital Phoenix: How Computers are Changing Philosophy*. Blackwell, Oxford, 1998, pp. 93 – 116.

[7] Bondarenko, A., Dung, P. M., Kowalski, R. A. Toni, F.: An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence* **93** (1997), pp. 63 – 101.

[8] Byrne, R. M. J., Suppressing valid inferences with conditionals. *Cognition,* **31** (1989), pp. 61 – 83.

[9] van Ditmarsch, H., van der Hoek, W., and Kooi, P.: *Dynamic Epistemic Logic*. Springer, Dordrecht, 2008.

[10] Dung, P. M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and *n*-person games. *Artificial Intelligence* **77** (1995), pp. 321-357.

[11] Fagin, R., Halpern, J. Y., Moses Y. and Vardi, M. Y.: *Reasoning about Knowledge*. MIT Press, Cambridge (MA), 1995.

[12] Gelfond, M and Lifschitz, V.: The Stable Model Semantics for Logic Programming. *ICLP/SLP* (1988), pp. 1070 – 1080.

[13] Halpern, J.Y., Harper, R., Immerman, N., Kolaitis, P.G., Vardi, M.Y., Vianu, V.: On the unusual effectiveness of logic in computer science. *Bulletin of Symbolic Logic* **7** (2001) pp. 213 – 236.

[14] Hintikka, J.: *Knowledge and Belief: An Introduction to the Logic of the Two Notions*. Cornell University Press, Ithaca, 1962.

[15] Kováč, L.: Information and Knowledge in Biology: Time for Reappraisal. *Plant Signaling and Behavior*, **2** (2007), pp. 65 – 73.

[16] Lewis, C. I.: *Survey of Symbolic Logic.* University of California Press, Berkeley, 1918.

[17] Lewis, C. I., Langford, C. H.: *Symbolic Logic*. Century, London, 1932.

[18] Makinson, D.: Ways of doing logic: what was different about AGM 1985? *Journal of Logic and Computation* **13** (2003), pp. 3 – 13.

[19] Mares, E.: *Relevant Logic. A Philosophical Interpretation*. Cambridge University Press, Cambridge, 2004.

[20] Oaksford, M. and Chater, N.: The probabilistic approach to human reasoning. *Trends in Cognitive Sciences* **5** (2001), pp. 349 – 357.

[21] Stenning, K., van Lambalgen, M.: *Human Reasoning and Cognitive Science*, MIT Press, Cambridge (MA), 2008.

[22] Wason, P. C. Reasoning about a rule. *Quarterly Journal of Experimental Psychology,* **20** (1968), pp. 273 – 281.

[23] Wason, P. C.: Reasoning. In B.M. Foss (Ed.): *New Horizons in Psychology I*. Penguin, Harmondsworth, 1966.

[24] Von Wright, G. H.: *An Essay on Modal Logic*. North-Holland Publishing Company, Amsterdam, 1951.

Evolutionary and Genetic Fuzzy Systems

Ján Vaščák[1]

Abstract. In the last twenty years evolutionary fuzzy, or more known genetic fuzzy, systems have been developed to a large extent of numerous designs and types. They approved efficacy and quality also in comparison to other adaptive systems like e.g. neural fuzzy systems. Principally, there are two basic groups of systems built on the theories of evolutionary and fuzzy systems — own evolutionary fuzzy systems, where evolutionary algorithms are used for knowledge acquisition because fuzzy systems are not able to self-learn and fuzzy evolutionary systems, where fuzzy logic performs some auxiliary functions for evolutionary systems. Therefore, the structure of this chapter has three basic parts, where main approaches are explained. Firstly, it deals with adaptation possibilities of fuzzy systems and basic evolutionary methods for this task. Secondly, some aspects of enhanced approaches connected with the first part are described and thirdly, several examples of fuzzy evolutionary systems accomplish the introduction into this area of computational intelligence.

1 Introduction

Fuzzy logic has found since the last three decades a huge number of applications in various areas of everyday life. Without any doubts it is the most spread representative of artificial intelligence. It enables to describe many problems, which are difficult to be mathematically formulated or loaded by inaccurate and incomplete information. As the most successful praxis-oriented product of fuzzy logic there is the well-known *fuzzy controller*, which is a central part of a more general notion the so-called *fuzzy inference system* (FIS). These systems are easy to understand and hereby to design comparing to other approaches. Modularity and possibility to incrementally build knowledge are further advantages of FISs for knowledge engineers. Finally, usual robustness of solutions consisting in their generalization ability as well as in processing uncertain information favour these means over many others in a broad

[1] Technical University of Košice, Department of Cybernetics and Artificial Intelligence, Center for Intelligent Technologies, Letná 9, 042 00 Košice, Slovakia, E-mail: jan.vascak@tuke.sk

spectrum of use. As FISs are general approximators of any analytical function so they are convenient means from the mathematical viewpoint, too.

However, a conventional FIS needs an expert or a group of experts for constructing its *knowledge base* (KB), which is mostly in form of *fuzzy production rules* as *IF <antecedent> THEN <consequent>*. Already in the early 1990s a need for adaptation means being able to automatically create KBs came into foreground of the research because of many communication and social problems connected with humans as it was firstly encountered at *expert systems*. Also their eventual lack played an important role for developing this area.

As seen in fig. 1, mainly *artificial intelligence*, especially *computational intelligence* as its part, offers a whole variety of adaptation possibilities. The two most utilized groups of FIS adaptation methods are based either on neural networks or evolutionary, especially genetic, algorithms. Although the first group, i.e. the so-called *neural fuzzy systems*, is thanks a very numerous community of neural networks experts more developed and neural networks can be efficiently used also for optimization (basic principles using neural networks are exhaustively described e.g. in [19]) we will further deal mainly with *evolutionary* (EAs) and *genetic algorithms* (GAs). Such hybrid systems connecting EAs or GAs with FISs, whose role is constructing or optimizing FIS, are mostly denoted as *genetic fuzzy systems* (GFSs), including EAs, too.

Similarly, as Lin and Lee divide *neural fuzzy systems* (NFSs) into *neuro-fuzzy* and *fuzzy-neuro* systems [19] there is such a division possible also in the area of GFSs. Intrinsic meaning of GFSs is only for systems, which use GAs (EAs in general) for constructing KBs whereas in *fuzzy genetic systems* (FGSs) fuzzy logic is used for improving functionality of GAs. However, such fuzzy 'improved' GAs again serve usually for constructing KBs and so they are often a part of GFSs (see an application in [24]). This chapter deals mainly with GFSs but in sect. 5 also some applications of FGSs are described.

FISs were originally represented by rule-based systems and the GFSs research was focused just on these systems during the 1990s. An overview of adaptation methods used for rule-based FISs as well as some application examples are summarized in [5]. However, meanwhile further concepts based on fuzzy logic appeared, e.g. fuzzy relational rules or use of fuzzy logic in constructing regression models and clustering, which can be solved by GAs. Besides, new requirements arose like fulfilment of multiple criteria, especially trade-off between accuracy and interpretability of obtained KBs. Of course, the task of handling high dimensional data sets of complex optimization tasks grows steadily on importance.

The aim of this chapter is to offer an overview about mutual interconnections between GAs (EAs) and FISs. From this reason sect. 2 describes learning aspects of FISs and features possible approaches if using GAs or EAs. Sect. 3 deals with

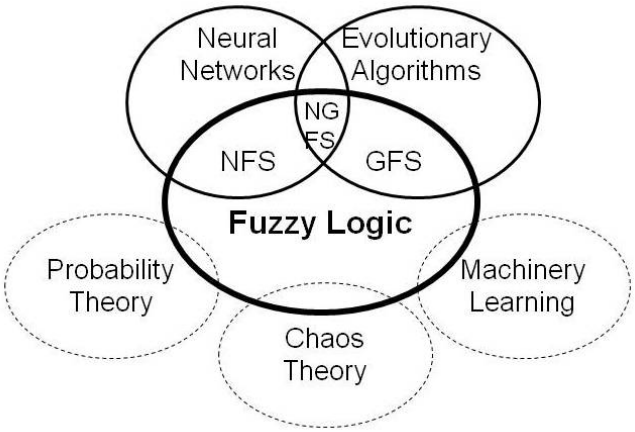


Figure 1. Hybridization possibilities of FISs with other means of artificial intelligence; NGFS — Neural Genetic Fuzzy Systems*, NFS —* Neural Fuzzy Systems*, GFS —* Genetic Fuzzy Systems*.*

adaptation of conventional rule-based FISs whereas the subsequent sect. 4 describes further advanced methods, which have been developed mainly in the last decade and regard some special aspects important for high quality KBs. Opposite to previous sections, sect. 5 describes some examples of using fuzzy systems for enhancing the performance of evolutionary systems. Finally, in sect. 6 some concluding remarks and outlooks into future are sketched.

2 Fuzzy Inference Systems — Ways of Adaptation

Although there are possible several kinds of knowledge representations in FISs but in most cases it is the rule-based one. Thus we will deal only with this knowledge representation in this section. A rule-based FIS is comprehensible indeed but on the other hand side such a kind of representation is heterogeneous because knowledge is stored in two incompatible algebraic structures: (a) base of *membership* and *scaling functions* and (b) rule base (RB). These structures have to be handled separately and adaptation methods will be different, too. In the case of FIS we consider *parametric* and *structural* learning, respectively. Parametric learning tries to find optimum numeric values for parameters of *membership* and *scaling functions*, whose structures, i.e. types like triangular, Gaussian, etc. are known in advance. Whereas rules divide a given *state space* into clusters, which are represented by individual rules. In other words, the rules define the structure of such a division, see fig. 2. Hence the principal task of each structural adaptation method is to transform a given

state space division into a form of RB. Each transformation process can be described also as a function and individual rule adaptation methods differ each other just by their transformation functions.

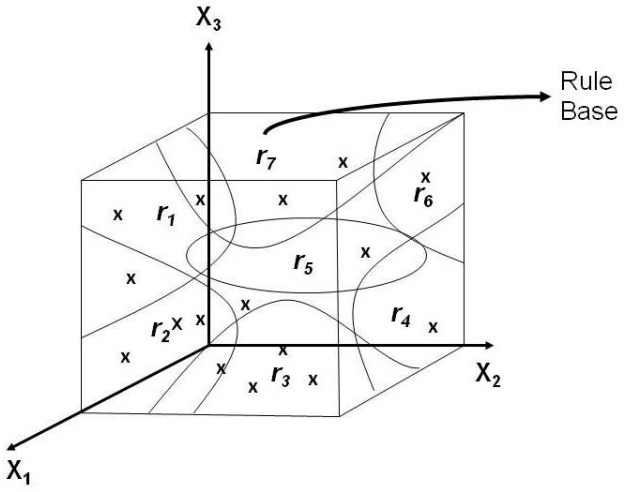


Figure 2. An example of state space division into clusters by rules r_i; x — samples.

As fuzzy inference rules are composed of uncertain linguistic terms (values), which are described just by *membership functions* (MFs) μ, it is apparent that parametric learning should precede the structural one. However, this process is just reverse to a human learning process. At first a human learns a basic structure of a given problem and after that he/she searches parametric values. The consequence of this fact is that some parameters like for example number of linguistic values must be determined manually in advance. Hence basic learning methods are not fully automatized and they require some manual interventions, too.

Another division of adaptation approaches for FIS is whether its structure is preserved, i.e. the so-called *direct* or *indirect* adaptation, where the conventional mechanism of FIS is transformed into another structure as its part. Almost all approaches based on neural networks belong to indirect adaptation because FIS is transformed to a special neural network, which is functionally equivalent to a FIS [19]. In this case it can be difficult to extract a KB for a FIS but mostly the indirect adaptation is quick and some methods enable to design a KB incrementally and on-line. Other approaches represent mostly the group of the direct adaptation, whose basic structure can be depicted as it is seen in fig. 3. There is no need of a special method for knowledge extraction but these methods work only off-line.

We see that the structures of FIS as well as KB are preserved and the adaptation

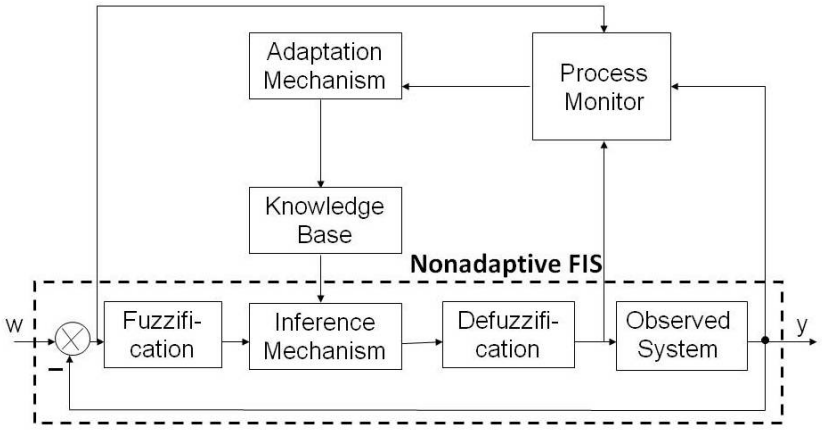


Figure 3. Structure of direct FIS adaptation; w — desired value.

part forms a superstructure over the conventional (nonadaptive) FIS. Usually there are two main modules in the adaptation part. The *adaptation mechanism* immediately calculates and performs changes in KB. However, to be able to calculate them a kind of information about KB quality is necessary. This is the task of the *process monitor*, which can be *parameter-based* or *performance-based*. The first type of the process monitor represents modeling of an observed system in reality. This type of monitoring is indeed efficient but creating a model is a difficult task. The second type of the monitor represents a quality evaluation, which in notions of EAs and GAs is the value of the fitness function and first of all genetic-based adaptation methods are related to *performance-based* monitors where the adaptation mechanism is responsible for generating candidate parameters of KB and selecting the optimum ones.

Finally, it is still necessary to explain differences between notions *adaptation, learning* and *tuning* because in the literature there are some discrepancies. The *adaptation* has a general meaning about the ability of a system to change its parameters or even structure in dependency on its environment. Thus for instance, in the conventional control theory each system owning this ability is adaptive. However, in artificial intelligence the adaptivity is always connected with learning. Here an adaptive system must be able to self-learn, which has many forms and levels of complexity. Therefore, the notion *learning* is often reserved for *structural adaptation*, i.e. creating and modifying rules whereas *tuning* is used in connection with setting-up parameters of membership and scaling functions, i.e. *parametric adaptation*. Further, the expression *adaptation* will comprise both *learning* and *tuning*, sometimes used also the term *self-organizing*, or it will be used generally without

any differences [9].

3 Adaptation of Rule-based Fuzzy Inference Systems Using Genetic Algorithms

Adaptation methods of this kind of FISs can be divided into three basic groups depending on parts of KB, which are adapted:

1. parametric adaptation of membership and scaling functions,
2. rule base learning,
3. adaptation of a complete KB.

The first group relates to tuning only because RB has to be defined in advance. The membership as well as scaling functions are suited during the adaptation process exactly to match a given RB. If some changes will be done in RB it will be necessary to start a new adaptation. This kind of adaptation processes is the simplest one and it does not require any special modifications of a conventional GA. The process monitor (see fig. 3) represents a fitness function, which evaluates the performance quality of a given FIS, where we can see generally three inputs (exceptions are possible) into the monitor, i.e. values of inputs and outputs of the observed system as well as deviations from the desired value w. For instance, if we adapt a fuzzy controller for any control problem a fitness function will be surely dependent on the control deviation $w-y$. Inputs (actuators) and output values can be helpful for determining dynamics of such a system.

The adaptation mechanism is created by a GA itself. All function parameters are encoded into one individual, which mutually competes with others in the frame of a population. Consecutive processing of a given GA should improve parameters of given functions, whose types and numbers need to be given in advance, too. Hence there is a fixed number of adapted parameters, i.e. the length of a given chromosome is also fixed. There are possible both binary and real coding schemes as depicted in fig. 4. If we have n inputs and m outputs and the i^{th} linguistic variable has ik MFs denoted as μ_{Ii}^{ij} for inputs or μ_{Oi}^{ij} for outputs (j as index for linguistic values $j = 1,\ldots,ik$) and considering np as number of parameters for each function (we suppose one uniform type) then the total code length L for tuning MFs will be

$$L = np.\sum_{i=1}^{n+m} |ik|, \tag{1}$$

where $|ik|$ is the number of linguistic values defined on the linguistic variable i. If we take into consideration also scaling functions then the chromosome code will be extended by the sum of their numbers of function parameters. However, scaling is in the field of fuzzy logic not accepted well because already small changes of scaling factors may affect the stability of a controller but using proper combination of MFs and rules can always substitute the need of scaling. Scaling also deforms the real meaning of obtained linguistic values, which is another reason for their refusal.

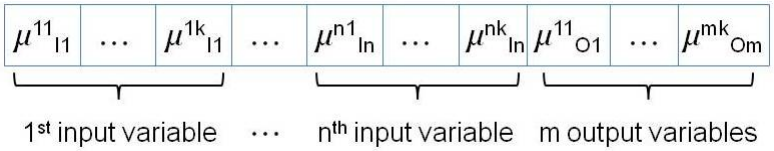


Figure 4. An example of real coding for a chromosome representing adaptation of all MFs.

Other two groups of adaptation methods require more sophisticated modifications of GAs because there will be used more complicated structures of coding and changes will meet also operators of crossover. GAs (EAs) alone will be only elements of these extended structures, which are only roughly defined thus enabling a variety of modifications. In other words, we will not speak about exact methods or algorithms but rather about methodologies or approaches. Basically, regarding RB encoding there are two main approaches: *Pittsburgh* and *Michigan*, which will be described in next sections. Combining approaches for RB constructing and conventional GAs for setting-up parameters of membership and scaling functions as two independent parts we can obtain a complete design of KB.

3.1 Michigan Adaptation Approach

This approach [14] utilizes some techniques of *machine learning* as well as conventional GAs. Its roots lie in the so-called *learning classifier system*, whose output is a set of rules (classifiers) actively interacting with external environment. The system communicates with its environment by input and output messages as information from sensors and to actuators, respectively. The rules are classified by their *strengths* and compete mutually to acquire more strengths, which enable them to be performed (fired) with higher priority. Besides, GA is used for generating new rules, which are offsprings of the rules with higher strengths or, in our context, fitness.

The general structure of Michigan approach consists of three basic modules, see fig. 5:

– *performance system* (PS),

– *credit assignment system* (CAS),

– *rule (classifier) discovery system* (RDS).

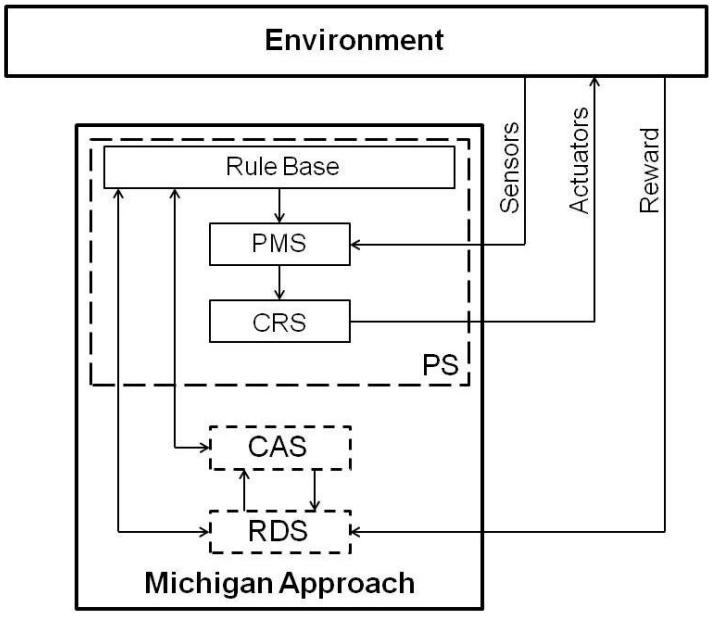


Figure 5. Basic structure of Michigan-type adaptation system.

PS is not only a RB but it also provides means for interactions of rules with the environment, which are performed by message processing. It is equipped with sensors on the input side and actuators on the output side. PS utilizes a specific inner language, which processes all messages in this module. The sensed states of the environment are translated into messages in the form of an alphabet of the inner language. These messages are processed in a set of rules in such a manner that those rules are activated, whose antecedents match with input messages. Subsequently, the messages describing the action parts (consequences) of the activated rules are sent to the output interface, where they are transformed into actions. Thus a set of activated rules is chosen in each computational cycle. Specially important there is the so-called *pattern-matching system* (PMS), which is similar to a rule-based inference system and its task is to identify matched rules. Besides, there is a *conflict resolution system* (CRS), which tries to discover any conflicts, inconsistencies or redundancy among rules.

Even if PS interacts with dynamic environment and reacts to its changes but there is not performed any knowledge acquisition. Therefore, the other two modules provide means for learning the RB. The first module the CAS uses a method of the so-called *bid competition* being inspired by auction trade. Matched rules offer a part of their strengths. After such an auction the winner(s) pays (pay) this part of its (their) strength(s) for the possibility to be fired. If the fired action is rewarded then the reward will enhance strengths of all fired rules otherwise they will lose. In such a manner a hierarchy of rules with different strengths is created in RB. Some rules are gradually removed or at least weakened and some others are strengthened until the environment is changed. There is a number of methods, which realize CAS, e.g. the *bucket brigade algorithm* or *profit sharing plan* [6]. As CAS contributes to the discovering of rule conflicts considerably it is often connected together with CRS as a CA/CR system.

CAS acts as a rule filter. After a certain time RB is reduced to a small set of rules with high strengths or fitness. Such an approach converges to solutions in a subspace, where probability of optimal solutions is low. To stimulate the system to search also in other parts of the space of solutions it is necessary to add new rules when the CAS reaches a steady state. This is the task of RDS, which utilizes GAs for this purpose. Mainly the rules with high strengths are selected for parents and GA will generate a population of offsprings, which will be after that included into RB and using CAS the strengths will be calculated for them. The only difference from a GA is that the selection of rules is performed in CAS and not in RDS.

This whole learning process of creating new rules is performed cyclically, where some additional inner cycles exist, too. It will be stopped if no new rules are created, which can happen only if there are no new significant changes of the environment. The pseudocode of the basic algorithm is shown in fig. 6.

The mentioned process of Michigan approach was described very roughly, where only basic processes were noticed. There is a number of various modifications, e.g. [3, 27] but all of them have a common characteristic feature, which differs from Pittsburgh approach described in sect. 3.2. Systems based on Michigan approach represent their individuals as rules. Thus a population is equivalent to a RB, which is the result of competitions among individual rules (individuals).

3.1.1 Iterative Rule Learning Adaptive Approach

Although the *iterative rule learning* (IRL) approach is derived from the mentioned Michigan one, because it is based on the same type of encoding, i.e. rules correspond to individuals, still it is mostly characterized as an independent approach (the third one). Its main motivation is to simplify Michigan approach and also to embed some positives of Pittsburgh approach (sect. 3.2) [35].

```
Initialize starting RB ← random creation with equal strengths
do while termination criteria are not fulfilled
    Sense inputs from the environment and encode them
    Call PMS to determine the set of matched rules
    Call CRS: to detect conflicts among matched rules &
                & to select a subset of active rules
    if some inconsistencies occurred
      Call CRS
    else
      Send actions to actuators
    end if
    if reward is obtained
      Call CAS to distribute reward
    end if
    if CAS reached a steady state
      Call GA to generate new rules
    end if
end do
```

Figure 6. Pseudocode of Michigan-type adaptation system.

GA is also used for generating and selecting rules but the principal difference to Michigan approach is a fact that in each iteration step only one rule with the best strength is selected and added to the resulting RB, which is held separately from the learning system. After that it is possible to exclude all examples from the data set that are covered by this rule. In such a manner the data set is consecutively pruned until it will be cleared, which is the final stopping criterion for the learning process because the selected rules cover completely a given data set.

Such an approach is very simple it requires only defining a criterion (or more criteria) for selecting the best rule for the final RB and eventually other stopping criteria being able to asses the completeness of RB. However, there is also a significant drawback. The rules (individuals) are evaluated individually regardless of a fact that rules create a unity and they cooperate in reality. Rules are separated, which leads to a redundant final RB showing over-fitting. From this reason a second stage of *post-processing* is still needed, which would simplify RB removing any redundancy.

IRL approach thanks to its simplicity and ability to use various selection criteria has found a wide range of use and at present it is one of the most researched means in this scientific area. Especially popular it is in multi-objective problems, where mainly a trade-off between accuracy and interpretability (simplicity, comprehensibility) of the proposed RB grows on importance. The most known IRL systems in this area are MOGUL [7] and SLAVE [11].

3.2 Pittsburgh Adaptation Approach

From the viewpoint of encoding the information about rules there is still another possibility. An individual will represent a whole RB. Hence the population will be comprised of a set of RBs. It means that not individual rules will mutually compete but whole RBs as entities. As only one RB is chosen at the same moment so there is no need of solving conflicts, i.e. the module CRS will be omitted. Also from other reasons mentioned later the structure of Pittsburgh approach is not only simpler but it also resembles to the structure of GA more than Michigan approach, which utilizes GA only in one module (RDS).

Pittsburgh approach [31] was originally motivated just by its older Michigan's opposite and it tries not only to simplify the structure of Michigan approach, where a number of various auxiliary tasks is necessary to be done by often very heterogenous approaches but also to spread the range of convenient problems. Michigan approach needs for evaluation (CAS) to define the so-called *performance index* and that can be a quite complicated problem because the credit assignment is not trivial. If a reward is obtained it will be distributed in a way between all rules, which more or less contributed to this reward. Defining such a way may not be again easy. However, there are many applications, where simple error measures are satisfactory for defining the evaluation and in Pittsburgh approach it is directly assigned to a given RB. Thus CAS can be reduced to an *evaluation system* (ES) in this case. Finally, the main simplification is in the absence of a complicated PS because only one RB is chosen instead of a set of mutually competing rules affected by eventual conflicts. In other words, PS is reduced to a conventional *base of RBs* (BRB) containing their complete population, see fig. 7.

ES, BRB and RDS (like in Michigan approach) create the basic structure of Pittsburgh approach. Besides, there is an additional module a conventional *rule-based system* (RBS), which performs basic interactions between Pittsburgh approach and its environment. If we look at BRB we see its structure as well as size are much more complex and greater than the RB in Michigan approach (approximately multiplied by the number of RBs), which is the tax for structural simplicity. First of all this will be visible in the ES, which is computationally very demanding. Further limitations are connected with the code length. Using a simple GA requires that all RBs have constant lengths to be able to apply conventional crossover operators if generating new RBs, i.e. offsprings. It means we require from all RBs a fixed number of rules, which is a very rigid demand. Therefore, new types of crossover operators are needed but they would be very different from the natural prototype.

The rule learning approach is started with a set of initial RBs. They interact through the RBS with the environment consecutively, which sends responses back to the learning system. Unlike the reward in sect. 3.1 they can be positive (reward) as

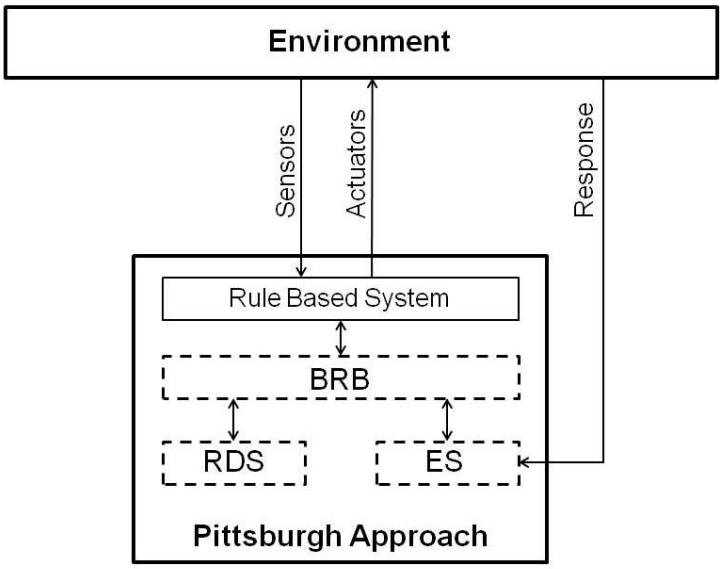


Figure 7. Basic structure of Pittsburgh-type adaptation system.

well as negative (penalty). ES performs evaluation of these RBs and assigns fitness to them. Once the evaluation process is finished the module RDS starts selection of individual RBs, which will advance to generating new RBs by crossover operators, where they are recombined from old ones. The mutation serves as a generator of principally new rules. After completing a new population the cycle will be again repeated. A pseudocode of Pittsburgh approach is in fig. 8.

Comparing Michigan and Pittsburgh approach we see Michigan system is much more complex but if it is once designed then it requires much less computational effort and has higher search ability for finding good rules than Pittsburgh approach [15]. However, for simpler applications and where the inductive learning based on examples is dominant Pittsburgh approach is suitable enough. Nowadays these systems are incorporated to more complex systems, which are used e.g. for pattern recognition problems, scheduling [30] or as means for data mining like for instance systems KEEL [1], SGERD [21] and KASIA [29].

```
Initialize a starting set of RBs
do while termination criteria are not fulfilled
    for i = 1 to number of RBs
        Interact with the environment by RB_i
    end for
    Evaluate RBs
    Select parent RBs
    Generate by crossover new RBs
    Apply mutation to RBs
    Create new population
end do
```

Figure 8. Pseudocode of Pittsburgh-type adaptation system.

4 Modifications and Enhancements of Genetic Fuzzy Systems — Present State-of-Art

The methodologies described in the previous sections 2 and 3 are fundamental for designing further modifications and enhancements in the area of GFSs, which have been proposed in huge numbers exceeding the range of this chapter. Therefore, this section describes only some selected approaches, which are in focus of present research presented on renowned conferences like e.g. World Congress on Computational Intelligence or in some overview papers as [13].

These modifications relate evolutionary methods as well as various modifications of systems based on fuzzy logic. There is a lot of reasons why so many methods have been proposed. We mention the two most important reasons. Firstly, basic GFS approaches exhibit many weaknesses like computational complexity of GAs, low quality of extracted rules, their high number and hereby low interpretability. Secondly, the mentioned GFSs are primarily based on using conventional GAs but many new evolutionary oriented methods have been proposed as for instance differential evolution, parallel GAs, particle swarm optimization or new approaches for calculating fitness functions like multi-objective EAs. Analogously, a conventional Mamdani fuzzy controller is not the only fuzzy system. There are several modifications of fuzzy rules like TSK rules, fuzzy relational rules, various linguistic models or other systems as fuzzy regression, fuzzy clustering, fuzzy cognitive maps, etc. Some of these methods and systems will be shortly described in next sections.

4.1 Design of TSK Fuzzy Controllers by Genetic Algorithms

Takagi-Sugeno-Kang (TSK) fuzzy controllers are the most used fuzzy controllers in real applications at all and especially from this significance reason we will show

a way how to set-up their parameters. This kind of controllers [33] was derived from the original Mamdani-type fuzzy controller as a simplified variant where only antecedent parts of RB are in the form of fuzzy sets but consequents are functions with crisp outputs, which depend on input variables, i.e.

$$\text{IF } x_1 \text{ is } LX_1 \;\; \& \;\; \ldots \;\; \& \;\; x_n \text{ is } LX_n \quad \text{THEN} \quad u^* = f(X_1, \ldots, X_n), \tag{2}$$

where x_i are input values of variables X_i to the fuzzy inference rule (2) and u^* is the output value in a numerical form (not more fuzzy). The output function f can have various forms. For our considerations let us suppose it in the form of a linear combination $u^* = w_1.X_1 + \ldots + w_n.X_n$, where w_i are function parameters.

As mentioned above, there are at least two basic groups of parameters in a rule-based fuzzy controller, i.e. parameters of MFs and RB that create together a KB. However, often the minimization of RB is required, not only from reasons of computational complexity but simpler RBs are more robust, too. In [18] a GA-based approach is described, where in total three types of parameters are adjusted, beside that for MFs and RB also the number of rules. The basic idea of the KB design consist in defining a structured individual (chromosome), which represents the whole KB. From this viewpoint the proposed approach resembles to a specific form of Pittsburgh approach.

The concrete form of an individual depends on the types of membership and output functions we use for KB design. If a MF MF_j is described by k parameters, e.g. a Gaussian function is described by its *centre* and *variance*, the output function OF_j by l parameters (in the case of a linear combination there are coefficients w_i) then an individual will have up to np parameters:

$$np = k.\sum_{i=1}^{n} |X_i| + l.\prod_{i=1}^{n} |X_i|, \tag{3}$$

where $|X_i|$ is the number of linguistic values defined on the input variable X_i and j is the ordering index $j = 1, \ldots, \sum |X_i|$. The expression $\prod |X_i|$ gives us the number of all mutually consistent (non-contradictory) rules, which is an upper limit of the number of parameters and it can be of course reduced. The chromosome structures of both function types as well as the total structure of the individual for a complete KB are depicted in fig. 9. We can see the final individual arose by merging particular chromosome structures of these functions. We need a special list, where the combinations of *MFs* and the related *OF* in form of their indexes are saved to be able again to rewrite the chromosome to a rule list. After applying a GA we will get one winning individual. Other individuals can be removed. Finally, the winner

will be backwardly rewritten into a rule list.

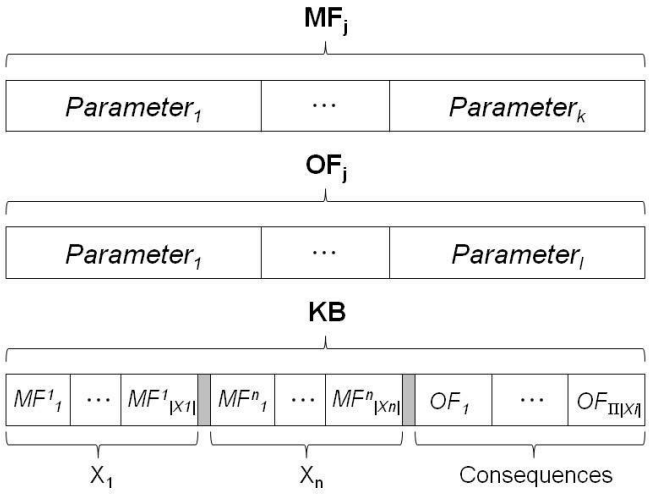


Figure 9. Chromosome structure of the TSK controller.

This chromosome structure represents the so-called *approximate type* because we cannot assign obtained MFs to their linguistic values like *small, warm*, etc. The reason is that in our case there are definitely some linguistic values, which appear in various combinations of several rules, e.g. *If distance is short and speed is small* ... or *If distance is short and speed is high* ... However, there is no mechanism for securing that obtained MFs for both occurrences of the term *short* will be identical, too. If it would be possible in such a case then we will get a *descriptive type*.

The coding of the chromosome was proposed originally as binary but in [8] a comparison between binary and real coding was done. Experiments showed that the real coding works quicker but not so much accurately than the binary one. However, if the real coding values are sampled into a finite set of allowed values its quality becomes better.

This system tries also to minimize the number of rules but a rule removal mechanism is needed. During processing GA there will arise also some MFs, which have low heights (small values of grades of membership). In such a case these functions will be removed. If a substantial part of the antecedents is missing due to such a removal then the whole rule will be deleted. The definition of a fitness function depends on a given application but its dividing by the number of remaining rules causes that a RB with a smaller number of rules will be preferred to some extent.

As seen in (3) this method supposes partitions of variables $|X_i|$, i.e. numbers of linguistic values are predefined, which is a considerable disadvantage because it is not easy to estimate correct numbers but they influence the quality of designed KB very strongly. In [20] a *self-constructing evolution algorithm* (SCEA) is presented, which tries to automatically design optimal numbers of linguistic values using the so-called *sequence search based on dynamic evolution* (SSDE). Its processing resembles to IRL approach where in each iteration step only one rule is created. In such a manner individuals do not represent a whole RB but only one rule and the corresponding population describes various possible combinations of this rule.

If we depict individual samples (elements) of a given training data set as points mostly we see they are concentrated in several groups known as clusters (see fig. 2), which define the number of rules and hereby MFs as well because one cluster corresponds to one rule. These clusters represent a knowledge structure, which will be created by structural learning, whose the first task is determining whether a new rule should be extracted from the training data or not. At the beginning the rule base is empty and the role of SSDE is during a series of new populations to consecutively build a RB. If a sample entering the algorithm does not activate any of existing rules, i.e. none of their strengths α exceeds a given threshold α_T then a new population will start and a set of mutually competing chromosomes will be generated, which one winning chromosome will be added to RB from. New populations are produced until all training samples are covered by at least one rule, which is a stopping criterion, too. After that the second stage of learning, the parametric one, will be processed, where using the same training data the parameters of membership and output functions will be optimally tuned. Thus SCEA can be described by a pseudocode in fig. 10.

Initial values of MFs for the first chromosome C_1^j depend on the input values $x_1, \ldots, x_n$ of the sample S. For instance, if MFs are chosen to be Gaussian ones then the mean values will be adjusted to these inputs and variances as well as parameters of the output functions will be randomized from given intervals of allowed values. Further chromosomes C_k^j will be modified from C_{old} in dependence of $f(C_{old})$ — the better the fitness of C_{old} the smaller its modification, i.e. the smaller difference between C_{old} and C_{new}. Still one final remark to this approach is convenient to be mentioned. Intentionally we used the term population instead of generation because after adding the winning rule the original population is canceled and a totally new one is created (therefore repeatedly $k = 0$). There is no continuation of the original population through some modifications into its next generations, i.e. no inheritance.

$j \leftarrow 1$
do while all samples are not covered by RB
  Enter new sample S
  if for S all rule strengths $\alpha \leq \alpha_T$
    $k = 0$
    Create the first chromosome of the new population $C_1^j = C_{old}$
    do while $k \leq k_{max}$ (*SSDE algorithm*)
      Create a competing chromosome $C_k^j = C_{new}$
      Evaluate fitness values $f(C_{old})$ and $f(C_{new})$
      if $f(C_{new}) \geq f(C_{old})$
        $C_{old} \leftarrow C_{new}$
      end if
    end do
  end if
  Add the rule from C_{old} to RB
  $j \leftarrow j + 1$
end do
Process parametric learning

Figure 10. Pseudocode of SCEA.

4.2 Evolutionary Fuzzy Regression Analysis

The aim of regression analysis is to express a generalized relationship in the form of a function between a *dependent variable* y and *independent variables* $x_1, \ldots, x_n$, which can represent e.g. measured samples and in a state space they are depicted as points. Such a regression function can be of various types but the most utilized one is linear and therefore it is known as *linear regression analysis* (although nonlinear forms also exist, e.g. [16]). In other words, the regression is a functional approximation of a set of measured data as it is depicted on an example in fig. 11. For our next considerations we will deal with a linear n-dimensional regression in the following form:

$$y = a_0 + a_1.x_1 + \ldots + a_n.x_n + \varepsilon, \tag{4}$$

where a_i are parameters and ε is the difference (error) between the real sample value y_s and the modelled value y, i.e. $\varepsilon = y_s - y$. The goal of the regression analysis is to determine the parameters a_i ($i = 1, \ldots, n$) at minimizing ε ($\varepsilon \to 0$).

Many situations exist, which are characterized by nonlinear relations, whose parameters are fuzzy [26]. Also in such cases linear approximation is often possible if we realize convenient linearization substituting parts of nonlinear functions by a linear polynomial as (4). However, fuzziness cannot be treated by conventional means. Thanks the so-called *extension principle* constructing any arithmetic

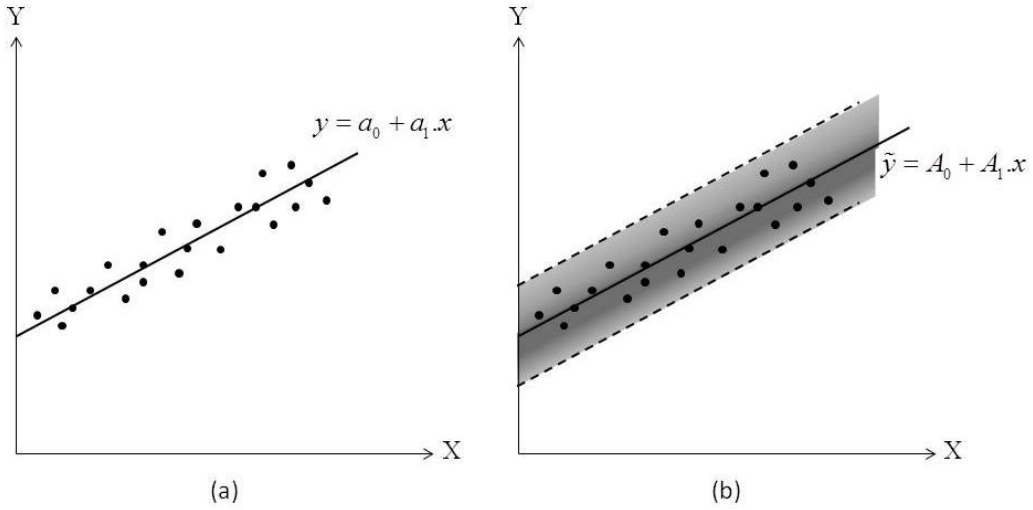


Figure 11. Examples: (a) conventional linear regression function $y = a_0 + a_1.x$, (b) fuzzy linear regression function $\tilde{y} = A_0 + A_1.x$; • — samples.

function is enabled, whose arguments are *fuzzy numbers*, i.e. special MFs for characterizing meanings like '*approximately x*'. More detailed information about this domain can be found e.g. in [22]. In such a case to avoid confusing we rewrite (4) into:

$$\tilde{y} = A_0 + A_1.x_1 + \ldots + A_n.x_n + \varepsilon, \tag{5}$$

where $\tilde{y}$ is fuzzy as well as A_i are fuzzy parameters, too. Further, we will suppose that $\varepsilon = 0$. Graphically, the differences between linear and fuzzy regression are schematically depicted in fig. 11. Fuzzy regression covers also other samples although with lower grades of membership (different levels of shading correspond to the membership) unlike the conventional approach, where ε values are evident.

The first approach how to set-up parameters in (5) was proposed by Tanaka [34] known also as *possibilistic model*. It is based on minimizing the fuzziness of used MFs, i.e. the sum of their supports $J = c_1 + \ldots + c_n$, at conjoint securing that the proposed regression model will fit input data to the degree $h \in [0;1]$, the so-called *h-certain factor*, which is given by a user and influences the fuzziness of the model. Higher values of h cause bigger covering of input data but also higher fuzziness of the model. Further, we will suppose triangular types of MFs, see fig. 12, with

following definitions of A_i:

$$\mu_{A_i}(x_i) = \begin{cases} 1 - \frac{|x_i - a_i|}{c_i}, & a_i - c_i \leq x_i \leq a_i + c_i \\ 0, & \text{otherwise} \end{cases}, \tag{6}$$

where c_i is the spread of the MF A_i with its peak value a_i, i.e. $\mu_{A_i} = A_i(a_i, c_i)$. In [34] the parameters of $\mu_{\tilde{y}_k} = \tilde{y}_k(a_y^k, c_y^k)$ for the k-th sample $(x_1^k, \ldots, x_n^k)$ using properties of the extension principle are derived as:

$$a_y^k = a_0 + \sum_{i=1}^{n} a_i . x_i^k, \tag{7}$$

$$c_y^k = c_0 + \sum_{i=1}^{n} c_i . |x_i^k|. \tag{8}$$

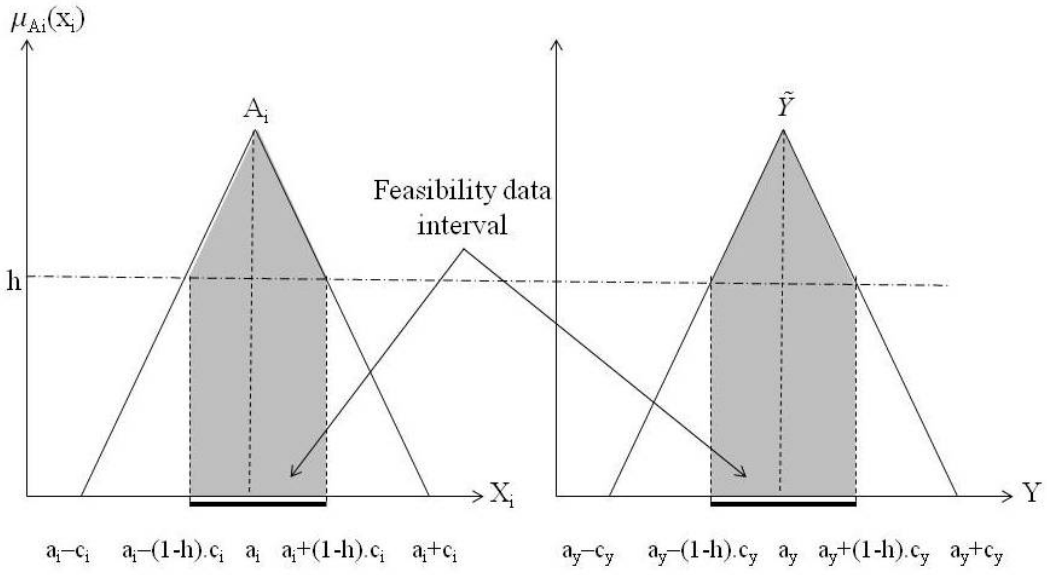


Figure 12. Definitions of MFs for $\tilde{y}$ and A_i.

If we have M training samples the function J can be rewritten as:

$$J = \sum_{i=1}^{n} (\sum_{k=1}^{M} |x_i^k|). \tag{9}$$

Again, it can be proven that the model with the h-certain factor will be found if minimizing (9) under following conditions valid for all samples $(x_1^k,\ldots,x_n^k,y_s^k)$:

$$a_0+\sum_{i=1}^{n}a_i.x_i^k+(1-h).[c_0+\sum_{i=1}^{n}c_i.|x_i^k|]\geq y_s^k, \tag{10}$$

$$a_0+\sum_{i=1}^{n}a_i.x_i^k-(1-h).[c_0+\sum_{i=1}^{n}c_i.|x_i^k|]\leq y_s^k. \tag{11}$$

Applying linear programming of the problem (9)–(11) we get resulting definitions of fuzzy parameters A_i.

As already mentioned, for design of an approximation of although nonlinear functional dependencies often a linear regression model is enough convenient (e.g. if for $y=a_0+a_1.x_1+a_2.x_2^2$, which is nonlinear, we put a substitution $x_2'=x_2^2$ then we will get (4)). In [4] a combination of genetic programming and linear regression analysis was used for designing linear regression models of nonlinear functions. Genetic programming is responsible for constructing fuzzy polynomials with help of hierarchical trees, which define the structure of such a polynomial as a set of linear as well as nonlinear elements. The intermediate nodes of such a tree represent either an operation of summation or multiplication. Only these two kinds of arithmetic operations are needed for constructing a polynomial like (4). Terminal nodes are in the form of independent variables $x_1,\ldots,x_n$ or fuzzy parameters A_i. The role of the regression analysis is to determine these parameters A_i. The overall pseudocode of the proposed algorithm is shown in fig. 13.

$j\leftarrow 1$
Initialize the population $P(j)=\{C_1(j),\ldots,C_l(j),\ldots,C_N(j)\}$
Assign fuzzy parameters A_i to all chromosomes of $P(j)$ by [34]
Compute fitness for all chromosomes of $P(j)$
do while until terminal condition is fulfilled
  Do pattern selection $P(j+1)\leftarrow P(j)$
  Do crossover of $P(j+1)$
  Do mutation of $P(j+1)$
  Assign fuzzy parameters A_i to all chromosomes of $P(j+1)$ by [34]
  Compute fitness for all chromosomes of $P(j+1)$
  $j\leftarrow j+1$
end do

Figure 13. Pseudocode of evolutionary fuzzy regression.

The chromosomes in this algorithm are encoded hierarchical trees. Basically, in both chromosomes a tree node is randomly selected and remaining subtrees are either

mutually interchanged at the crossover or substituted by a random tree structure at the mutation. The quality of an individual can be rated by two criteria — accuracy and simplicity (interpretability) and the fitness function of the l-th chromosome could look as e.g.:

$$fitness_l = \frac{1 - MAE_l}{1 + \exp[c_1.(L_l - c_2)]}, \tag{12}$$

where c_1 and c_2 are penalty coefficients, L is the number of nodes encoded in the l-th chromosome and *MAE* is the well known *mean absolute error* between the real sample value and the modelled one:

$$MAE = \sqrt{\frac{1}{M}.\sum_{k=1}^{M}\left|\frac{\varepsilon}{y_s^k}\right|}. \tag{13}$$

4.3 Multi-Objective Evolutionary Fuzzy Systems

Recently, a very high interest has been put on constructing KBs, which would be able not only to accurately function but also they would be comprehensible, which is the original aspect of fuzzy systems. Eventually, further requirements regarding the form and properties of KBs come into foreground. These criteria are often contradictory and therefore, it is necessary to incorporate a mechanism, which would keep a balance (trade-off) among them. This is the reason for constructing various *multi-objective evolutionary fuzzy systems*, which are typical with a number of miscellaneous information measures for evaluating the quality of proposed KBs as well as transformations for getting KBs into a suitable form. Some of these means will be described in next sections.

4.3.1 Granularity Transformations

Among a number of possible factors and requirements put on a resulting KB there are the three especially significant ones: *accuracy, RB complexity* and *partition integrity*. Whereas the first two criteria are apparent (RB complexity, i.e. number of rules and size of their antecedents) the partition integrity depends on several deeper relations and its definition is rather intuitional and implicit but most experts agree with following common properties of a fuzzy partition with a high integrity grade [2]:

1. there is reasonable number of linguistic values, i.e. fuzzy sets defined on a given linguistic variable, i.e. the universe of discourse,

2. all fuzzy sets are normal, i.e. at least one element has the grade of membership equal to 1, i.e. full membership,

3. fuzzy sets should be distinguishable enough, i.e. overlap of their MFs should not be too significant,

4. the universe of discourse is fully covered, i.e. all its elements belong at least to one fuzzy set.

Concerning the first property, psychological experiments showed that for most humans the maximum reasonable number of linguistic values T_{max} is between *7–9*. Concerning the remaining properties, these are fulfilled if the partitions satisfy the following condition $\sum \mu(x) = 1$ for all $x \in X$ and MFs μ. A uniform partition (equidistant division of the universe of discourse) strengthens these properties and makes RB built on such a partition and subsequent definitions of MFs easy readable. The problem is that after applying an automatic learning approach, which tries to approximate its designed KB to training data so accurately as possibly affects the partitions of universes of discourses, i.e. supports of some MFs will be broader and of some others closer (see fig. 15). However, such a KB is less interpretable — the more distant from a uniform partition the less comprehensible. To find an equilibrium between a readable but not accurate KB design based on uniform partitions and an accurate but not readable one based on real partitions we introduce the so-called *virtual RB* and a piecewise linear transformation between the virtual and real RB [2]. It is obvious that the more the piecewise linear transformation resembles to a line then the fuzzy partition will tend to a uniform one. We define the so-called *integrity index*:

$$I = 1 - \frac{2.\sum_{i=1}^{n+1} d_i}{(n+1).(T_{max}-1)}, \tag{14}$$

where the expression $n+1$ stands for n input and one output variables if the rule structure is like (15). The measure d_i (explained later) has its maximum values $(T_{max}-1)/2$, in other words (14) is in a scaled form, i.e. $I \in [0;1]$, where $I = 1$ means the partition remains fully uniform.

Let us consider rules of the Mamdani type in the following form:

$$\text{IF } x_1 \text{ is } LX_1 \ \ \& \ \ \ldots \ \ \& \ \ x_n \text{ is } LX_n \quad \text{THEN} \quad u \text{ is } LU, \tag{15}$$

where LX_i is a general indication of a linguistic value of the i-th input variable X_i ($n+1$-th for LU) that takes the j-th value from the T_i number of linguistic

values defined on a given universe of discourse denoted as $A_{i,j}$ (in our case also the corresponding MF).

The first task is how to solve mapping a fuzzy partition with the maximum number of MFs, i.e. $P_i' = \{A_{i,1}', \ldots, A_{i,max}'\}$, to another one with an optimized number of MFs $P_i'' = \{A_{i,1}'', \ldots, A_{i,opt}''\}$ ($opt < max$). Each MF is defined by a set of parameters, e.g. a triangular MF $A_{i,j}$ is described by three points $a_{i,j}, b_{i,j}$ and $c_{i,j}$, which are the left margin, peak and the right margin, respectively. The most natural way is to order a MF $A_{i,j}'$ to another MF $A_{i,l}''$ if their peaks $b_{i,j}'$ and $b_{i,l}''$ have the minimum distance, see fig. 14. We see that more MFs from P_i' can be assigned to one MF in P_i''. These assignment rules $A_{i,j}' \rightarrow A_{i,l}''$ enable transformation of an accurate but difficult readable RB created on the maximum partitions P_i' to a simpler RB defined on P_i''. In other words, we created a virtual RB on P_i' at first and then using the assignment rules $A_{i,j}' \rightarrow A_{i,l}''$ we transform it to a RB defined on P_i''. However, P_i'' is not yet the final partition. One more transformation is still needed.

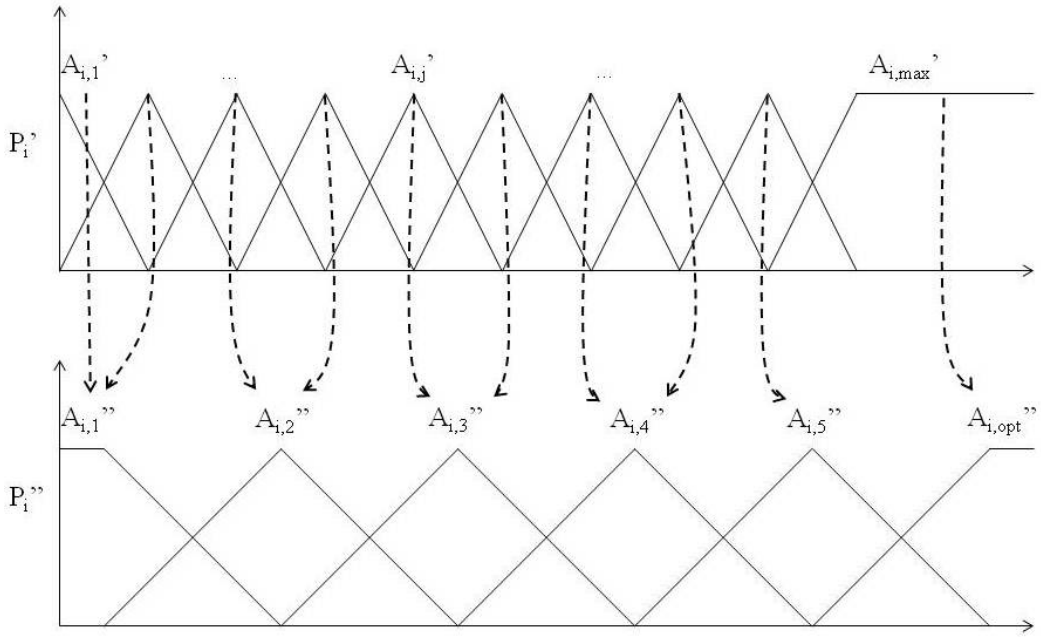


Figure 14. Process of mapping a fuzzy partition P_i' with the maximum number of MFs to a fuzzy partition P_i'' with the optimized number of MFs.

The second task deals with a transformation between MFs created on an originally uniform maximum fuzzy partition P_i' and the final $P_i = \{A_{i,1}, \ldots, A_{i,opt}\}$ one with the optimized number of MFs, which is a certain compromise between the

uniform and real partition based on approximation of training data. For this purpose we create beside the virtual partition P_i' still one more virtual P_i''', which is obtained by a learning process from training data, see fig. 15.

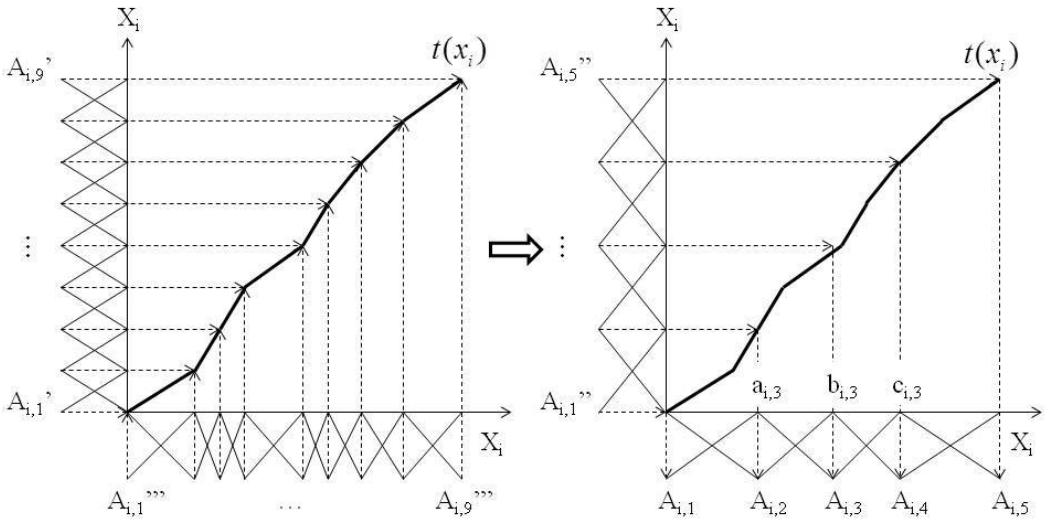


Figure 15. Piecewise linear transformation between virtual fuzzy partitions $P_i' \to P_i''' \to P_i'' \to P_i$.

The process of obtaining the final fuzzy partition P_i is depicted in fig. 15 by arrows. Firstly, we define the maximum number of MFs T_{max} and create a uniform maximum partition P_i'. Then using a learning approach we generate a virtual RB on fixed P_i'. On the other hand side using the same training data we generate real MFs on P_i'''. The coordinate intersections of P_i' and P_i''' peaks determine the break points, which represent bounds of linear pieces of the transformation function:

$$t(x_i) = \frac{b'_{i,j} - b'_{i,j-1}}{b'''_{i,j} - b'''_{i,j-1}} . (x_i - b'''_{i,j-1}) + b'_{i,j-1}. \tag{16}$$

After obtaining the value of optimized number of MFs (in a way) we can construct a uniform fuzzy partition P_i'' with T_{opt} MFs and the transformation function $t(x_i)$ will be copied into the coordinate system $P_i \times P_i''$. The inverse function of $t(x_i)$ will finally calculate real MFs built on the fuzzy partition P_i. Concurrently, the virtual RB is mapped into a RB proposed on MFs of the partition P_i. If more than one rule from virtual RB are mapped into one rule of the real RB then remaining rules will be omitted. We see we can obtain MFs as well as RB at one moment, which is similar to a human approach. Further, we can also see in fig. 15 that the resulting

partition P_i resembles to a uniform partition much more than the original definition of MFs from P_i'''. This shift to uniformity is due to piecewise linearity, where only the parameters $a_{i,j}, b_{i,j}$ and $c_{i,j}$ are transformed and linearly interconnected. If they all hit the same linear piece of the transformation function $t(x_i)$ then such a MF will preserve the uniformity.

In the case we use an evolutionary approach first of all we need to define the structure of a chromosome and the fitness function, which is presented by the *integrity index I* (14), where the *dissimilarity measure* d_i is calculated as:

$$d_i = \sum_{j=2}^{n-1} \left| b'_{i,j} - b'''_{i,j} \right|. \tag{17}$$

The chromosome C is composed of three parts $C = \{C_1, C_2, C_3\}$, which are represented by: (C_1) virtual RB designed on P', (C_2) searched optimal granularities, i.e. numbers of MFs and (C_3) definitions of MFs constructed on P''', respectively. In all three parts of a chromosome the parameters for all variables are contained and the chromosome, which has the best value of the fitness function (14) will have the best interpretable KB, too.

4.3.2 Pareto-based Approaches in Rule Extraction

In many adaptation applications we meet with a problem that too many rules have been extracted and such a RB is unreadable or even influences of individual rules behave as errors (e.g. overlaps, contradictions, low rule weights, etc.). Therefore, it is necessary to define information measures, which could evaluate the criteria of goodness or interestingness for individual rules. (In the literature there are lots of various measures, e.g. *integrity index* in sect. 4.3.1.) It was proven that *Pareto-optimal rules* suit very well to these criteria. The aim is to select significant rules from a given set of rules, which would represent a smaller but functionally equivalent RB to the original one.

Further, we will consider weighted rules for a classification problem, where given samples belong at least to one of m classes C_h ($h = 1, \ldots, m$), i.e. the k-th rule with n inputs and weight w_k is in the following form:

$$(R_k:) \quad \text{IF } x_1 \text{ is } A_{k,1} \ \ \& \ \ \ldots \ \ \& \ \ x_n \text{ is } A_{k,n} \quad \text{THEN} \quad C_h \text{ with } w_k. \tag{18}$$

As the total number of classes m is different from the number of rules therefore, we need to use also different indexes — h for classes and k for rules.

To measure the goodness of each rule basically two evaluation criteria *confidence*

$conf(\alpha_k \Rightarrow C_h)$ and *coverage* $covr(\alpha_k \Rightarrow C_h)$ are defined [25]:

$$conf(\alpha_k \Rightarrow C_h) = \frac{\sum\limits_{x_p \in C_h} \alpha_k(x_p)}{\sum\limits_{p=1}^{M} \alpha_k(x_p)}, \tag{19}$$

$$covr(\alpha_k \Rightarrow C_h) = \frac{\sum\limits_{x_p \in C_h} \alpha_k(x_p)}{M_h}, \tag{20}$$

where α_k is the rule strength, $x_p = (x_{p,1}, \ldots, x_{p,n})$ is a sample with n elements ($p = 1, \ldots, M$) and M_h is the number of samples belonging to the class h.

The confidence relates the rule strengths of samples belonging to the class h and rule strengths of all M samples. If the rule k classifies samples unambiguously only to the class h then $conf = 1$ else it can limit to 0. In other words, the confidence is the rule weight w_k, too. The coverage is convenient if there are imbalanced data, i.e. if there are significant differences between numbers of samples belonging to individual classes. It expresses a mean rule strength for correctly classified samples.

At first, we need to obtain a set of 'rough' rules, which will be processed by the Pareto-based approach. We can use any extraction algorithm, which generates yet too many rules or knowing fuzzy partitions (see sect. 4.3.1) we will generate all possible combinations of input linguistic values $A_{k,i}$. After evaluating (19) and (20) for all rough rules we start with their selection. Rules, which have both confidence and coverage values smaller than given thresholds, will be omitted. Alike the rules with high confidence and small coverage will be removed because they are although accurate but very specific. From the remaining rules the so-called Pareto-optimal, eventually *near* Pareto-optimal, rules will be extracted, which usually guarantee satisfactory classification of samples.

The relation between Pareto-optimal and near Pareto-optimal rules is evident in fig. 16. A rule R_i is ε-*dominated* by another rule R_j if at least one of the following two conditions is satisfied:

$$covr(R_j) > covr(R_i) + \varepsilon_{covr} \quad \text{and} \quad conf(R_j) \geq conf(R_i) + \varepsilon_{conf}, \tag{21}$$

$$covr(R_j) \geq covr(R_i) + \varepsilon_{covr} \quad \text{and} \quad conf(R_j) > conf(R_i) + \varepsilon_{conf}, \tag{22}$$

where ε_{conf} and ε_{covr} are confidence and coverage margins, respectively and generally we speak about the ε-*dominance*. If a rule R_i is not ε-dominated then it is a ε-*Pareto-optimal rule* (ε stands for *near*), see grey circles in fig. 16 (b). Of course, if $\varepsilon = 0$ we get a Pareto optimal rule (black circles). The relation between ε_{conf} and

ε_{covr} is evident from fig. 16:

$$\frac{\varepsilon_{covr}}{\varepsilon_{conf}} = \frac{\max\{covr(R_k)\} - \min\{covr(R_k)\}}{\max\{conf(R_k)\} - \min\{conf(R_k)\}}, \tag{23}$$

where R_k are all ε-Pareto-optimal rules. These R_k rules become candidate rules, which proceed to the final stage of constructing an optimized RB concerning the number of rules and their interpretability.

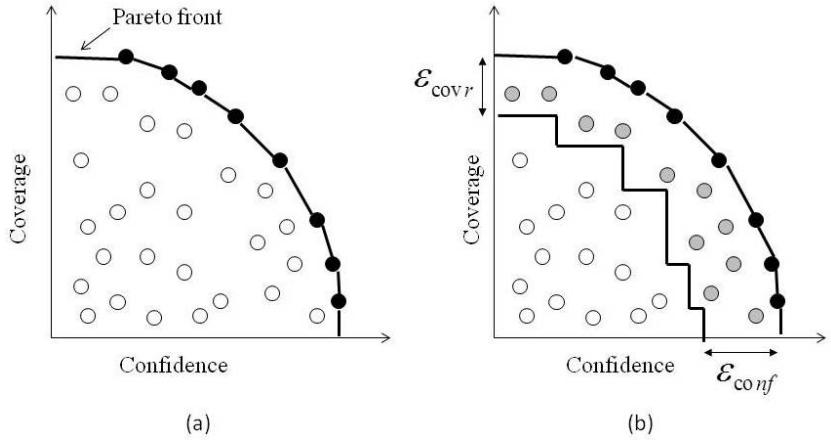


Figure 16. Example of: (a) Pareto-optimal rules — black circles, (b) near Pareto-optimal rules with ε-dominance margin — grey circles [25].

After selecting N candidate rules we create a population of chromosomes $C = (c_1, \ldots, c_N)$, which are binary strings of 0 and 1. If $c_k = 0$ the candidate rule R_k ($k = 1, \ldots, N$) is not included into the RB represented by the given chromosome C. Such a population is basically processed by a usual GA, where fitness of C reflects two properties:

– $f_1(C)$ — average quality of classification for each class,

– $f_2(C)$ — number of selected rules.

The goal is to maximize $f_1(C)$ and to minimize $f_2(C)$. The final fitness f can be calculated as simple division $f(C) = f_1(C)/f_2(C)$. The winning RB has the highest fitness value.

5 Fuzzy Genetic Systems — Some Examples

As already mentioned in the introduction, GAs (EAs) can be used not only for design of KBs for fuzzy systems, in other words, GA as an adjective and fuzzy as a noun but the view can be also reverse, i.e. fuzzy systems help to GAs to be more efficient or computationally simpler (fuzzy as an adjective and GA as a noun). There is potentially a very broad variety how to utilize fuzzy systems but in spite of that FGSs are not so much spread than GFSs. However, there are some suitable examples, which can show possibilities for penetrating fuzzy logic into evolutionary computing. Some of these ideas and applications are described in this section.

Fitness calculation belongs to the most costly operations during a computational cycle of GA. To eliminate the risk of trapping in a local extreme the population used needs to be large enough. Moreover, usually there are necessary at minimum thousands of populations to be able to reach the optimal solution. Therefore, fitness approximation methods have been developed to spare computations and time.

One possibility is to propose a simpler function, which would approximate the original fitness function, see fig. 17. However, approximation usually fits only some characteristic points of the fitness function and others can considerably differ. The fitness function is very sensitive to extremes, where is a risk the global extreme will by shifted, the so-called *false optimum problem*. Therefore, another idea is to use both the original and approximation functions. A smaller part of individuals will be evaluated by the original fitness function and other individuals (the majority) by the approximation one. For grouping individuals into these two groups the fuzzy C-means clustering is used [36]. Using this approach individuals are grouped into clusters, whose centers are then evaluated by the original fitness function $f_O(v_k)$ and other individuals by a special approximation model $f_A(s_i)$, where v_k are centers of clusters C ($k = 1,\ldots,c$) and s_i are remaining N individuals. The final fitness of s_i calculated from all fuzzy clusters (an individual can belong to several fuzzy clusters) is computed as [36]:

$$f(s_i) = \sum_{k=1}^{c} [\rho_{k,i}.f_O(v_k) + (1-\rho_{k,i}).f_A(s_i)].\mu_k(s_i), \tag{24}$$

where $\mu_k(s_i)$ is the grade of membership of s_i to the cluster k and $\rho_{k,i}$ is the *reliability* of the fitness value, which depends on the distance s_i from the given cluster centre v_k, i.e. $d(v_k,s_i)$ and $\rho_{k,i} = 1/\exp(d(v_k,s_i))$. In such a manner the biggest influence of the cluster centre is in its surroundings. Otherwise, it decreases exponentially and the approximation model grows on its importance.

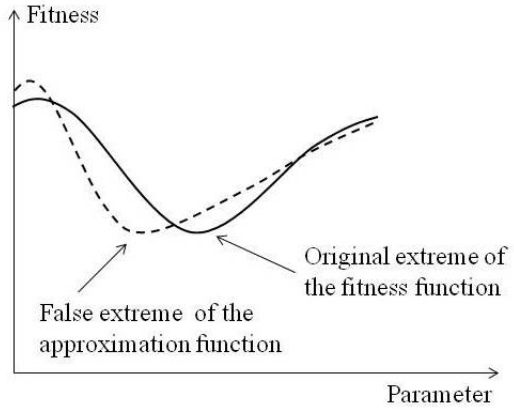


Figure 17. Example of the false optimum effect.

Further, we show an application of a fuzzy controller as a means for processing heuristic rules based on expert experience, which are used for determining some learning parameters in an optimization process. In [17] the use of a fuzzy controller and *K-nearest neighbours* algorithm for setting-up parameters of the *differential evolution* (DE) method is proposed.

DE [32] represent a relatively new branch in evolutionary computing and has been approved in various applications. Comparing to other GAs and EAs the approach of DE is characterized by several significant improvements like finding global optimum, number of adaptation parameters and convergence speed. However, determining the stopping criterion, i.e. number of iterations, is difficult. There are no known analytical derivations and so users are made to use a trial and error approach. However, an experienced expert knowing the information of the population can estimate the state of the searching process, too. For this reason it is necessary to know distribution of the population in the search space, i.e. a type of clustering is necessary. In this case the K-nearest neighbours method is used. K represents the number of the nearest elements that join on the classification of the investigated element (individual) in the search space into exactly one of disjoint clusters. The value K can be changed. If the mean distance between the investigated element and its K neighbours is less than a given threshold value then these elements share a common cluster else a new cluster will be created and accounted to the total number of clusters, which is the information influencing the number of DE iterations.

To determine the number of iterations the so-called *Iteration windows* method has been designed. After several initial DE iterations these are stopped and information about K and number of clusters is sent to a fuzzy controller, where heuristics and expert's experience are contented in KB of the controller as fuzzy IF–THEN

rules. Outputs of the controller are the value K for the K-nearest algorithm and the size of the iteration window for next stage, which means the number of iterations in the consecutive stage. The process is repeated until the window's size converges to 0 and hereby DE adaptation stops.

Similarly, fuzzy logic can be used in connection with a very spread optimization approach although not owing any stochasticity as EAs — *tabu search* (TS) [10]. TS is a generalization of the *mountain climbing* algorithm, which unlike to its ancestor does not generate its neighboring solutions by mutations but using allowed transformations t_j ($j = 1,\ldots,m$), i.e. a solution x_i ($i = 1,2,\ldots$) generates its neighbours $x_{i,j}$ forming a surrounding (neighborhood) $N_i = \{x_i, x_{i,1}, \ldots, x_{i,m}\}$, where $x_{i,j} = t_j(x_i)$. Definition of transformations depends on a concrete application, e.g. in the case of navigation they are allowed movements, or in decision making they are permitted actions in general. In each step the fitness values of elements from N_i are evaluated and the best one $x^*_{i,j}$ is selected for generating N_{i+1} in next cycle. In such a manner we get a series $x_1, x_2, \ldots$, where x_1 is the initial solution (point), $x_2 = x^*_{1,j}$, etc. obtaining a TS trajectory, see fig. 18. Together with corresponding transformations, we can a TS trajectory describe as a complete chain $x_1 \to t_{1\to 2} \to x_2, \ldots, x_i \to t_{i\to i+1} \to x_{i+1}, \ldots$, too.

However, the risk of such an approach is that the trajectory after several steps converges to a local optimum creating a closed loop. To prevent this situation a *short term memory* with a *tabu list*, i.e. a list with prohibited transformations is created, which is the main principal difference from the mountain climbing algorithm. For a certain number of steps the algorithm keeps in the tabu list inverse transformations t_j^{-1} to all that, which formed the TS trajectory (it is required each transformation t_j has also its inverse form t_j^{-1}). In other words, a tabu list prevents forming a closed loop, where its size is responsible for the quality of obtained results, beside others for the process of *exploration* and *exploitation*, too. If its size, i.e. number of considered steps, is too small then the algorithm can be trapped in a local optimum (closed loop) and if the size is too big then the global optimum can be 'skipped'.

Further, the so-called *long term memory* is defined, which penalizes too often used transformations. Each transformation, which is a part of the chain $x_1 \to t_{1\to 2} \to x_2, \ldots, x_i \to t_{i\to i+1} \to x_{i+1}, \ldots$ is extra counted as a *frequency* $\omega(t_j)$. If we search for a minimum (maximum) of a fitness function then to (from) its values a $\chi.\omega(t)$ value will be added (subtracted), respectively, where χ is a penalization constant. The long term memory can in such a way handicap the optimum solution if it would be reached by a too often used transformation. In such a case another solution is chosen as the optimum one and exploitation of another perspective region is induced. Similarly, as the short term memory also the size of its long variant influences the

search quality of the TS algorithm.

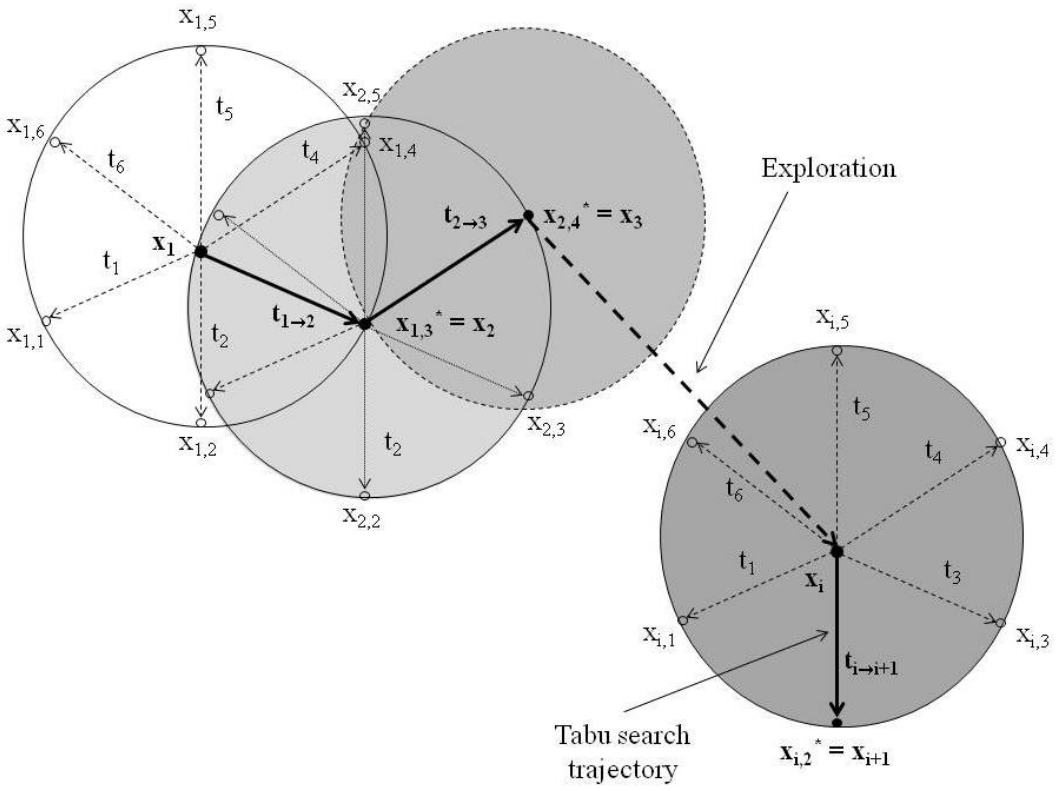


Figure 18. An example of tabu search trajectory visualization.

Determining the sizes of the short as well as long term memories is cardinal for finding the optimum solution or at least a near one to the optimum. However, it is a hard work and needs to do some analyses based on heuristics. One possibility is to do *visual diagnosis* [12], where a *distance radar* is used. This approach helps extracting relevant quantities and relations among them. For instance, in [24] following quantities were identified to control the sizes of the short *SMS* and long *LMS* term memory (*SMS* and *LMS* as output variables):

1. *IR — improvement rate* determines the current potential of improving a solution. It is the ratio between the number of all feasible transformations able to improve the solution and the total number of transformations.

2. *DTS — distance to solution* defines the distance of the current x_i and previous solution x_{i-1}.

3. *EXPE — exploration evaluation* gives us the information about the exploration measure of the search space.

Besides, further quantities and measures can be defined like diversity of solutions in the frame of one N_i or fitness proportions between the local optimum and its neighboring solutions [23]. These quantities can be fuzzified and as linguistic variables form fuzzy IF–THEN rules. They can be proposed manually or using an adaptation approach. In [24] a GFS is featured, which contemporarily controls values *SMS* and *LMS* as well as itself adapts its KB for this type of memory size control, see fig. 19. Against, [23] describes a GA based on predefined fuzzy rules, which control its processing, where the final solution represents again *SMS* and *LMS* values.

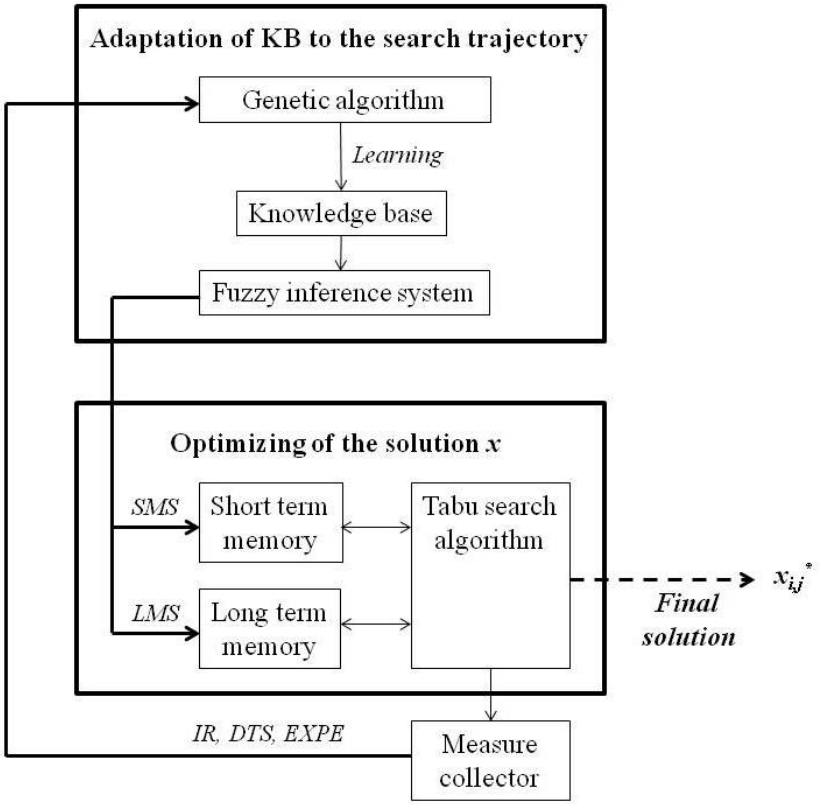


Figure 19. Control of tabu search short and long term memories by a self-adapting GFS.

6 Outlooks and Conclusions

In the Herrera's overview paper [13] current research trends of GFSs are divided into several categories:

1. enhancement of Michigan adaptation approach,
2. genetic learning of fuzzy partitions,
3. multi-objective genetic learning,
4. learning genetic models based on low quality data (noises and vagueness),
5. GA-based techniques for mining fuzzy association rules,
6. genetic adaptation of inference engine components.

In this chapter we could tackle more or less only the first three categories (sections 3.1, 4.2 and 4.3, respectively), which, of course, mutually often overlap. It is only a short and not exhaustive digest, which is restricted to the most known approaches in the area of GFSs, which should give a rough image about their potentials and possible use. The means described in sect. 3 represent basics of this area, which are already mostly closed but give still the potential to GFSs for their steady improvement by incorporation further modules, e.g. for utilization of low quality data or multi-objective approaches.

Besides, EAs (GAs) have spread also into further areas like, e.g. *fuzzy cognitive maps*, fuzzy clustering or *Type–2 FISs*. Of course, we cannot forget an intrinsic property of fuzzy logic — its 'fuzzification' ability, i.e. fuzzy logic is able to easy penetrate into other concepts, whose indication is the presence of the word 'fuzzy' in their names. This is the base for FGSs and some examples are contained in sect. 5. Their categorization is not easy because of the variability how fuzzy logic is incorporated into these systems.

EAs are only one of several means for automatic adaptation of FISs. There are also neural networks [19] and various interpolation and statistical methods [28]. However, EAs also exhibit very interesting properties in general, like their optimization capabilities and robustness against local minima traps. On the other hand side their computational complexity excludes them mostly from the on-line adaptation, which is required in many applications. From this reason maybe two eventual solutions seem to be perspective. Firstly, it is the development of further biologically and socially inspired analogies related to EAs like *swarm optimization* or *migration algorithms* and secondly, it is the use of parallel GAs and co-evolutionary approaches.

Acknowledgements: Research supported by the National Research and Development Project Grant 1/0667/12 "Incremental Learning Methods for Intelligent Systems" 2012–2015.

References

[1] Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F.: KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Journal of Multiple-Valued Logic and Soft Computing **17**(2–3), 255–287 (2011)

[2] Antonelli, M., Ducange, P., Lazzerini, B., Marcelloni, F.: Exploiting a three-objective evolutionary algorithm for generating Mamdani fuzzy rule-based systems. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. FUZZ, pp. 1373–1380 (2010)

[3] Casillas, J., Carse, B., Bull, L.: Fuzzy-XCS: A Michigan genetic fuzzy system. IEEE Transactions on Fuzzy Systems **15**(4), 536–550 (2007)

[4] Chan, K.Y., Dillon, T., Kwong, C.: Using an evolutionary fuzzy regression for affective product design. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. FUZZ, pp. 3242–3249 (2010)

[5] Cordón, O., Gomide, F., Herrera, F., Hoffmann, F., Magdalena, L.: Ten years of genetic fuzzy systems: Current framework and new trends. Fuzzy Sets and Systems **141**(1), 5–31 (2004)

[6] Cordón, O., Herrera, F., Hoffmann, F., Magdalena, L.: Genetic Fuzzy Systems — Evolutionary Tuning and Learning of Fuzzy Knowledge Bases, series Advances in Fuzzy Systems — Applications and Theory, vol. 19. World Scientific (2001)

[7] Cordón, O., del Jesús, M., Herrera, F., Lozano, M.: MOGUL: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. Int. Journal Intelligent Systems **14**(11), 1123—-1153 (1999)

[8] Damousis, I., Dokopoulos, P.: A fuzzy expert system for the forecasting of wind speed and power generation in wind farms. In: Proc. The 22nd IEEE on Power Industry Computer Applications (PICA), Sydney, Australia, pp. 63–69 (2001)

[9] Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Control, second edn. Springer (1996)

[10] Glover, F.: Future paths for integer programming and links to artificial intelligence. Computers & Operations Research **13**(5), 533—549 (1986)

[11] González, A., Pérez, R.: SLAVE: a genetic learning system based on an iterative approach. IEEE Transactions on Fuzzy Systems **7**(2), 176—-191 (1999)

[12] Halim, S., Lau, H.C.: Tuning tabu search strategies via visual diagnosis. In: K.F. Doerner, M. Gendreau, P. Greistorfer, W. Gutjahr, R.F. Hartl, M. Reimann (eds.) In: Metaheuristics: Progress as Complex Systems Optimization, pp. 365–388. Kluwer (2007)

[13] Herrera, F.: Genetic fuzzy systems: Taxonomy, current research trends and prospects. Evolutionary Intelligence **1**(1), 27–46 (2008)

[14] Holland, J.H., Reitman, J.: Cognitive systems based on adaptive algorithms. In: D. Waterman, F. Hayes-Roth (eds.) Pattern-Directed Inference Systems, pp. 313–329. Academic Press, London, United Kingdom (1978)

[15] Ishibuchi, H., Yamamoto, T., , Nakashima, T.: Hybridization of fuzzy GBML approaches for pattern classification problems. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics **35**(2), 359–365 (2005)

[16] Johanyák, Z.C., Kovács, S.: A brief survey and comparison on various interpolation-based fuzzy reasoning methods. Acta Polytechnica Hungarica **3**(1), 91–105 (2006)

[17] Lai, J.C., Leung, F.H., Ling, S.H.: A new differential evolution with self-terminating ability using fuzzy control and K-nearest neighbors. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. CEC, pp. 503–510 (2010)

[18] Lee, M., Takagi, H.: Integrating design stages of fuzzy systems using genetic algorithms. In: Proc. The second IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE), San Francisco, USA, pp. 613—-617 (1993)

[19] Lin, C.T., Lee, C.S.G.: Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems. Prentice-Hall PTR, New Jersey, USA (1996)

[20] Lin, S.F., Chang, J.W., Cheng, Y.C., Hsu, Y.C.: A novel self-constructing evolution algorithm for TSK-type fuzzy model design. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. CEC, pp. 570–576 (2010)

[21] Mansoori, E.G., Zolghadri, M.J., Katebi, S.D.: SGERD: A steady–state genetic algorithm for extracting fuzzy classification rules from data. IEEE Transactions on Fuzzy Systems **16**(4), 1061–1071 (2008)

[22] Mareš, M.: Computations Over Fuzzy Quantities. CRC-Press, Boca Raton, USA (1994)

[23] Marques, V., Gomide, F.: Fuzzy coordination of genetic algorithms for vehicle routing problems with time windows. In: Proc. of the Fourth International Workshop on Genetic and Evolutionary Fuzzy Systems (GEFS), Mieres, Spain, pp. 39–44 (2010)

[24] Marques, V., Gomide, F.: Memory control of tabu search with genetic fuzzy systems. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. FUZZ, pp. 2251–2257 (2010)

[25] Nojima, Y., Kaisho, Y., Ishibuchi, H.: Accuracy improvement of genetic fuzzy rule selection with candidate rule addition and membership tuning. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. FUZZ, pp. 527–534 (2010)

[26] Oblak, S., Škrjanc, I., Blažič, S.: If approximating nonlinear areas, then consider fuzzy systems. IEEE Potentials **25**(6), 18–23 (2006)

[27] Orriols-Puig, A., Casillas, J., Bernadó-Mansilla, E.: Fuzzy-UCS: A Michigan-style learning fuzzy-classifier system for supervised learning. IEEE Transactions on Evolutionary Computation **13**(2), 260–283 (2009)

[28] Pozna, C., Troester, F., Precup, R.E., Tar, J.K., Preitl, S.: On the design of an obstacle avoiding trajectory: Method and simulation. Mathematics and Computers in Simulation **79**(7), 2211–2226 (2009)

[29] Prado, R., García-Galán, S., Muñoz Expósito, J., Yuste, A.: Knowledge acquisition in fuzzy-rule-based systems with particle-swarm optimization. IEEE Transactions on Fuzzy Systems **18**(6), 1083–1097 (2010)

[30] Prado, R., García-Galán, S., Yuste, A., Muñoz Expósito, J., Bruque, S.: Genetic fuzzy rule-based meta-scheduler for grid computing. In: 4th Int. Workshop on Genetic and Evolutionary Fuzzy Systems (GEFS), Mieres, Spain, pp. 51–56 (2010)

[31] Smith, S.: A learning system based on genetic adaptive algorithms. Ph.D. thesis, Department of Computer Science, University of Pittsburgh, USA (1980)

[32] Storn, R., Price, K.: Differential evolution — a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization **11**(4), 341—-359 (1997)

[33] Takagi, T., Sugeno, M.: Fuzzy identification of systems and its applications to modeling and control. IEEE Transactions on Systems, Man and Cybernetics **15**(1), 116–132 (1985)

[34] Tanaka, H., Uejima, S., Asai, K.: Linear regression analysis with fuzzy model. IEEE Transactions on Systems, Man and Cybernetics **12**(6), 903–907 (1982)

[35] Venturini, G.: SIA: a supervised inductive algorithm with genetic search for learning attribute based concepts. In: Proceedings of European conference on machine learning, Viena, Austria, pp. 280—296 (1993)

[36] Yoon, J.W., Cho, S.B.: Fitness approximation for genetic algorithm using combination of approximation model and fuzzy clustering technique. In: Proc. of IEEE World Congress on Computatinal Intelligence (WCCI), Barcelona, Spain, vol. CEC, pp. 2531–2536 (2010)

Sentio, Ergo Sum: From Cognitive Models of Emotions towards their Engineering Applications

Mária VIRČÍKOVÁ[1], Peter SINČÁK[2]

Abstract. As robots are moving out from the industry to environments tenanted by humans, current trends in the field of Human-Computer Interaction are moving beyond cognition and expanding into the social experience which involves also artificial emotions. Emotional technology in its two forms – as an expression of artificial emotions of the systems and as systems capable of recognizing human emotions – contributes to the creation of personalized systems. Also, following neuroscience that says that emotions support human decision-making process, this paper continues in the statement of Minsky that says: the issue is not whether intelligent machines can have emotions, but whether machines can ever be intelligent without them. We try to move from the knowledge about human emotional processes to implement models of artificial emotions. This paper presents a survey of such emotional models and it tries to demonstrate that methods of computational intelligence can be appropriate tools for systems that involve emotions.

1 Introduction – from human emotion to emotion machines

American writer Carnegie [6] aptly remarked: "When dealing with people, remember you are not dealing with creatures of logic, but creatures of emotions." Human emotions act as a system with feedback provided by the

[1] Center for Intelligent Technologies, Department of cybernetics and artificial intelligence, Faculty of electrical engineering and informatics, Technical university of Kosice, Letná 9, 040 01 Košice, E-mail: maria.vircikova@tuke.sk, www.ai-cit.sk
[2] Center for Intelligent Technologies, Department of cybernetics and artificial intelligence, Faculty of electrical engineering and informatics, Technical university of Kosice, Letná 9, 040 01 Košice, E-mail: peter.sincak@tuke.sk, www.ai-cit.sk

body and often we attach greater weight to them than to the rational justifications while deciding or doing something. How is it in case of cooperating with machines? Should machines remain creatures of pure logic? Should not they be transformed towards creatures of emotions? And, as a result, would it drift towards intuitive mutual cooperation man-machine?

Many psychological scientists and behavioral neuroscientists affirm that emotion influences thinking, decision-making, actions, social relationships, well-being, and physical and mental health in human. Inspired by the psychological models of emotions, researchers in artificial intelligence and cognitive robotics have begun to recognize the utility of computational models of emotions for improving complex, interactive programs. At this point we notify that the whole process of representing emotions as mechanisms and functions for implementation in machines is approximate. Interacting with agents that have a model of emotions can form a better understanding of the user's moods, emotions and preferences and can thus adapt itself to the user's needs. Software agents may use emotions to facilitate the social interactions and communications between groups of agents and this way they can help in coordination of tasks, such as among cooperating robots. Moreover, synthetic characters can use a model of emotion to simulate and express emotional responses, which can effectively enhance their believability. Furthermore, emotions can be used to simulate personality traits in believable agents.

1.1 Emotional machines

Emotion (or emotional) machine is a term usually given to a software or a hardware equipped with the ability to recognize or/and express emotions. Some authors like Picard[34] talks about "machines that have emotions" but considering the philosophical ambiguity of this collocation and the current state of the art of the applications which are still far from trustworthy imitation of human emotional intelligence, it sounds a part of the sci-fiction vocabulary.

Nevertheless, a wide spectrum of existing projects and applications try to better understand human emotional behavior and make a model according to their needs and expectations, implement this model to machines that interact with people. Such projects believe that during the process of machine migration to the human society they will be considered beneficial and intuitive partners. The future cooperation between machines and us how we believe it will be can be summed up with these words: machines fully adapting to man – that man no longer has to adapt his behavior to machines.

In academic circles and laboratories first attempts and ideas to build emotional machines are being grown for about twenty years now. For example, computers which "feel out" when their human companion is nervous and in such case they provide solutions to complete a task easily. Or robots which,

when detect fatigue of their human partner, they offer him cup of coffee – after they previously have observed this human habit. Thereunto, robotic toys that can express emotions and in this way they improve mood of lonely elder people.

Humanoids are of a similar morphology like humans – they are constructed to communicate in a manner that supports the natural communication modalities of humans by – for example facial expressions, gestures, body postures, motion patterns, gaze direction or voice.

The design of computational model of emotions is not based only on computing technology, but is closely linked to the research findings of many areas and disciplines that study human emotional processes.

1.2 Interest of computer science in emotional technology

During the last decade, new findings of research in emotions from the areas of psychology and neuroscience have attracted the attention of scientists from the circles of computer science and artificial intelligence ([8], [33], [18]). The group supporting the belief that emotions play a key role in human cognitive processes is being extended, emphasizing its importance in problem solving and decision making processes. In particular, artificial intelligence, an area dedicated to modeling and simulation of cognitive processes, denotes a growing interest in emotions. According to [15], emotions present decisive element in modeling perception, learning, decision making, memory and resulting behaviors.

The area of computer science has essentially two, sometimes over-lapping, sub-areas which study and develop emotion technology – Human-Computer Interaction (HCI) and a newer area of Computational Models of Emotions (CME).

HCI is a broad scientific discipline focusing on the interaction between man and machine, exploring ways to improve their relationship and cooperation. One of many current aims of the HCI is the development of engineering tools with the ability to measure, model and respond to emotions by developing algorithms, sensors or other software and hardware.

For the most part, CME deals with the construction of biologically inspired mechanisms that simulate aspects of human emotional functions, such as the evaluation of emotional stimuli, the activation of certain types of emotions or the creation of emotional responses.

Picard [35] proposes four main motivations for giving machines certain emotional abilities:

1. to build robots and synthetic characters that can emulate living humans and animals – for example, to build a humanoid robot,

2. to make machines that are intelligent, even though it is also impossible to find a widely accepted definition of machine intelligence,
3. to try to understand human emotions by modeling them and
4. to make machines less frustrating to interact with.

The research field dealing with the issues regarding emotions and computers called „affective computing" is an emerging, but promising area.

2 Implementation of emotional models

As said before, research of systems with an internal architecture based on emotions has experienced many different approaches, e.g. ([25], [42], [1], [19], [12], [41]). In general, these studies focus on the computational architectures which are inspired biologically – by models of emotional processes studied particularly in neurosciences. Their common goal is to integrate emotions into machine control processes ([25],[42],[1], [19]) and the evolution of artificial, synthetic emotion [23][41]. The mayor hypothesis below these systems is that the incorporation of an emotional model improves the machine performance – its decision making, action selection, and management of behaviors, autonomy and the interaction with people.

2.1 Definition(s) of emotion

There are three levels associated with feelings that can be distinguished depending on time they last: emotion, mood and personality. Emotions reflect short-term impact – they weaken and disappear. Moods are of medium length – lasting longer than emotions and they have greater impact on human cognitive functions. In [30] the mood is calculated as follows:

$$mood = \begin{Bmatrix} positive : if \sum_{i=-n}^{-1} I_i^+ > \sum_{i=-n}^{-1} I_i^- \\ neg : otherwise \end{Bmatrix} \quad (1)$$

where I_i^+ stands for the intensity of positive emotions in i and I_i^- for the intensity of negative emotions in time i.

The third level is personality. It reflects individual differences of mental characteristics. Personality may evolve over time, but it is a relatively permanent structure.

Approaches of modeling emotions differ from one another. If we seek the uniform concept for the modeling of emotions which result to artificial,

simulated emotions, we probably fail immediately when trying to clarify basic terms.

Table 1. Definitions of emotion

Author(s), year	*Definition*
James & Lang, 1884	Emotion is a response to psysiological changes.
Arnold, 1960	An emotion is a tendency toward an object intuitively judged as good (beneficial) or away from an object intuitively judged as bad (harmful). This attraction or aversion is accompanied by a pattern of physiological changes for approaching or avoiding.
Solomon, 1973	Emotions are judgements characterized by their temporal mode and their content evaluation; emotions are also strategic choices in the aim to protect oneself and to increase respect of oneself (inspired by Sartre's theory, 1938
Greenspan, 1988	Emotion is a conscious mental process affecting a major component of the body; it also has a lot of influence on one's thought and action, notably to plan social interaction strategie
Ortony and Turner, 1990	Emotions are valence reactions to events, agents or objects
Lazarus, 1991	He highlights that appraisals are necessary and sufficient for emotion. Adding that the notion of coping allows an individual to choose strategies to confront future problem
Scherer, 2005	In general, emotion can be seen as a sort of process that involves different parts, including subjective feeling, cognition, physical expression, the tendency of action or desires, and the neurological process.
Descartes et al. (de Sousa, 2008)	Emotions are made up of primary emotions and are measured in function to a limited number of finite dimensions (ex. level of stimulation, intensity, pleasure or aversion, one's own intention or that of others, etc.)

Searching for the unified definition of artificial emotion is like searching for the definition of artificial intelligence in the sense that, even psychologists do not agree with a single, coherent definition of emotion in humans. Literature often gives many different definitions, some of them [44] demonstrating the attempt of their categorization. In some aspects most of the theories agree on

that in general the emotion is understood as a state of feelings, including thoughts, physiological changes and, subsequently, the external manifestations in the form of expressions and behaviors. We could say that emotions interpret feelings binary – into two categories of pleasant and unpleasant. Almost all psychologists also agree with that there is a set of basic emotions and a variety of other different emotions arise from these. Table 1 illustrates some of definitions of emotions. A survey of the basic emotions as viewed by chosen psychologist is illustrated in Table 2. These psychologists assume that the basic emotions are instinctive and are on the evolutionary basis and other emotions are formed from the basic ones and can be learned by the experience.

Table 2. Psychologists and the set of basic emotions proposed by them[3]

Plutchik	Acceptance, anger, anticipation, disgust, joy, fear, sadness, surprise
Ekman Friesen Ellsworth	Anger, disgust, joy, sadness, surprise
Frijda	Desire, happiness, interest, surprise, amazement, sadness
Izard	Anger, contempt, disgust, anxiety, fear, guilt, interest, joy, shame, surprise
James	Fear, grief, love, anger
Mowrer	Pain, joy
Oatley Johnson	Anger, frustration, anxiety, happiness, sadness

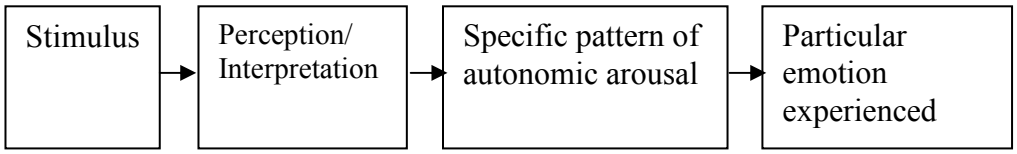
Not only are there many different definitions of emotions, but on the other hand, also a great number of multidisciplinary perspectives. The concept of emotion has its place not only in psychology, but also in neuroscience, philosophy, cognitive informatics and at present time it increasingly resonates in computer science. Many new formulations of theoretical, cognitive and computational models are being developed.

For example, several different theories about the process which leads to some emotional state were designed as illustrated on Figure 1. The theory of James-Lang states that the event raises physiological arousal, which is then interpreted. After the interpretation of this arousal, we are able to live the emotion. The Cannon-Bard theory says that the physiological arousal and the emotion are simultaneous.This theory does not describe the role of thoughts like

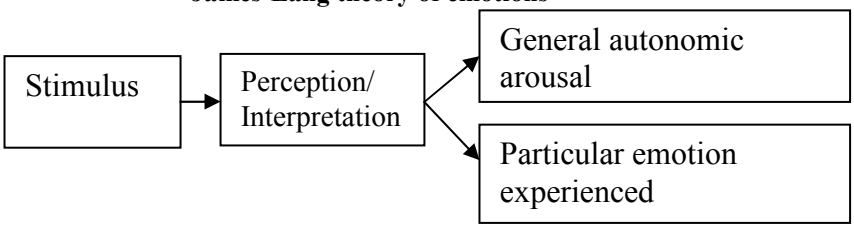
[3] Set of basic, fundamental emotions. Other emotions can be obtained as a mix of them.

the Schachter-Singer theory does. According to it, the event causes an arousal for which there must be some reason The Facial-feedback theory describes changes in facial muscles – for example, a smile reflects joy.

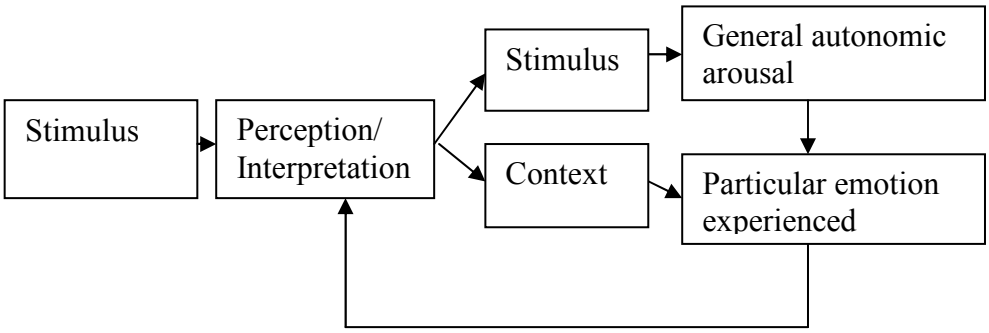
Ekman studied emotions that corresponded with the facial expression – traveling round the globe he figured out the basic emotions (see Table 1, line 2). The main feature of his set of the basic emotions is the fact that the resulting facial expressions are not subject of culture, but are universal – not depending on geographic places, religions, languages, genders, etc.



James-Lang theory of emotions



Cannon-Bard theory of emotions



Although it may be impossible to categorize emotions within crisp boundaries, it can be liken to the color palette. Like from the three basic colors – red, green and blue – many different other hues can be generated – from the basic emotions many other emotions with diverse intensities can be obtained. This is the elementary idea of Plutchik's psychoevolutionary theory of emotions, which was based on the theory of colors. The term 'psychoevolutionary' derives from his proposal that the primary emotions

activate different behaviors in the context of survival. As can be deducted from Table 3., for example – fear is an emotion causing vomiting in living organisms and thus it helps prevent poisoning. Like a painter mixes his basic colors to obtain any color of the entire color spectrum – mixing the basic emotions all the emotion spectrum can be mingled.

Table 3. From stimulus to resulting behavior by Plutchik.

Stimulus (event)	*Cognition*	*Emotional state*	*Resulting Behavior*	*Consequence*
Threat	Danger	Fear	Escape	Safety
Obstacle	Enmity	Anger	Attack	Destruction of the obstacle
Obtain valuable object	Occupation	Joy	Maintain/ Repeat	Acquisition of resources
Loss of a valuable object	Leaving	Sadness	Cry	Re-acquistion of the lost object
Member of a group	Friendship	Acceptance	Care for the other	Mutual support
Intolerable object	Poison	Disgust	Vomit	Expulsion of poison
New territory	Exploration	Anticipation	Mapping	Obtain knowledge
Unexpected event	Curiosity	Surprise	Stop	Acquisition of time needed for orientation

From the psychological point of view the emotional processes and states are so complex and can be analyzed from so many angles that to build a complete picture of emotions is virtually impossible. However, if we want to create artificial intelligent systems, is it necessary to understand these complex processes in humans and to what extent? Do we need it for simulation of intelligent machine problem solving? Minsky [29] thinks that the emotion from the perspective of artificial intelligence is nothing special saying that every emotional state is just a different style of thinking.

We can divide the modeling approaches in two kinds, in general. Depending on the degree of abstraction of intervening variables, there is the Black Box modeling and the second one is Process modeling. Models of the Black Box provide little information concerning the mechanisms involved, but they are useful for practical decision making and for providing a sound

grounding for theoretical and empirical study [45]. The purpose of process modeling is usually the attempt to simulate naturally occurring processes using hypothesized underlying mechanisms. As said by [45], given the complexity of involved components, models of this kind seem to be as difficult to realize as they are useful to increase our knowledge.

Emotions should have their own temporal dynamics, as proposed by Fellous [14]. The implementation of emotions as states does not simulate their way of emergence – how they increase of decrease in intensity. One of the most appealing limitations of many approaches which try to simulate emotional processes is their inability to adapt. Most of the previously developed computational models of emotions were designed in a way that it responds in a pre-determined manner to concrete situations. Psychology shows the importance of memory and experience in the emotional process. Exceedingly artificial intelligence tries to design systems which do not have pre-programmed behavior, but rather learn from experience. Methods of artificial intelligence could be appropriate tools for modeling such adaptive emotional behaviors, as discussed in chapter 3.

Picard [35] argue that even without a precise definition, one can still begin to say concrete things about certain components of emotion, at least based on what is known about human and animal emotions. Of course much is still unknown about human emotions, so we are nowhere near being able to model them, much less duplicate all their functions in machines. Also, all scientific findings are subjects to revision – history has certainly taught us humility, that what scientists believed to be true at one point has often been changed at a later date. Just as it is not necessary to fully understand the experience of seeing a color to map the neural dynamics of color vision, it is not necessary to fully understand the conscious experience of joy, sadness, fear, or anger to map the neural dynamics of those emotions (even more, the neural dynamics of positive and negative affect in general).

3 Computational intelligent techniques for modeling emotions

This chapter tries to demonstrate the convenience of using computational intelligence to deal with issues which may occur when designing an internal model of emotions in artificial systems.

3.1 Neural networks for modeling processes that involve emotion

The ability of neural networks to learn can be useful for example in learning different aspect about/from the environment – the association of objects, the sequences of events or the users'expectations. It can help the system to adapt dynamically, what increases its trustworthiness and improves its interaction

with humans. Moreover, emotions are associated with typical patterns of behaviors and neural processes characteristic for specific brain regions. As said by Levin [26], emotions correspond to modeling by neural networks as any other psychological process, which was ever simulated using the neural networks. Particularly useful can be to understand the complex interactions of emotions and cognitive processes such as attention, memory and decision making. Computational neural networks have been used for modeling of cognitive and behavioral processes that involve emotion in various works.

Several applications of neural networks concerning the emotion modeling are in [26]:

- Models of emotional influences on attention
- Models of emotionally influenced decision making
- Models of specific emotions
- Models of emotional disorders

Sharada & Ramanaiah [42] develop an emotional intelligent agent architecture based on a Neuro-Fuzzy system as event processor and a Hopfield network as emotional state calculator. The output layer of the event processor generates a numeric pattern according to the input event. This pattern is taken up by the Hopfield network for emotional state calculation.

Neural networks have accomplished successful results in tasks of pattern recognition in general. Emotions have been recognized in speech, facial expressions, as well as in motion patterns.

3.2 Capturing the fuzziness of emotions using fuzzy logic

Fuzzy logic is known for its ability to capture the uncertainty and the complex character of emotions. For example in [30] fuzzy logic has shown that it can achieve smooth transitions in behaviors with a relatively small set of rules.

It is clear that emotions are vague, thus it is impossible to construct a credible emotional model in which some discrete values would be associated with different emotional states without any continuous transition between each other. One can be sad to some extent – in the word of fuzzy logic, one can be sad with an appropriate level of membership to the set of sadness. Fuzzy logic provides an expressive language to work with both quantitative and qualitative (linguistic) descriptions of the model and it allows the model to produce complex emotional states or the output behaviors.

Nasr [30] introduced three concepts concerning fuzzy systems involving emotions:

- fuzzy goals representing the degree of success and failure to achieve goals,
- fuzzy memberships of events to goals, as an event can influence two or more goals with different memberships and
- fuzzy mapping used to map the mixture of emotions on behaviors.

His system FLAME represents emotions using fuzzy sets and maps events to emotions and emotions to concrete outputs – behaviors using fuzzy rules.

The example of rule for behavior selection can be:

IF *emotion_1 is A1* ***AND*** *emotion_2 is A2 ....* ***AND*** *emotion_k is Ak*
AND *event is E* ***AND*** *reason is (E, R)*
THEN *BEHAVIOR is B*

where *k* is the number of involved emotions, *A1, A2* and *Ak* are fuzzy sets which define the emotional intensity (high, medium, low). The behaviors are represented as singletons (discrete states) and the selected resulting behavior is the one with the maximum value.

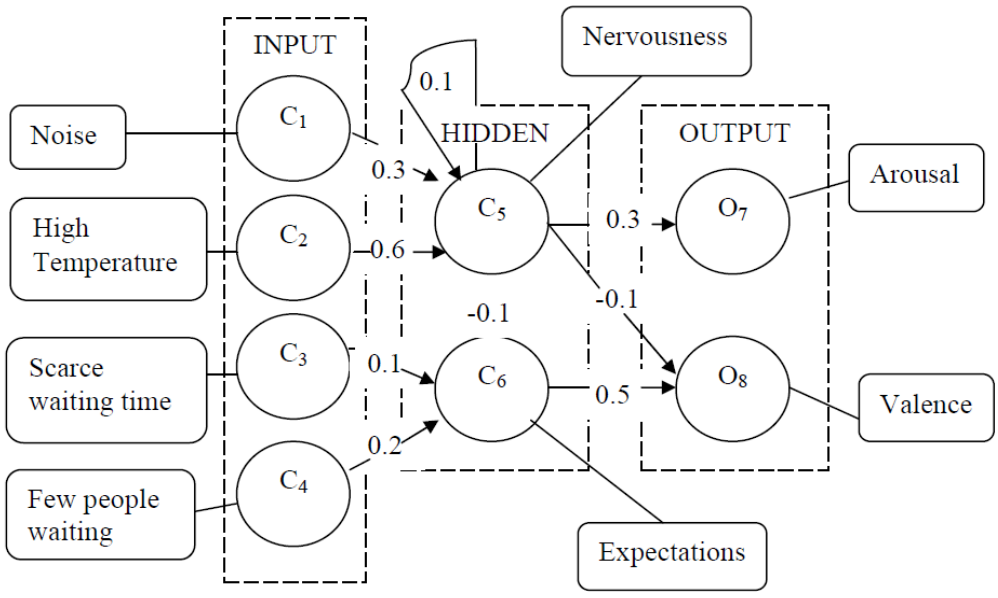


Figure 2. Fuzzy Cognitive Map emotion forecasting by [40]: A three-layered architecture.

Salmeron[40] uses Fuzzy Cognitive Maps (FCM), supervised learning fuzzy-neural systems, for forecasting artificial emotions from sensors´raw data in autonomous systems integrated into complex environments with high uncertainty. FCM are signed fuzzy weighted digraphs, usually involving feedback composed of nodes indicating the most relevant factors of a decisional environment and edges between representing the relationship between each other. He states that the FCMs can indicate the relationships between the environmental variables and the emotions as well as can make what-if simulations and forecasting emotions according to previous environmental conditions.

3.3 Evolutionary computation for the evolutionary character of emotions

The family of evolutionary algorithms also seems to involve interesting techniques for creation a trustworthy emotional model. Darwin and then also for example Plutchik [36] suggested that human emotions are of evolutionary character.

The artificial chromosome in [24] presents the key component for defining personality, which elements are being inherited. The advantage of this technique consists in providing an evolutionary nature – by the process of an artificial reproduction. Secondly, to some extent, they are able to simulate the phenomenon of emergence of emotions by the tool of mutation. If using evolutionary algorithms to design the system, an appropriate representation of the individual must be chosen and also the fitness function to evaluate the individuals must be clear defined.

One example of such architecture is MOEGPP, consisting of various blocks: perception, internal state and behavior, which defines the personality. The weights which connect these blocks (perception → internal state and internal state → selection of behavior) are encoded as genes representing the unique tendencies of the system "to feel" happy, sad, angry, etc. and the inherited way of the behavior depending on the input stimuli.

The artificial system in [24] is build from chromosomes C_k and each of the C_k has three such called gene vectors: a **F**undamental vector x_k^F, vector of the **I**nternal state x_k^I and a vector representing the **B**ehav**i**or x_k^B. The fundamental vector is represented by constant values and penalized factors in updating the internal state. The other two vectors represent the weights.

$$G = [C_1 \mid C_2 \mid \ldots \mid C_c], \text{ where } C_k = \begin{bmatrix} x_k^F \\ x_k^I \\ x_k^B \end{bmatrix}, \; k = 1, 2, \ldots, c \qquad (2)$$

$$x_k^F = \begin{pmatrix} x_{1k}^F \\ x_{2k}^F \end{pmatrix},\ x_k^I = \begin{pmatrix} x_{1k}^I \\ x_{2k}^I \\ . \\ . \\ . \\ x_{yk}^I \end{pmatrix},\ x_k^B = \begin{pmatrix} x_{1k}^B \\ x_{2k}^B \\ . \\ . \\ . \\ x_{zk}^B \end{pmatrix} \tag{3}$$

where y and z determine the size of the vectors.

The resulting genome as illustrated in Figure 3. is composed of fourteen chromosomes (C_1, C_2, ..., C_{14}), where:

- chromosomes C_1 – C_6 are connected to the module of motivation: C_1 – curiosity, C_2 – intimacy, C_3 – monotony, C_4 – avoidance, C_5 – greed, C_6 – control, C_7 – fatigue,
- chromosomes C_8 – C_{11} are connected to the module of homeostasis: C_8 – sleepiness, C_9 –hunger, C_{10} – happiness and C_{11} – sadness and
- chromosomes C_{12} – C_{14} are connected to the module of emotion: C_{12} – anger, C_{13} – fear, C_{14} – neutral state.

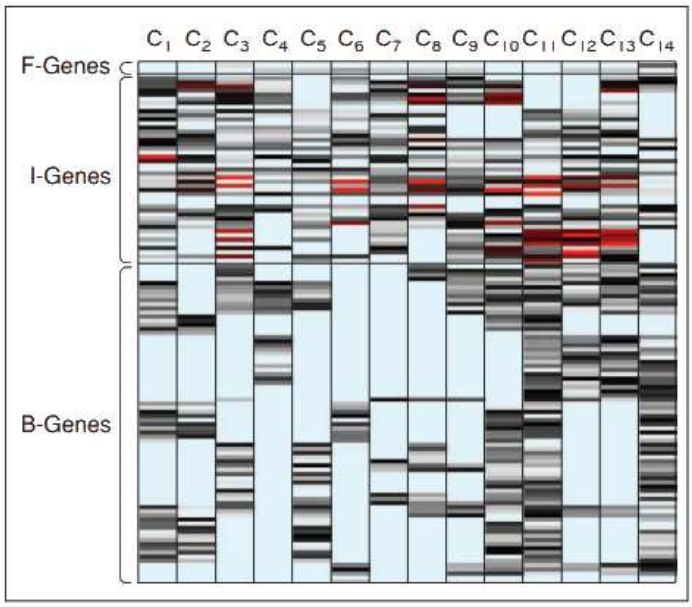


Figure 3. Resulting genome [24] composed from 14 chromosomes (C_1 – C_{14})

Table 4. Values of fitness functions of personality types which assigned by users.

Internal state		*AGREE-ABLE*		*ANTAGO-NISTIC*		*EXTRO-VERTED*		*INTRO-VERTED*		*CONSCIE-NTIOUS*		*NEGLI-GENT*	
mode	*state*	ψ_{1k}^{I}	ψ_{1k}^{B}	ψ_{2k}^{I}	ψ_{2k}^{B}	ψ_{3k}^{I}	ψ_{3k}^{B}	ψ_{4k}^{I}	ψ_{4k}^{B}	ψ_{5k}^{I}	ψ_{5k}^{B}	ψ_{6k}^{I}	ψ_{6k}^{B}
Motivation	curiosity	0.5	0.5	0.2	**0.8**	**0.8**	0.2	0.2	0.2	0.2	0.2	0.1	0.1
	intimacy	**0.8**	**0.8**	0.2	0.2	0.85	0.85	0.2	0.2	0.2	0.2	0.1	0.1
	monotony	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**	0.2	0.2	0.2	0.2	**0.8**	**0.8**
	avoidance	0.2	0.2	**0.8**	**0.8**	0.2	0.2	**0.8**	**0.8**	0.2	0.2	**0.5**	**0.5**
	greed	0.2	0.2	**0.8**	**0.8**	**0.5**	**0.5**	0.2	0.2	**0.5**	**0.5**	0.2	0.2
	control	0.1	0.1	**0.7**	**0.7**	**0.85**	**0.85**	0.2	0.2	0.2	0.2	**0.6**	**0.6**
	fatigue	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.1	0.1	**0.5**	**0.5**
Homeostasis	drowsiness	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.1	0.1	**0.5**	**0.5**
	hunger	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.3	0.3
	happiness	**0.8**	**0.8**	0.2	0.2	**0.65**	**0.65**	0.3	0.3	**0.65**	**0.65**	0.2	0.2
	sadness	**0.5**	**0.5**	**0.5**	**0.5**	0.2	0.2	**0.6**	**0.6**	0.3	0.3	0.2	0.2
Emotion	anger	0.2	0.2	**0.8**	**0.8**	0.2	0.2	0.3	0.3	0.1	0.1	**0.6**	**0.6**
	fear	0.2	0.2	**0.8**	**0.8**	0.2	0.2	0.3	0.3	0.1	0.1	**0.6**	**0.6**
	neutral	**0.5**	**0.5**	0.2	0.2	0.1	0.1	**0.7**	**0.7**	0.1	0.1	0.2	0.2

The operator of mutation of the *j*-th *I*-gene of the *k*-th chromosome in *i*-th genome which heads to the goal *l* and which minimum is in λil is defined as follows:

$$\begin{aligned} x_{ikj}^{I} &= x_{ikj}^{I} + N\left(\mu_{ikl}^{I}, \sigma_{il}^{I}\right) \\ \mu_{ikl}^{I} &= \gamma * \left(\overline{\psi}_{kl}^{I} - \phi_{k}^{I}\left(T_{s}, G_{i}\right)\right) \\ \sigma_{il}^{I} &= \frac{\kappa * \sqrt{f_{l}\left(T_{s,} G_{i}\right)}}{v} \end{aligned} \tag{4}$$

where $N\left(\mu_{ikl}^{I}, \sigma_{il}^{I}\right)$ is a biased normal distribution and the average value of the μ_{ikl}^{I} is defined as the difference between normalized preference value of the l th objective and the evaluated possession ratio. $f_{l}\left(T_{s,} G_{i}\right)$ is the selected objective value, γ and κ are the scaling factors for the mean and standard deviation of normal distribution, respectively, and ν is the number of gene.

We think that the personality is crucial in building a reliable emotional agent. The use of evolutionary algorithms makes possible to generate specific personalities along with diverse personalities.

4 Survey of computer emotional models

During the history of artificial intelligence many attempts tried to make models describing the human mind and some of these involved emotional processes. Probably one of the first famous projects interested in emotions in artificial intelligence was the work of Simon, which around 1967 constructed a model based on the motivational states like hunger and thirst. The simulation consisted in the following process: if the level of hunger achieved some level, the process of thinking was interrupted. Several projects in which emotions were important for decision making processes or for the communication with robots appeared in 90-ties (e.g Sugano a Ogata 1996, Shibata et al. 1996, Breazeal, Brooks).

Table 5. Comparison of the projects of cognitive models of emotions according to the processes they involve

Name of the model	*Processes involved*
EMA	Emotions, Mood
Kismet	Emotions, Mood, Personality, Basic needs
Flame	Emotions, Mood, Basic needs
MAMID	Emotions, Personality
Alma	Emotions, Mood, Personality
MADB	Emotions, Mood, Basic needs, Attitudes
FAtiMA	Emotions, Mood, Personality
Cathexis	Emotions, Mood, Personality, Basic needs

Following systems contain in some form an emotional mode listed depending on the year when they appeared. Figure 4. presents their classification.

CogAff (Cognition and Affect Project **-** CogAff Architecture, 1995), developed at the University of Birmingham. It is a three-level architecture which tried to model an agents´ behavior primarily designed for simulation and game industry. They used a concept of primary, secondary and tertiary emotions.

FLAME **(**Fuzzy Logic Adaptive Model of Emotions, 2000**)** [30], developed at the Northwestern and Texas University. It is a computational model of emotions, which can be incorporated to intelligent agents and other

complex interactive programs. For demonstration purposes authors used a computer simulation of a pet, in which the adaptive components represent a basis for evaluation of the interaction by users. This model involves also learning – in three different forms: the association of an emotion and an object which triggered the emotion in past, reinforcement learning to estimate the event depending on the agent´s goals and the probabilistic approach for learning patterns of events and the heuristic approach for learning of actions which were pleasant or unpleasant for the agent as well as for the user.

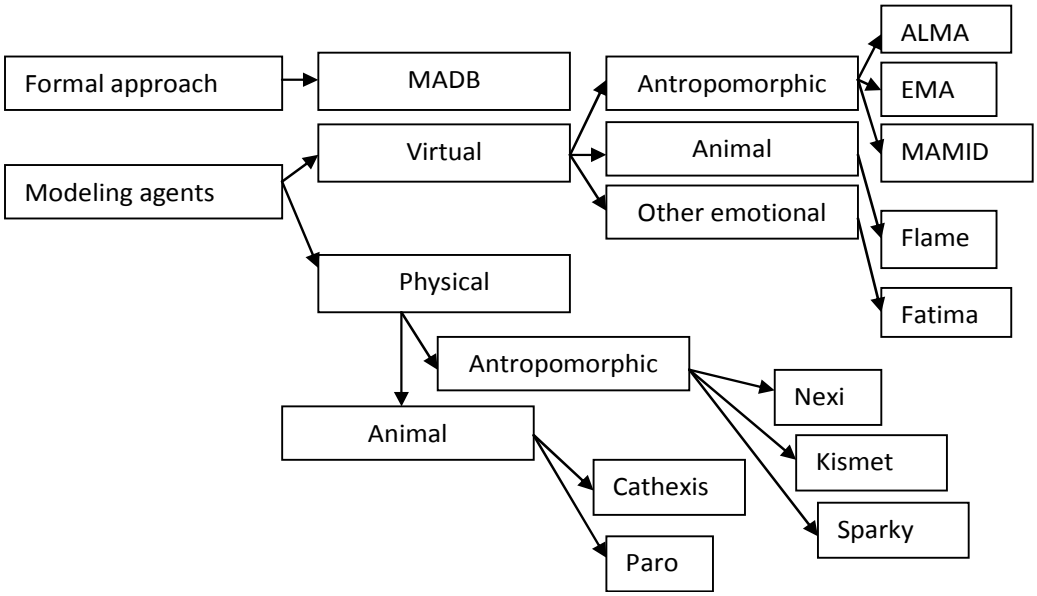


Figure 4. Classification of the emotional models

ParlE (Adaptative Plan Based Event Appraisal Model of Emotions ParleE, 2002) [5] is a quantitative, flexible, adaptive model of emotions for a conversational agent in a multi-agent environment which has multimodal communication capacities. This model assesses events based on learning and a probabilistic planning algorithm. It also models personality, as well as motivational states and their role in determining the manner in which the agent experiences emotions. Rousseau's model of personality (Rousseau, 1996) is used in this particular model, thus classifying personality into the different processes that an agent can carry out: perceiving, reasoning, learning, deciding, acting, interacting, revealing, and feeling - all the while showing emotion. However, the model lacks specifications of the exact influence of emotions on a planning process. Furthermore, the components of models of other agents seem to make the model not quite as flexible as the authors supposed.

Kismet (MIT, late 90-ties) is considered the first robot (robotic head, in fact) expressing emotions. It simulates social interactions between people and based on this idea the robot had it´s own robotic teacher – an assistant helping him achieve better communication skills. Emotions are modeled from the functional perspective (7 basic emotions – anger, disgust, joy, sadness, serenity and surprise). The successors of Kismet on MIT are other humanoid robots like Nexi, Cog or Leonardo.

ALMA (A Layered Model of Affect, 2005) is an architecture that integrates three major affective characteristics: emotions, moods and personality that cover short, medium, and long-term affect. The use of this model consists of two phases: In the preparation phase appraisal rules and personality profiles for characters must be specified. In the run-time phase, the specified appraisal rules are used to compute real-time emotions and moods as results of a subjective appraisal of relevant input. They use affective states to color simulated dialogs and through verbal and non-verbal expression of emotions.

MAMID [22], 2005 is a cognitive-affective integrated symbolic architecture aimed at emulating aspects of human information processing, with particular focus on the role of emotion in decision-making. To this end, it models the cognitive appraisal process to dynamically generate emotions in response to incoming stimuli, and then models the subsequent effects of these emotions on distinct stages of decision-making.

EMA (EMotion and Adaptation, 2006) [28] is a computational model of emotion which is special in a single and automatic appraisal process that operates over a person's interpretation of their relationship to the environment. Dynamics arise from perceptual and inferential processes operating on this interpretation (including deliberative and reactive processes).

MADB (2007) [45] model provides a formal explanation of the mechanism and relationship between motivation, attitude, and behavior. The model can be used to describe how the motivation process drives human behaviors and actions, and how the attitude as well as the decision-making process help to regulate the motivation and determines whether the motivation should be implemented.

MOEGPP (multi-objective evolutionary generation process for artificial creatures' specific personalities, 2008) [24] is a model where the dimension of the personality model is defined as optimization objectives. In MOEGPP, the objectives are set as agreeable, antagonistic, extroverted, introverted, conscientious and negligent models by employing Big Five personality dimensions. An artificial creature is created as an autonomous one, which behaves depending on his internal state. This is composed of motivation, emotion and homeostasis, influences by perception, referring to the knowledge stored in memory and considering the context of the environment. The manner of response to the percepts depends on its personality.

GRACE (Generic Robotic Architecture to Create Emotions, 2008) [11] is presented like a generic model which defines its emotional process as a physiological emotional response triggered by an internal or external event. It is characterised by seven components applying the appraisal, coping, Scherer, and personality theories. Being generic lets it incorporate the functionalities of all of the above-sited models. Moreover, it integrates an ”Intuition” component, which does not exist in the other models, which allows it to obtain unforeseeable emotional reactions.

MAPH (Project Active Media For Handicap, 2008) [11] is a project whose objective is to give comfort to vulnerable children and/or those undergoing long-term hospitalization with the help of a robot which can be used as an emotional companion. As the use of robots in a hospital environment remains limited, we have decided to opt for simplicity in the robotic architecture, thus in the emotional expression as well. A component of the MAPH project, aims at maintaining nonverbal interaction with children between four and eight years of age. The project is essentially made of three main interdependent parts: Recognition and understanding of a child’s spoken language, Emotional interaction between the child and the robot and Cognitive interaction between the child and the robot.

FATIMA (Fearnot AffecTIve Mind Architecture, 2011) [13] is a generic and flexible architecture for emotional agents, with what we consider to be the minimum set of functionalities that allows us to implement and compare different appraisal theories in a given scenario. Modular, the architecture proposed is composed of a core algorithm and by a set of components that add particular functionality (either in terms of appraisal or behavior) to the architecture, which makes the architecture more flexible and easier to extend.

Feelix Growing (FEEL, Interact, eXpress: a Global appRoach to develOpment With INterdisciplinary Grounding) [4] is a current project funded by the European Commission. The overall goal of this project is the interdisciplinary investigation of socially situated development from an integrated or global perspective, as a key paradigm towards achieving robots that interact with humans in their everyday environments in a rich, flexible, autonomous, and user-centred way. One of the main goals of the Feelix Growing project is the investigation of the roles of emotion, interaction, expression, and their interplays in bootstrapping and driving socially situated development, which includes implementation of robotic systems that improve existing work in each of those aspects, and testing in the key identified scenarios. Afterwards they think of the integration of the above capabilities in at least two different robotic systems, and feedback across the disciplines involved.

5 Engineering applications of emotional technology

The different studies in human-robot interaction, due to [39], focus on two major aspects: psychological robotics: studies on the behavior between humans and robots and robotherapy: use of robots as therapeutic companions for people suffering from psychological or limited physiological problems.

Robotherapy is defined as a framework of human-robot interaction with the goal to reconstruct a person's negative experiences through the development of new technological tools to create a foundation on which new positive ideas may be constructed It offers a methodological and experimental concept which allows stimulation, assistance, and rehabilitation of people with physical or cognitive disorders, those with special needs, or others with physiological disabilities. Its goal is to build a robot companion for the rehabilitation and comfort of children with physical or cognitive disorders. Research has allowed for numerous robot companions having such a purpose to be created. This novel idea is based on the works carried out on a robot with a very simple architecture, but maximal expresivity

Emotion in robotics, especially multi-agent robotics, plays a vital role in representing and characterizing robots' own behavior. Chakraborty in [7] presents this example: suppose an agent fails to grip an object because the object-size, measured by its diameter, is too large to be gripped by the robot. Alternatively, suppose a robot fails to grip an object because of slip, the agent under these contexts can represent its features by generating appropriate tone indicative of its failures. Different instrumental tones indicating various situations faced by the robots can be used to describe its failures or successes or to get rescued when it generates an alarming tone to its teammates. The oscillation of the robot's gripper when sustained for a long time indicates a failure of the robot in correctly positioning its arm. A teammate of the robot may notice this failure by its own camera and can provide necessary supports to rescue the partner in trouble.

According to [4] robots to be truly integrated in humans' every-day environment in order to provide services such as company, care-giving, entertainment, patient monitoring, aids in therapy, etc., they cannot be simply designed and taken off the shelf to be directly embedded into a real-life setting. Adaptation to incompletely known and changing environments and personalization to their human users and partners are necessary features to achieve successful long-term integration. This integration would require that, like children (but on a shorter time-scale), robots develop embedded in the social environment in which they will fulfil their roles. Involving emotional technology this could be easier to achieve.

We focus on the applications of robotics, but emotions find their place also in other various areas (the big domain is Game Industry), for example supporting the development of collaborative virtual environments. In [1] authors explore the augmentation of collaborative virtual environments with simple networked haptic devices to allow for the transmission of emotion through virtual interpersonal touch.

Also in the applications which involve conversational characters, a common understanding was that characters engage in a face-to-face conversation style with the user. In order to increase the believability of the virtual conversation partner, researchers have begun to address the modeling and emulation of human-like qualities such as personality and affective behavior. As a next step, systems with multiple characters have also been proposed by others. In this case, when focusing on multi-party conversations (rather than performing physical actions), emotions can be used in the dialog generation processes to inform the selection of dialog strategies and linguistic style strategies.

Emotion recognition technology was successfully used as a fear-type emotions recognition got audio-based surveillance systems, detection of children´ s emotional states in a conversaional computer games, real-life emotion detection within a medical emergency call centre, or as a robotic guard in prison, having software that analysed behavioral characteristics of inmates to help in deciding whether to alert the human guards. High-tech clothing with embedded biosensors and an Internet connection that can respond to user´s mood has wide area of application potential..

Zhou [47] proposes a framework of Emotion-Aware Ambient Intelligence (AmE). It integrates Ambient Intelligence, affective computing, emotion-aware services, emotion ontology, service-oriented computing, and service ontology. It provides an open environment for developing and delivering applications that include emotion-aware services.

Marreiros [26] proposes an architecture for an ubiquitous Group Decision Support System (u-GDSS) able to hold up asynchronous and distributed computational services. The proposed system will support group decision making, being available in any location, in different devices and at any time. One of the critical components of this framework is an emotional multiagent-based group decision making processes simulator.

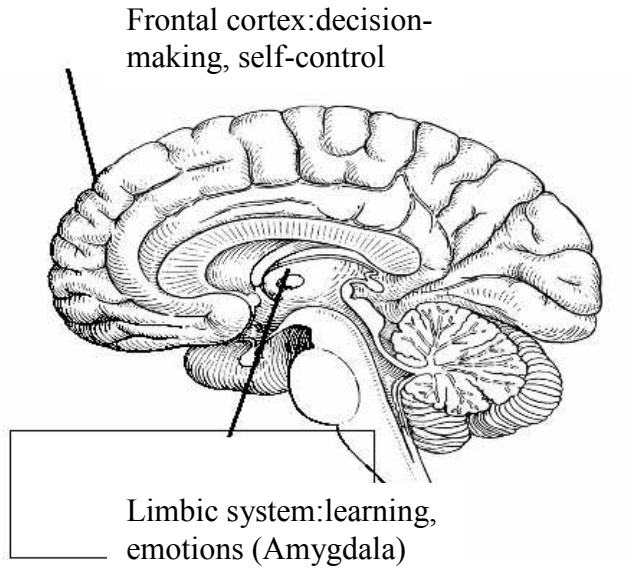


Figure 5. Neuroscience suggests that in the amygdala, the mechanisms through which emotion modulates memory and decision making may be inseparable.

6 Quo vadis artificial emotions?

Research questions concerning artificial emotion try to ask a question whether incorporation of emotions would be useful for example in tasks of: action selection, adaptation, social grounding, sensory integration, alarm mechanism, goal management, learning, focus of attention, memory control and internal models. The level of complexity of a system involving of these would require a large project which would prove or not this hypothesis.

Which are the most common reasons for the implementation of emotional intelligence to the artificial systems?

Wehrle in [45] summarises the motivational issues for modeling emotion-based artificial systems dividing depending on their key criteria into three areas:

1. Engineering, where the key criteria is the performance of the systems and thus, the main motivation is building good artifacts for a concrete task;
2. Human-Computer Interaction, where the principal criteria consists in performance and also acceptance and usability of the system – so the main motivation is to improve the human-computer interaction;

3. Science encompassing psychology, neuroscience, cognitive science and biology among others, where the critera is description, explanation and prediction. The principal motivation embodies in improving our knowledge about the nature of emotion and its implications.

A notable area where emotions are useful is the communication, for example to refer the internal state of an agent. Whether it comes to the facial expressions on the robotic head, verbal expressions or some kind of motion, emotions (or expressing emotions) serve like compact messages between individuals – even if one of them is a human and the other one is a robot. We believe that expressing emotions increase the effectiveness of communication between humans and machines.

Damasio [10] divided the tasks of emotions into four categories:
1. guide to information,
2. selective attentional spotlight,
3. motivator of behavior and
4. common currency for comparing alternatives

According to that emotions are of the motivational character, they could represent also a support for the autonomous behavior. Emotions have great influence on cognition and behavior of the people in all kinds of situations. According to [45], positive emotions often signal that activity toward the goal can terminate, or that resources can be freed for other exploits. In contrast, many negative emotions result from painful sensations or threatening situations. Negative emotions motivate actions to set things right or to prevent unpleasant things from occurring.

Neuroscience confirms the existence of interactions between amygdala and prefrontal cortex and their influence on emotion generation [15], but up to now the exact process has not been described and so the computer science cannot know how this knowledge could be used in computational intelligence and it remains an open question. It seems certain that, as we understand more about cognition, we will need to explore autonomous systems with limited resources that nevertheless cope successfully with multiple goals, uncertainty about environment, and coordination with other agents. In mammals, these cognitive design problems seem to have been solved, at least in part, by the processes underlying emotions [17].

Thus, artificial emotions could be one of the tools supporting dynamic and flexible decision-making even in the artificial agents. In [21] they compare robots without emotions to creations making decisions without passions. Not

only Minsky [29], but also other several researchers argue that cognitive processes go hand in hand with emotions and they argue against that emotions and cognition are opposites. From emotional states to goals and attachments and on to consciousness and self-awareness, he argues, we can understand the process of thinking in its intricacy.

Wehrle [45] asks a question: Whose emotions should robots have?

Do we allow the robot to establish its own emotional categorization which refers to its own physical properties, the task, the properties of the environment, and the ongoing interaction with this environment or can we put human emotion categories into an artificial agent? It might be useful to use emotions as design heuristics for adaptive systems, or to describe their behavior, but can we hope to ground these categories that have evolved in a different system?

The application potential of emotional technology in user interface systems is wide. From applications that match the mood of a car's warning voice to that of the mood of the car's driver (i.e., cheerful or sad) what decreases the accident rate compared to when there is a mismatch to complex emotional systems that in near future could turn into human everyday companions.

Acknowledgement: This work is the result of the project implementation: Development of the Center of Information and Communication Technologies for Knowledge Systems (ITMS project code: 26220120030) supported by the Research & Development Operational Program funded by the ERDF.

References

[1] Bailenson, J., N., Yee, N., Brave, S.: *Virtual Interpersonal Touch: Expressing and Recognizing Emotions Through Haptic Devices.* HUMAN–COMPUTER INTERACTION, 2007, Volume 22, pp. 325–353.

[2] Bazzan, A. L. C., Bordini, R.: *A framework for the simulation of agent with emotions: report on experiments with the iterated prisoner`s dilemma*. The 5th Int. Conference on Autonomous Agents, Montreal, New York, 2001. pp. 292-299.

[3] Bhatti, W, M., Wang, Y., Guan, L.: *A neural network approach for human emotion recognition in speech*. Proceedings of the 2004 international symposium on circuits and systems, 2004.

[4] Boucenna, S., Gaussier, P., Hafemeister, L, Bard, K.: *Autonomous Development Of Social Referencing Skills.* From Animals to Animats,

proceedings of the 11th Conference on the Simulation of Adaptive Beahvior, 2010.

[5] Bui, T. D., Heylen, D., Poel, M., Nijholt, A: *Parlee: An adaptive plan based event appraisal model of emotions*. Heidelberg (ed.), KI 2002: Advances in Artificial Intelligence, Vol. 2479 of Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 129–143.

[6] Carnegie, D.: *How to win friends and influence people.* Pocket books, 1936.

[7] Chakraborty, A.: *Emotional Intelligence*. Springer, 1998, pp. 261 – 270.

[8] Custódio L., Ventura. R., Pinto-Ferreira C: *Artificial emotions and emotion-based control systems*. Proceedings of 7th IEEE International Conference on Emerging Technologies and Factory Automation. 1999. V. 2. pp. 1415-1420.

[9] Daily, M, N. et. al.: EMPATH*: A Neural Network that Categorizes Facial Expressions*, Journal of Cognitive Neuroscience, Vol14, 8, 2006, pp. 1158-1173.

[10] Damasio, A., R.: *Descartes' error: Emotion, reason, and the human brainGrosset/Putnam,* New York, 1994.

[11] Dang, T., H.,H.., Letellier-Zarshenas, S., Duhaut, D.: *Grace generic robotic architecture to create emotions*. Advances in Mobile Robotics: Proceedings of the Eleventh International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, 2004, pp. 174–181.

[12] Davis, D. N.: *Modeling emotion in computational agents*, 2000. Available at http://www2.dcs.hull.ac.uk/NEAT/dnd/papers/ecai2m.pdf. Revised: 15.4.2012.

[13] Dias, J., Mascarenhas, S., Paiva, A.: *FAtiMA Modular: Towards an Agent Architecture with a Generic Appraisal Framework*, 2011.

[14] Fellous, J. M.: *From Human Emotions to Robot Emotion*s. Architectures for Modeling Emotions: Cross-Disciplinary Foundations." AAAI Spring Symposium, 2004. pp. 37-47.

[15] Freitas, J, S., Gudwin R., Queiroz, J.: *Emotion in Artificial Intelligence and Artificial Life Research: Facing Problems.* Lecture Notes in Computer Science, Springer-Verlag London, 2005.

[16] Frijda, N.H.: *The emotions,* Cambridge University Press, Cambridge, UK, 1986.

[17] Fredrickson, B., L.: *The Role of Positive Emotions in Positive Psychology: The Broaden-and-Build theory of Positive Emotions*. American Psychologist, Vol 56(3), Mar 2001, 218-226.

[18] Gadanho, S., Hallam, J.: *Emotion-triggered learning in autonomous robot control*. Cybernetics and Systems: an international journal. V. 32, 2001, pp.531-559

[19] Gadanho, S., Hallam, J.: *Emotion-triggered learning in autonomous robot control*. Cybernetics and Systems: an international journal. V. 32, July, 2001, pp.531-55.

[20] Gebhard, P., Kipp, K.: *Are Computer-generated Emotions and Mood plausible to Humans?* Proceedings of the 6th International Conference on Intelligent Virtual Agents (IVA'06), pp. 343-356, Marina Del Rey, USA, 2006.

[21] Hollinger G., Georgiev Y., Manfredi A., Maxwell B., Pezzementi Z., Mitchell B.: *Design of a Social Mobile Robot Using Emotion-Based Decision Mechanisms.* International Conference on Intelligent RObots and Systems - IROS - IROS , pp. 3093-3098, 2006.

[22] Hudlicka, E.: *A Computational Model of Emotion and Personality: Application to Psychotherapy Research and Practice*. Proceedings of the 10th Annual CyberTherapy Conference: A Decade of Virtual Reality, Basel, Switzerland, 2005.

[23] Izard C., E.: *The Many Meanings/Aspects of Emotion: Definitions, Functions, Activation, and Regulation.* Emotion Review, 2010, vol. 2no. 4 363-370.

[24] Kim, J. K., Le, C.H.: *Multi-objective Evolutionary Generation Process for Specific Personalities of Artificial Creature*, IEEE Computational Intelligence Magazine, 2008.

[25] Ledoux, J.: *The emotional brain: the mysterious underpinnings of emotional life*. New York, NY. Touchstone, 1996.

[26] Levine, D. S.: *Neural network modeling of emotion*. Physics of Life Reviews, 2007.

[27] Marreiros, G., Santos, R., Ramos, C., Neves, J., Novais, P., Machado, J., Bulas-Cruz, J.: *Ambient Intelligence in Emotion Based Ubiquitous Decision Making.* Proceeding of Artificial Intelligence Techniques for Ambient Intelligence, IJCAI'07 - Twentieth International Joint Conference on Artificial Intelligence, Hyderabad, India, 2007.

[28] Marsella, S., Gratch, J.: *Ema: A computational model of appraisal dynamics.*Agent Construction and Emotions, 2006.

[29] Minsky, M.: *The Emotion Machine.* Simon and Schuster, 2006.

[30] Nasr, E. et al.: *FLAME: Fuzzy Logic Adaptive Model of Emotions*. Autonomous Agents and Multi-Agent Systems, 3(3), 2000, pp. 219-257.

[31] Nehaniv, C.: *The First, Second, and Third Person Emotions: Grounding Adaptation in a Biological and Social World*. 5th International Conference of the Society for Adaptive Behavior (SAB'98). University of Zurich, Switzerland,August, 1998.

[32] Perlovsky, L.I.: *Toward physics of the mind: Concepts, emotions, consciousness, and symbols.* Physics of Life Reviews, 3, 2006, pp. 23–55.

[33] Petta, P., Trappl, R.: *Emotions and agents*. Multi-agents systems and applications. Springer-Verlag: New York, NY, USA, 2001, pp. 301 - 316.

[34] Picard, R. W.: *Affective computing.* Cambridge, MA: MIT Press, 2007.

[35] Picard, R.W.: *What does it mean for a computer to "have" emotions?* Emotions in Humans and Artifacts, pp 213-236. Cambridge, MA: MIT Pres

[36] Plutchick, R.: *The nature of emotions*. American Scientist, 89, 344-350.

[37] Rolls, E.T.: *The Brain and Emotion*. Oxford University Press, 1998.

[38] Russel, J. A.: *Core Affect and the Psychological Construction of Emotions*. Psychological Review, 110 (1), 145-172, 2003.

[39] Saint-Aim, S., Le-P, B., Duhaut, D.: *iGrace – emotional computational model for EmI companion robot.* Valoria Laboratory - University of Bretagne Sud France, Inria, 2010.

[40] Salmerón, J.: *Fuzzy cognitive maps for artificial emotions forecasting*. Applied Soft Computing, 2012.

[41] Scheutz, M.: *Useful roles of emotions in artificial agents: a case study from artificial life.* Proceedings of AAAI 2004. AAAI press, pp. 42-48.

[42] Sharada, G. , Ramanaiah, O. B. V.: *An Artificial Intelligence Based Neuro-Fuzzy System Emotional Intelligence.* International Journal of Computer Applications 1(13), pp. 74-79, 2010.

[43] Velásquez, J.: *Modeling emotion-based decision-making*. Proceedings of the 1998 AAAI Fall Symposium. Emotional and Intelligent: the Tangled Knot of Cognition, pp. 164–169.

[44] Wang, Y.: *On the Cognitive Processes of Human Perception with Emotions, Motivations and Attitudes*. Int. Journal of Cognitive Informatics and Natural Intelligence, 1(4), 1-13, 2007.

[45] Wehrle, T. *Motivations behind modeling emotions agents: Whose emotions does your robot have?* In Workshop W5: Grounding Emotions in Adaptive Systems (pp. 71–76). Workshop conducted at SAB'98: Fifth International Conference on Simulation of Adaptive Behavior, 1998.

[46] Yingxu Wang: *On the cognitive Processes of Human Perception with Emotions, Motivations, and Attitudes.* Int'l Journal of Cognitive Informatics and Natural Intelligence, 1(4), 2007, pp. 1-17.

[47] Zhou, J., Yu, C., Riekki, J., Karkkainen, E.: *AmE framework: a model for emotion-aware ambient intelligence.* The second international conference on affective computing and intelligent interaction, 2007.