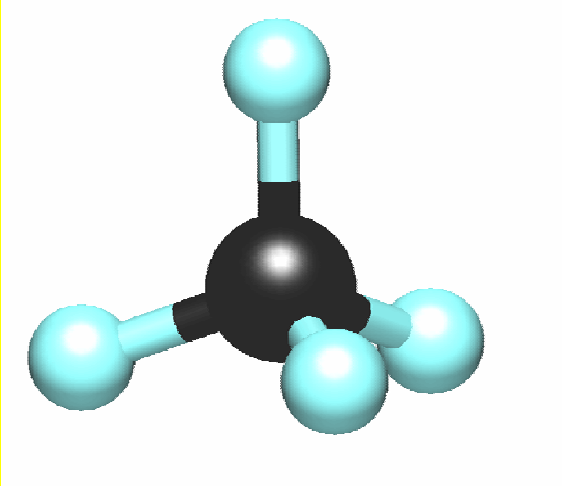
Teória grafov III

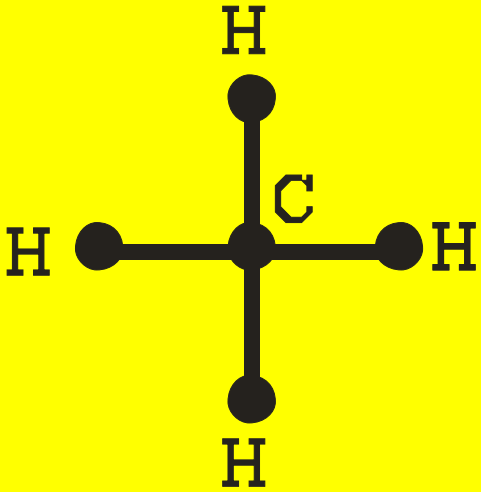
- stromy ako modely
- vlastnosti stromov
- binárne prehľadávanie
- prefixové kódy
- hry - piškvorky
- kostra

Strom – súvislý graf bez kružníc

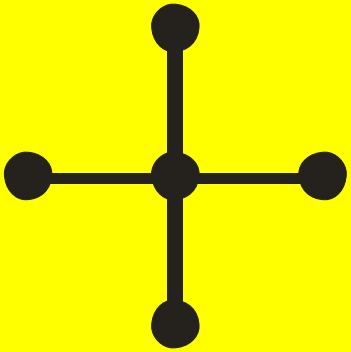
História: Anglický matematik **Arthur Caley** r. 1857 na chemické zlúčeniny – alkány



priestorový model

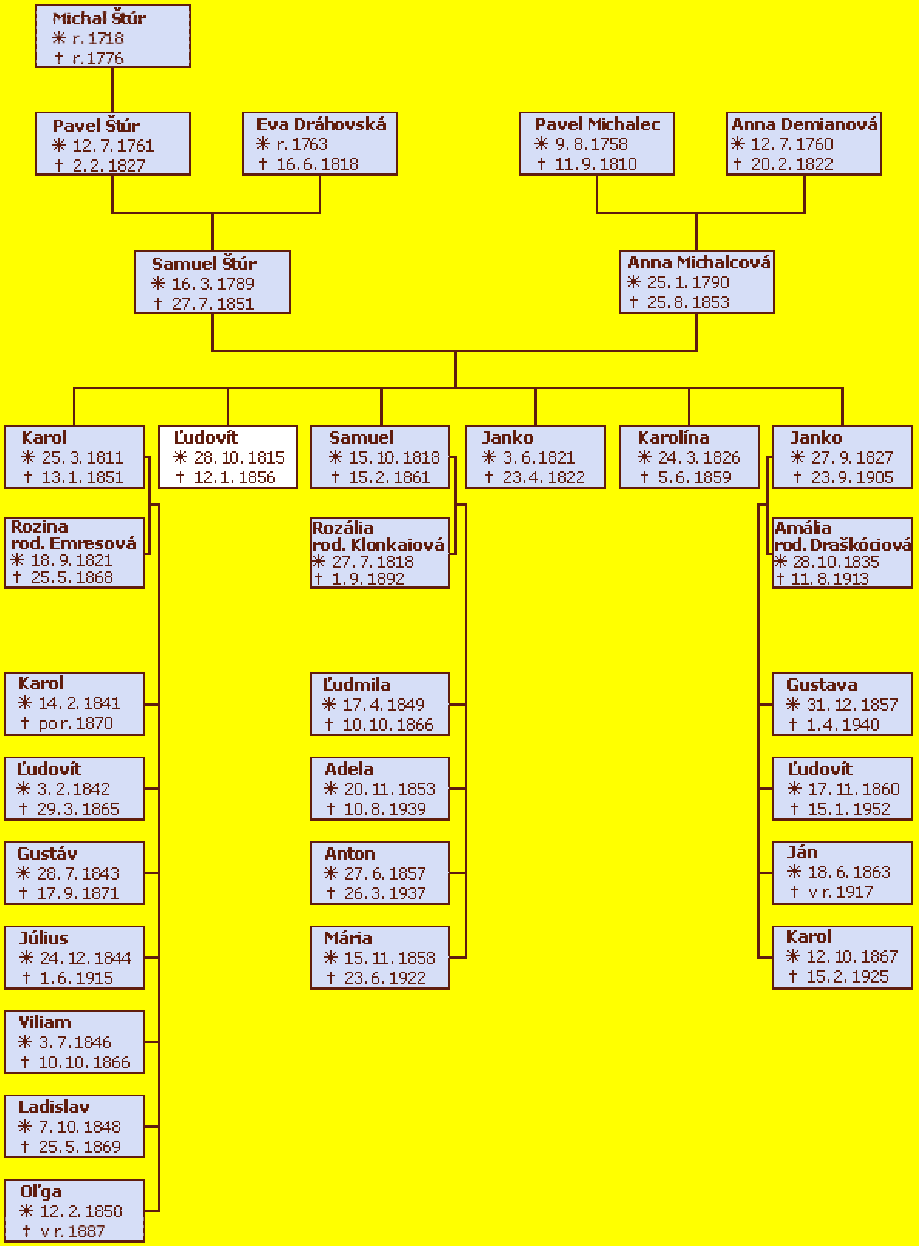


štruktúrny vzorec



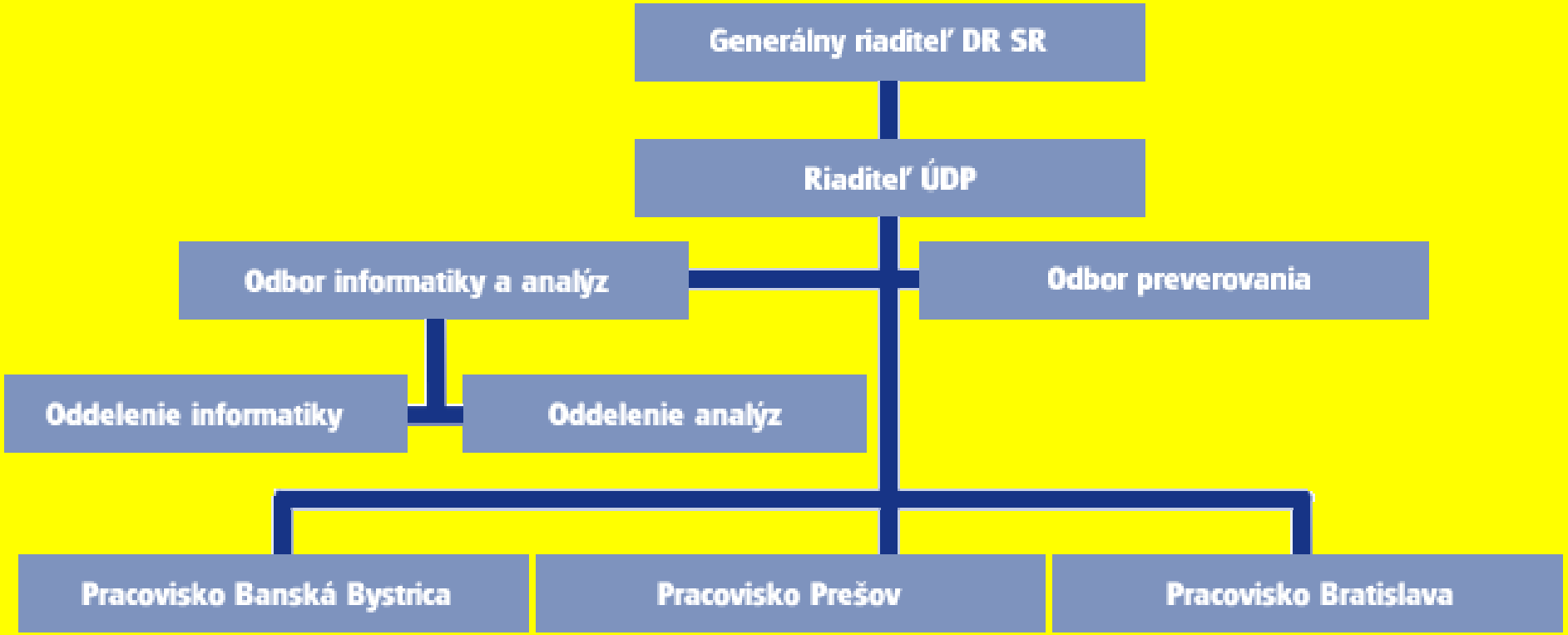
graf

Metán

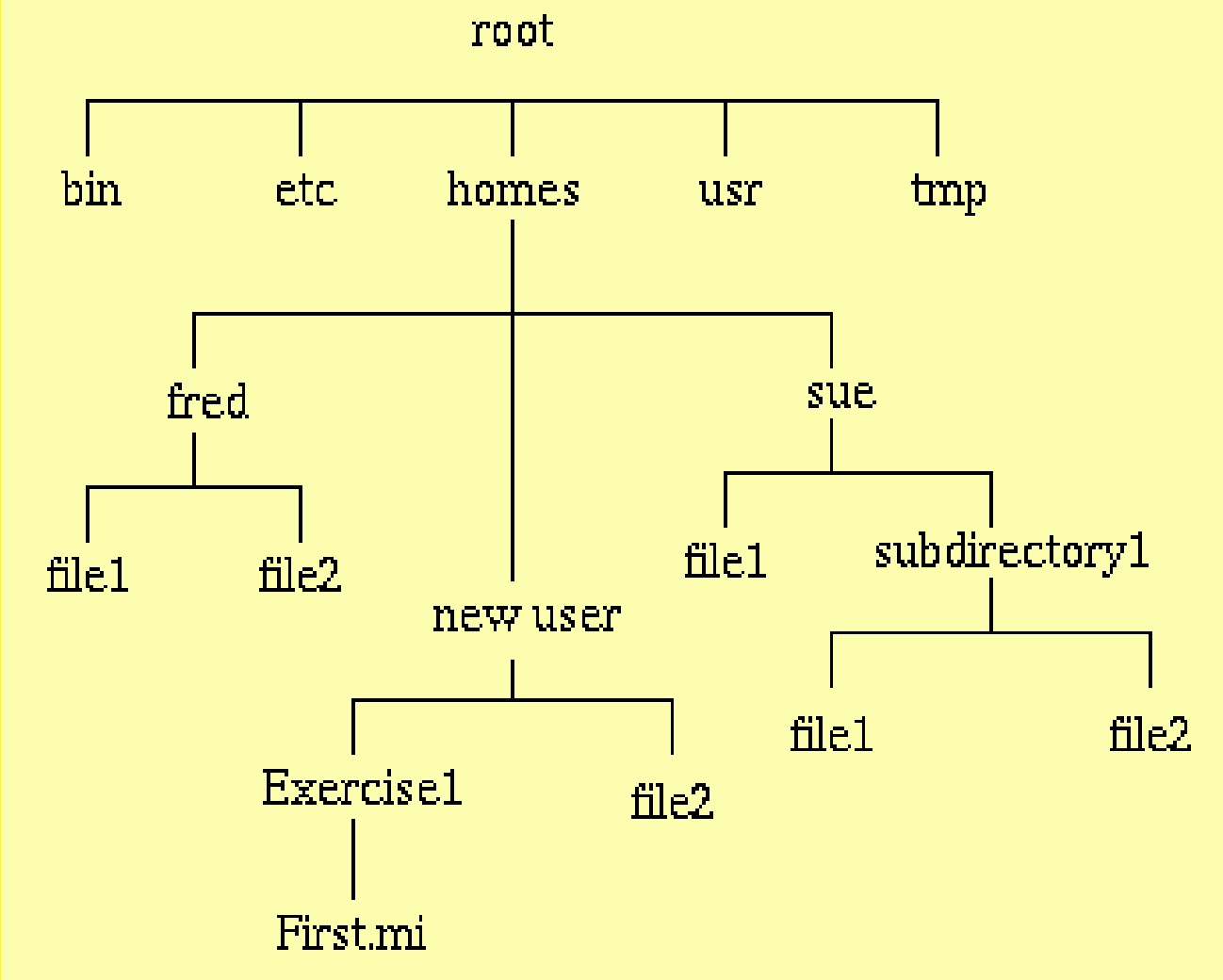


Použitie grafov – rodokmeň Ľudovíta Štúra

Organizačná štruktúra Úradu daňového preverovania



Štruktúra adresárov v počítači

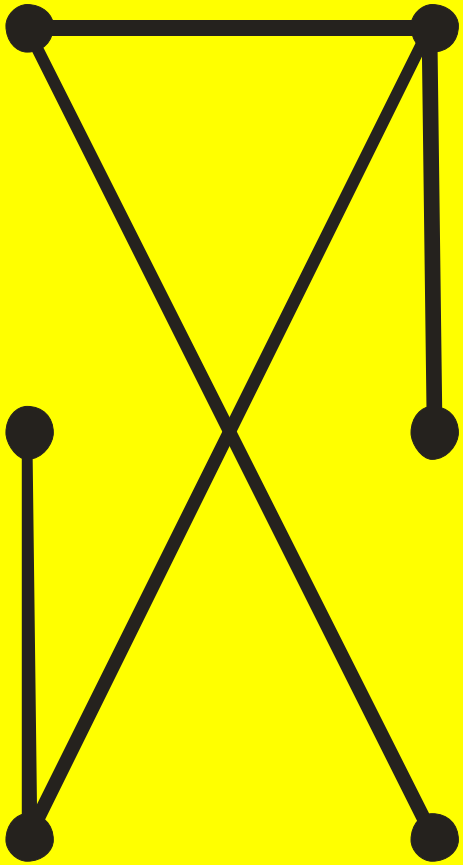


Použitie vo výpočtovej technike:

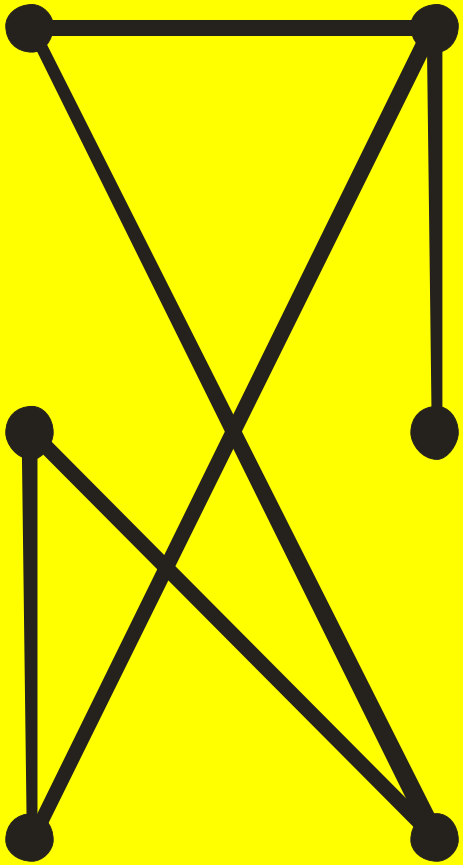
- rozmiestňovanie prvkov v databázach
- efektívne kódy na ukladanie a prenos informácií
- rozhodovacie stromy
- modely hier na určenie vyhrávajúcej stratégie
- nájdenie najlacnejšieho prepojenia uzlov komunikačnej siete

Vlastnosti stromov

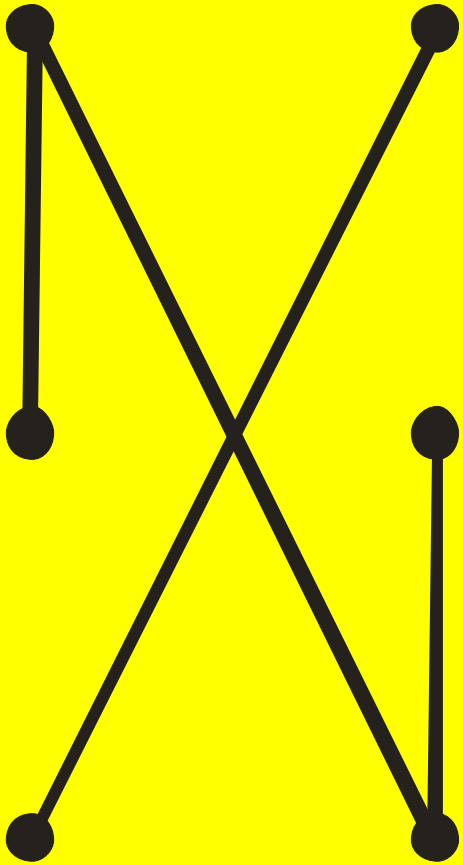
Nesúvislý graf bez cyklov (skladá sa zo stromov) voláme **les**



strom



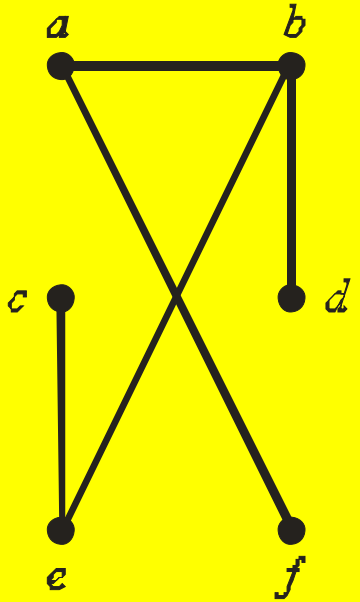
graf s cyklom



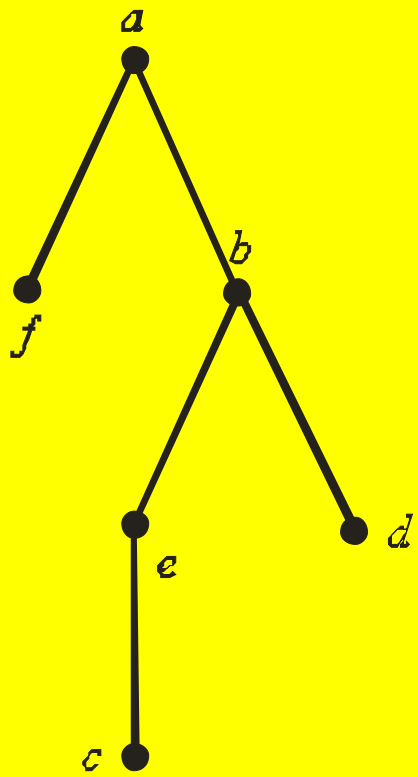
les

Veta: Neorientovaný graf je stromom iba vtedy, ak existuje jediná cesta medzi ľubovolnou dvojicou jeho vrcholov.

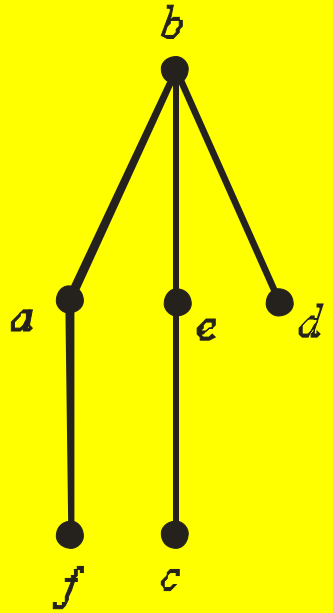
Koreňový strom (rooted tree) je strom, v ktorom bol jeden vrchol označený ako koreň. Každá hrana môže byť braná ako smerujúca od koreňa



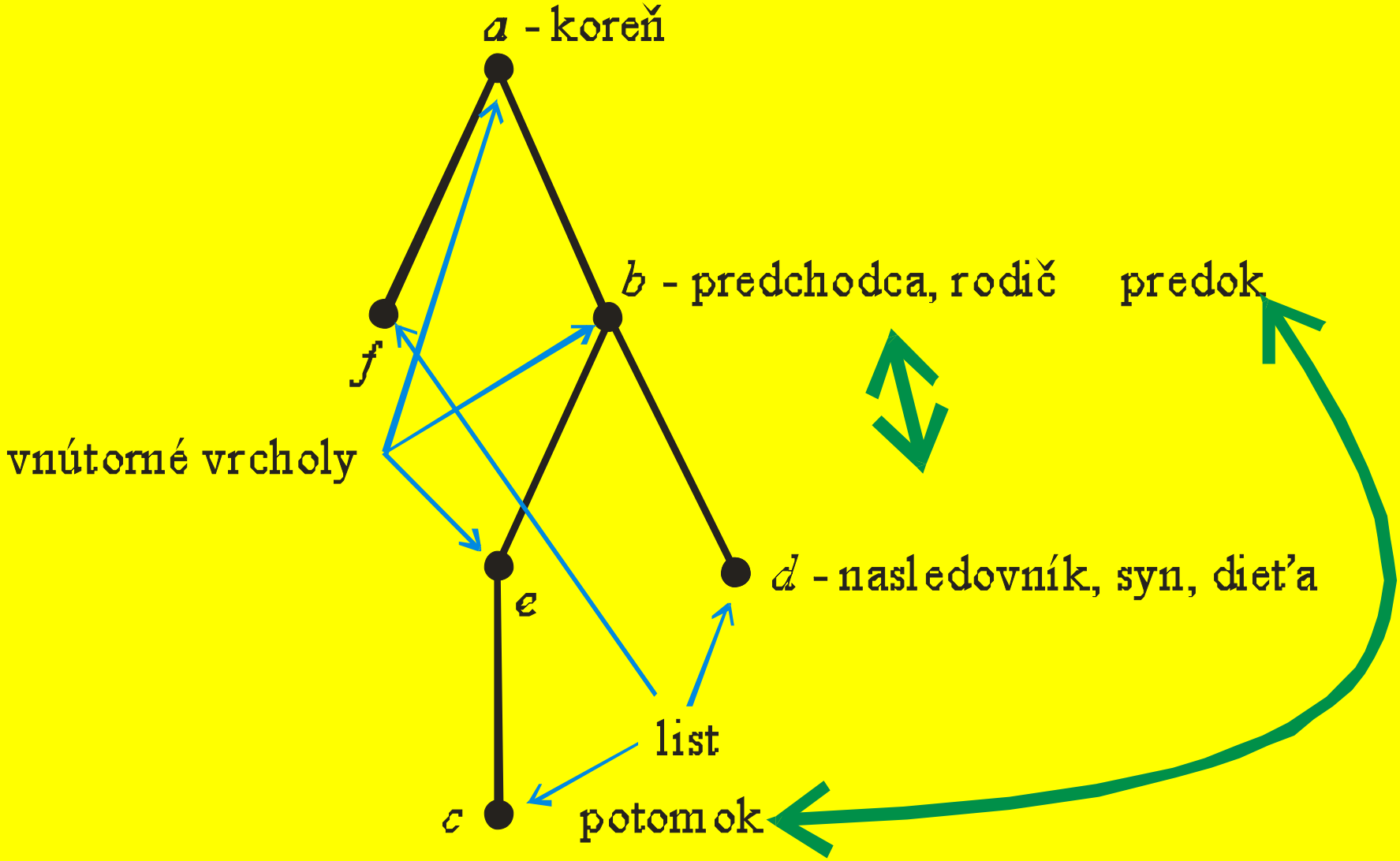
strom



s koreňom *a*



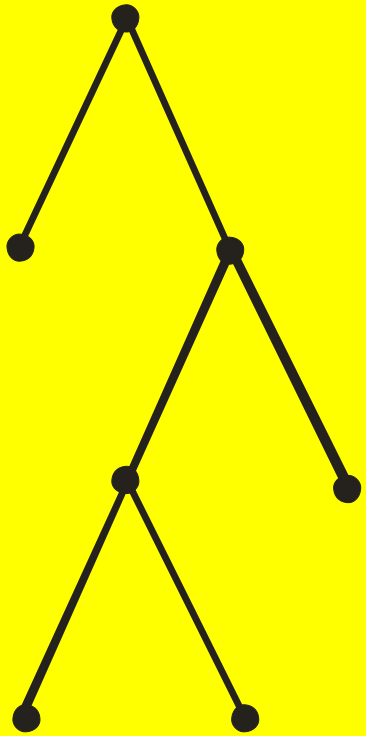
s koreňom *b*



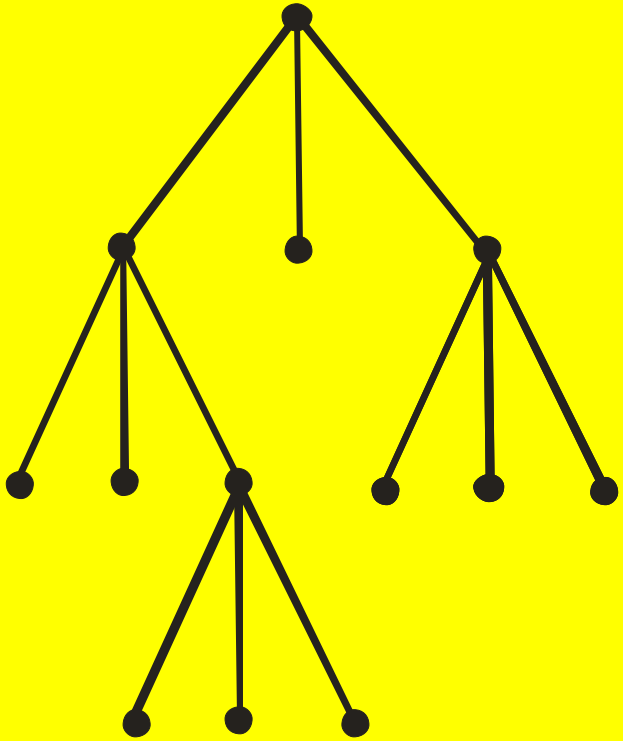
úroveň vrcholu je dĺžka cesty od vrcholu ku koreňu
hĺbka stromu = maximálna úroveň vrcholov

n-árny strom – každý vrchol maximálne *n* nasledovníkov

plne n-árny strom – každý vnútorný vrchol *n* nasledovníkov



binárny strom



ternárny strom

usporiadaný strom

ľavý nasledovník, pravý nasledovník

Veta: Strom o n vrcholoch má n-1 hrán.

Veta: Plne m-nárny strom s i vnútornými vrcholmi má $n=m\times i+1$ vrcholov.

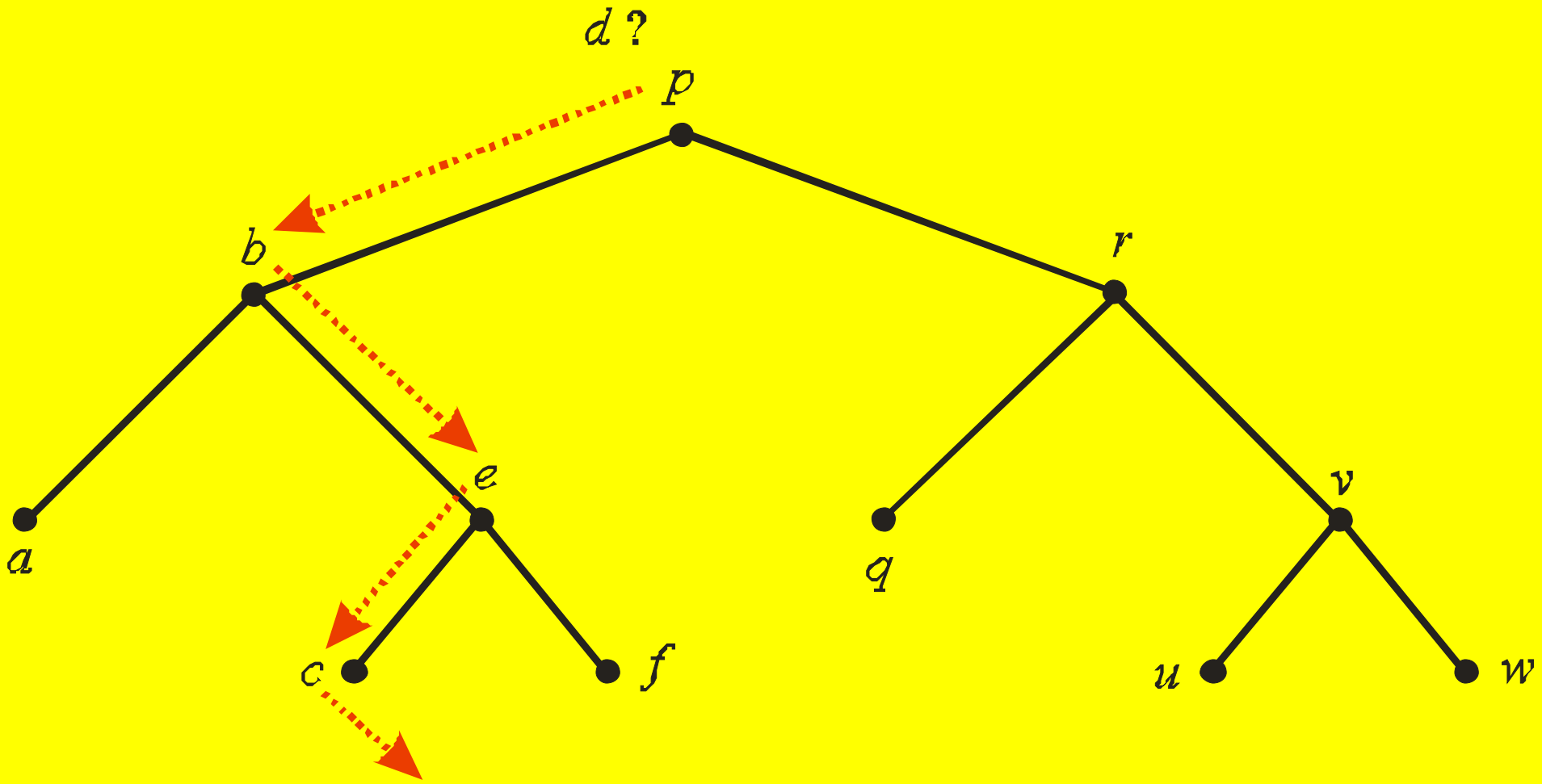
Koreňový plne m-nárny strom hĺbky h je vyvážený, keď všetky jeho listy sú na úrovni h alebo h-1.

Veta: V m-nárnom strome hĺbky h je maximálne m^h listov.

Veta: Keď m-nárny strom hĺbky h má l listov, potom $h\geq\lceil \log_m l \rceil$. Rovnosť platí pre plne m-nárny vyvážený strom.

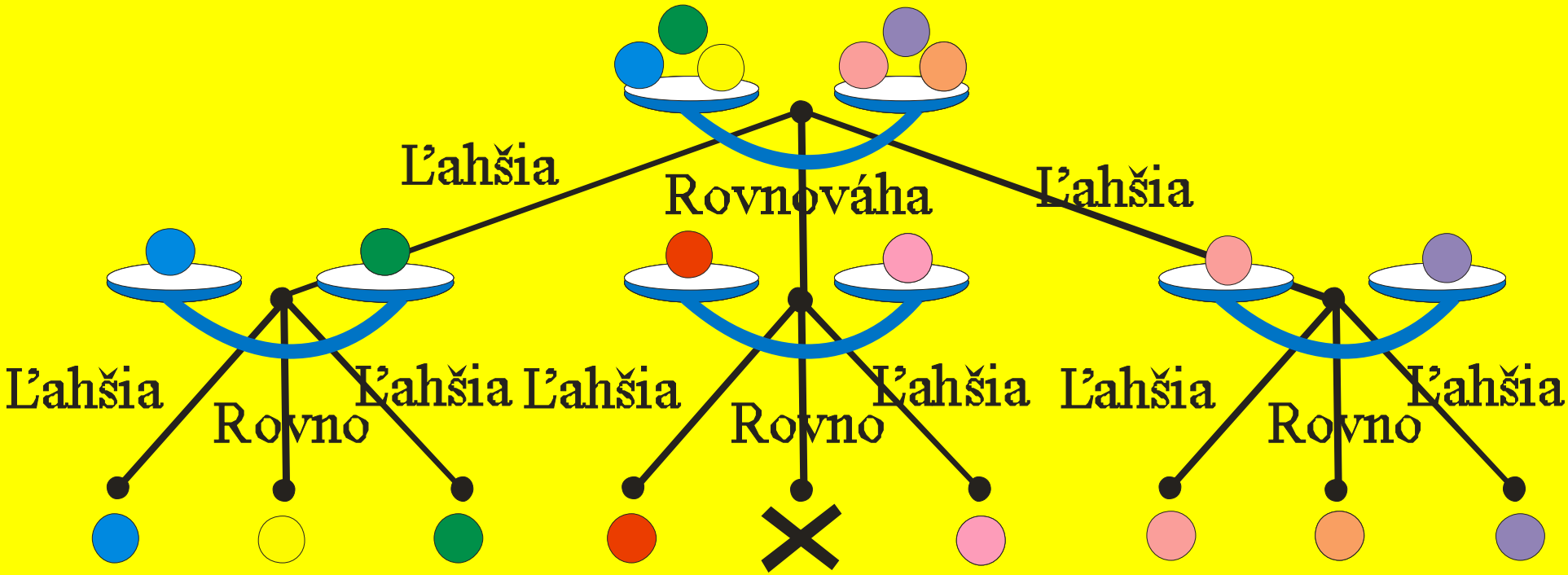
Príklad: Predpokladajme, že správu o novom víruse rozpošle každý príjemca o minútu po prijatí 10 kamarátom, a nikto nedostane správu dvakrát. Za ako dlho presiahne počet príjemcov množstvo ľudí na Slovensku?
$\lceil \mathrm{Log}_{10}5000000 \rceil=7$ minút

Binárne prehľadávacie stromy sú určené na rýchle vyhľadávanie, pre vyvážený strom nájdenie alebo pridanie prvku do databázy potrebuje maximálne $\lceil \log_2(n+1) \rceil$ porovnaní.



Rozhodovacie stromy

Máme 8 mincí, z ktorých je 1 falošná, ľahšia. Ako ju rozoznať čo najmenším počtom vážení?



Huffmanove kódovanie sa využíva aj pri stratovom pakovaní (napr. MP3) aj pri nestratovom (ZIP)

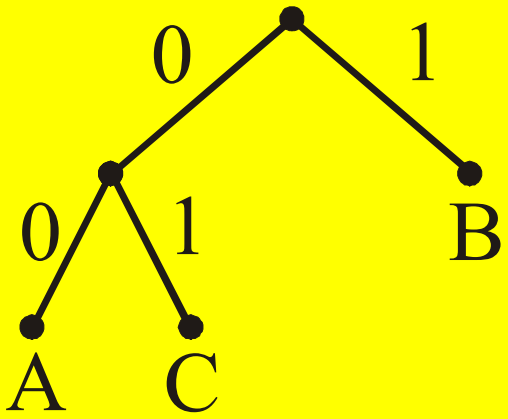
Ako zapísať viac informácii pomocou menšieho počtu znakov abecedy ? Priradiť znakom alebo vstupným informáciám, ktoré sa opakujú najčastejšie (pravdepodobnosť výskytu), čo najkratší kód.

Príklad:
Vo vstupnom súbore, ktorý chceme pakovať je napísané toto: AABBBBCC
V bytoch to vyzerá takto: 65 65 66 66 66 66 67 67
V bitoch : 10000001 10000001 1000010 1000010 atď.

Huffmanove kódovanie každému znaku priradí nejakú kratšiu postupnosť bitov. Napríklad znaku B priradí 1, znaku A priradí 01 a znaku C priradí 00.
Zakódovaná postupnosť: 01 01 1 1 1 1 00 00
Dokopy 12 bitov, čo sú takmer 2 byte. Pôvodný súbor, ktorý mal 8 bytov, sme skomprimovali na 2 byty.

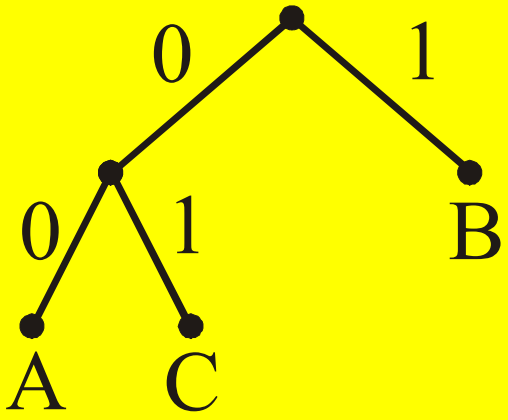
Aké značky mám priradiť jednotlivým znakom?
Treba dať pozor. Čo keby sme priradili značky zle? Napríklad znaku A by sme dali značku 0 a znaku B značku 01, a znaku C 1
Postupnosť bitov : 0001 braná zaradom, by bola preložená ako AAAC, ale my sme tak zakódovali AAB. Dá sa v tomto prípade vôbec zakódovať AAB?

Huffmanove kódovanie: vytvorí sa strom, ktorého vetvy určujú, či sa treba vybrať doľava alebo doprava. A keď narazí na koniec, tak sa pozrie aký tam je znak. Keďže v strome existuje z každého do každého vrcholu práve 1 cesta, nemôže dôjsť k omylu
Príklad stromu pre náš prípad: AABBBBCC



Úloha rozpakovať postupnosť bitov 1101001

Začneme v koreni, zoberieme prvý bit, je to 1, ideme teda doprava, je to koniec tejto vetvy ? Áno je, žiadne ďalšie z nej už nevedú. Teda napíšeme si písmeno B. Znovu začneme v koreni, zoberieme bit, je to 1, teda opäť si zapíšeme B. Znovu ideme do koreni, zoberieme ďalší bit, je to 0, ideme z koreňa doľava, je tam niečo uložené ? Nie, nie je, ďalej sa to vetví, tak zoberieme ďalší bit, aby sme videli, ako sa to vetví. Ďalší bit je 1, teda ideme doprava, tam narazíme na C. A takto postupujeme, kým máme nejaké bity na rozpakovanie.



Ako spakovať, teda urobiť Huffmanov strom?

Algoritmus na zostrojenie Huffmanovho stromu pre abecedu *A*

Krok 1. Zostroj graf $G=(V,H,p)$, kde $V=A$, a kde $p(v)$ je pravdepodobnosť znaku v. Polož H=prázdna množina. Všetky vrcholy prehlás za neoznačené.

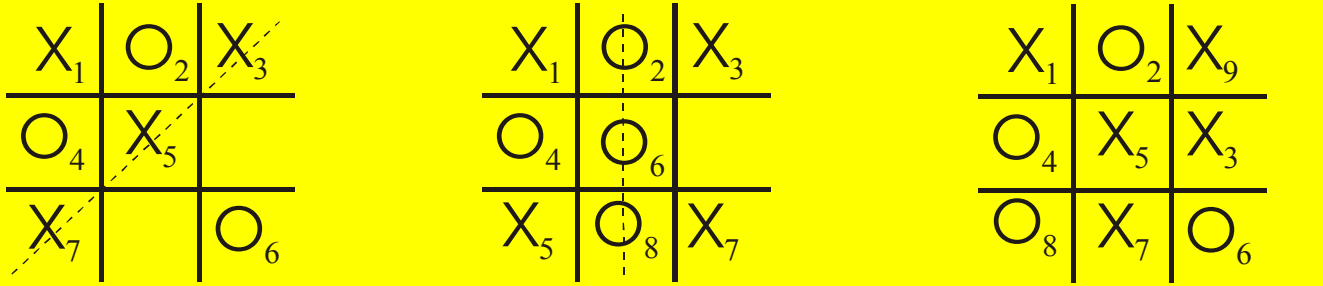
Krok 2. Nájdi 2 neoznačené vrcholy s najmenšími pravdepodobnosťami, označ ich, vytvor ďalší vrchol X a pridaj hrany spájajúce tieto vrcholy a vrchol X. Polož $p(X)$=súčtu pravdepodobností oboch vrcholov. Jednej hrane polož značku 0, druhej značku 1.

Krok 3. Ak je graf súvislý, prehlás posledne pridaný vrchol za koreň a choď na krok 4 inak choď na krok 2.

Krok 4. Graf G je teraz stromom, ktorý ma všetky vrcholy so stupňom 1 so znakmi abecedy A a cesta z koreňa do nejakého koncového vrcholu je vlastne binárnym kódom pre zakódovanie tohto znaku.

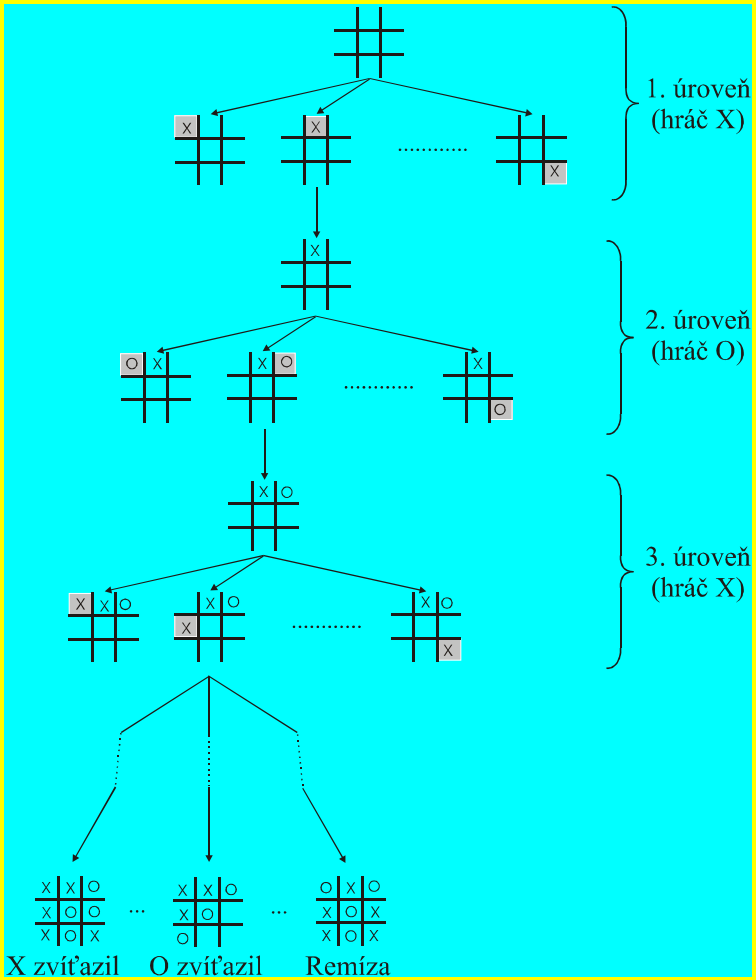
Hra piškvorky

Táto hra vyžaduje dvoch hráčov, prvý hráč je označený symbolom X a druhý hráč symbolom O. Snahou každého hráča je umiestniť na štvorcovej 3×3 hracej doske svoje symboly tak, aby tvorili buď riadok, stĺpec, alebo uhlopriečku.
Hra je zahájená hráčom X, potom nasleduje ťah hráča O. Toto striedanie hráčov sa opakuje tak dlho, až jeden z hráčov vyhral, alebo je na hracej doske umiestnených deväť symbolov.
Priebehy hier môžeme reprezentovať pomocou „stromu riešení“, kde je explicitne ukázané, ktoré ťahy boli použité.



Obrázok. Znázornenie 3 partií hry piškvorky. Indexy pri jednotlivých symboloch X a/alebo O znamenajú poradie ťahu. V prvých dvoch partiách bolo dosiahnuté víťazstvo, hráč umiestnil svoje symboly do riadku, stĺpca alebo diagonály, čo je naznačené prerušovanou čiarou. V tretej partii sa žiadnemu hráčovi takto nepodarilo umiestniť svoje symboly, po 9 ťahoch, keď sú obsadené všetky pozície hracej dosky, hra končí remízou.

Hra piškvorky patrí medzi tzv. symetrické hry, z pohľadu druhého hráča je hra identická s hrou prvého hráča (hráči sú rovnocenní, odlišujú sa len v tom, že jeden z nich zahajuje hru). Obaja hráči riešia rovnaký strategický cieľ (maximalizujú svoj zisk) a súčasne zabrániť v tejto akcii súperovi (minimalizujú súperov zisk).



Obrázok Znázornenie stromu riešení, ktorého vrchol (koreň) je prázdna hracia doska. Po prvom ťahu, ktorý hral hráč X, je obsadená jedna pozícia (zdôraznená vytieňovaným) symbolom X. V druhej úrovni, hranej hráčom O, je obsadená pozícia (vytieňovaná) symbolom O. Vetva stromu riešení končí vtedy, ak jeden hráč zvíťazil, alebo sú obsadené všetky pozície na hracej doske - hra skončila remízou.

Vo všeobecnosti existujú dva diametrálne odlišné prístupy k riešeniu problémov:

- *Prvý prístup* je založený na existencii modelu hry, ktorý obsahuje hierarchicky usporiadané pravidlá. Keď tieto pravidlá budeme dodržiavať, mali by sme dospieť ak nie víťaznej pozícii, tak aspoň k remíze. Žiaľ, tento model je zostrojiteľný len pre jednoduché hry, pre zložitejšie hry (napr. šach) už nie sme schopní ho zostrojiť s dostatočnou presnosťou a efektivitou. Obvykle sme v zložitých prípadoch schopní formulovať len rôzne všeobecné pravidlá (heuristiky), ktorých dodržiavanie v priebehu hry by malo viesť k víťazstvu.
- *Druhý prístup* je založený na rozsiahlom prehľadávaní stromu riešení, pomocou ktorého, aspoň teoreticky, sme schopní nájsť optimálny ťah v každej etape hry. Aj keď je tento prístup koncepčne veľmi jednoduchý, jeho numerická realizácia v počítači naráža na vážne problémy s enormnou veľkosťou stromu riešení. K numerickému zvládnutiu tohto prístupu musíme zaviesť niektoré podstatné zjednodušenia týkajúce sa hĺbky prehľadávania stromu riešení (tak napr. pri prehľadávaní stromu riešení ideme do maximálnej hĺbky 3 alebo 4). Toto zjednodušenie prehľadávania stromu riešení je obvykle kombinované s rôznymi heuristikami, ktoré ohodnocujú „koncové“ stavy hry.

Model hry piškvorky

Jeden zo základných princípov UI je návrh modelu študovaného problému, ktorý pomocou súboru pravidiel je schopný hrať piškvorky na dobrej úrovni. Uvedieme jednoduchý model, ktorý sa zakladá na tom, že ťah hráča je určený nasledujúcimi šiestimi pravidlami s klesajúcou prioritou:

Pravidlo 1. *Hráč vykoná ťah, ktorý vedie k jeho víťazstvu.*

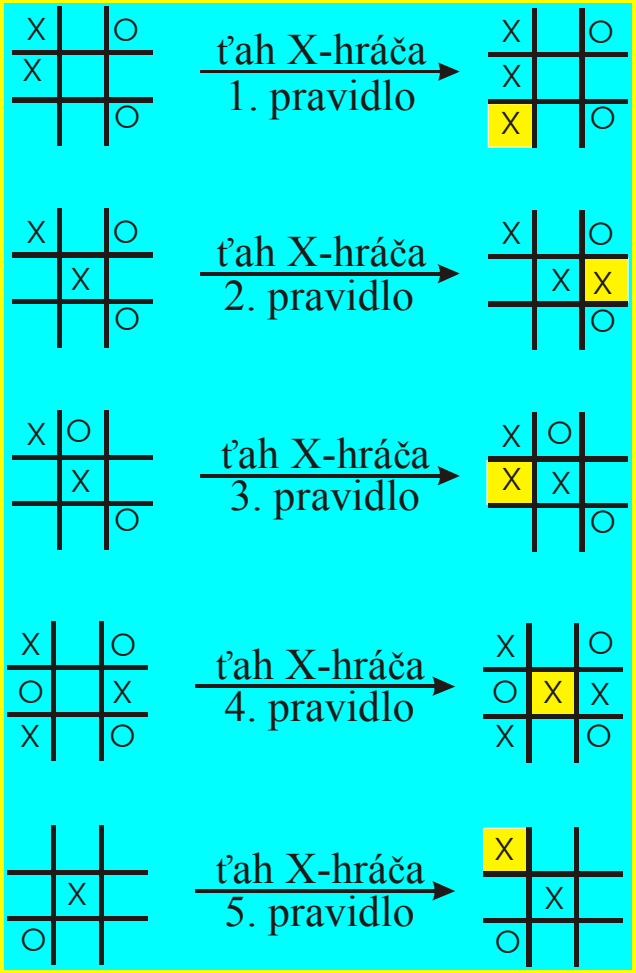
Pravidlo 2. *Hráč vykoná ťah, ktorý zabráni víťazstvu oponenta v nasledujúcom ťahu.*

Pravidlo 3. *Hráč vykoná ťah, ktorý pripraví možnosť dvojnásobného použitia 1. pravidla v nasledujúcom ťahu (tzv. vidlička).*

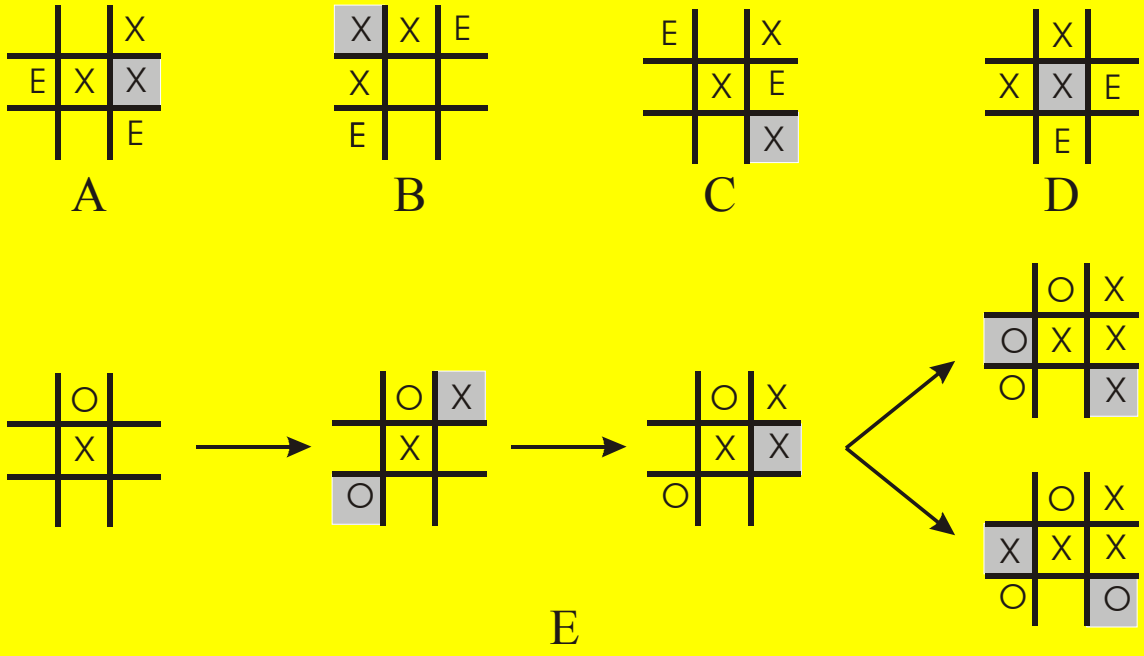
Pravidlo 4. *Hráč vykoná ťah, ktorým obsadí stredné pole.*

Pravidlo 5. *Hráč vykoná ťah, ktorým obsadí rohové pole.*

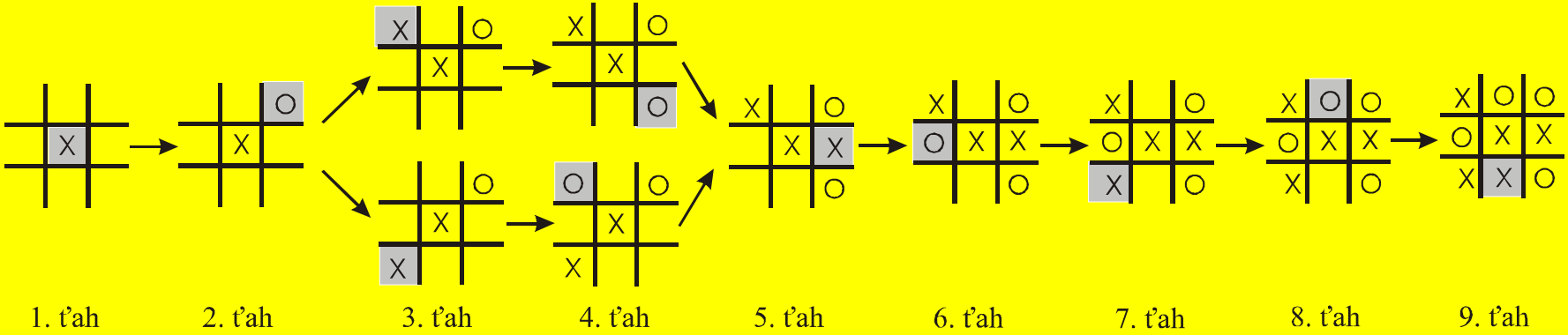
Pravidlo 6. *Hráč vykoná náhodný ťah.*



Obrázok. Diagramy ilustrujú jednotlivé pravidlá modelu hry. Poznamenajme, že v dôsledku 4. pravidla je prvým ťahom vždy obsadenie prostredného poľa.



Obrázok. Diagramy A-D znázorňujú základné typy vidličkových pozícií, ktoré sú aplikovateľné použitím pravidla 3. Symboly E znázorňujú príklady dvojice pozícií, ktoré musia byť prázdne, aby sa dala "vidlička" v ďalšom ťahu doplniť na víťaznú trojicu. Diagram E ukazuje pozíciu, ktorá je prehraná pre hráča O už po druhom ťahu. Pravidlá nášho modelu hry používa prvý hráč X.



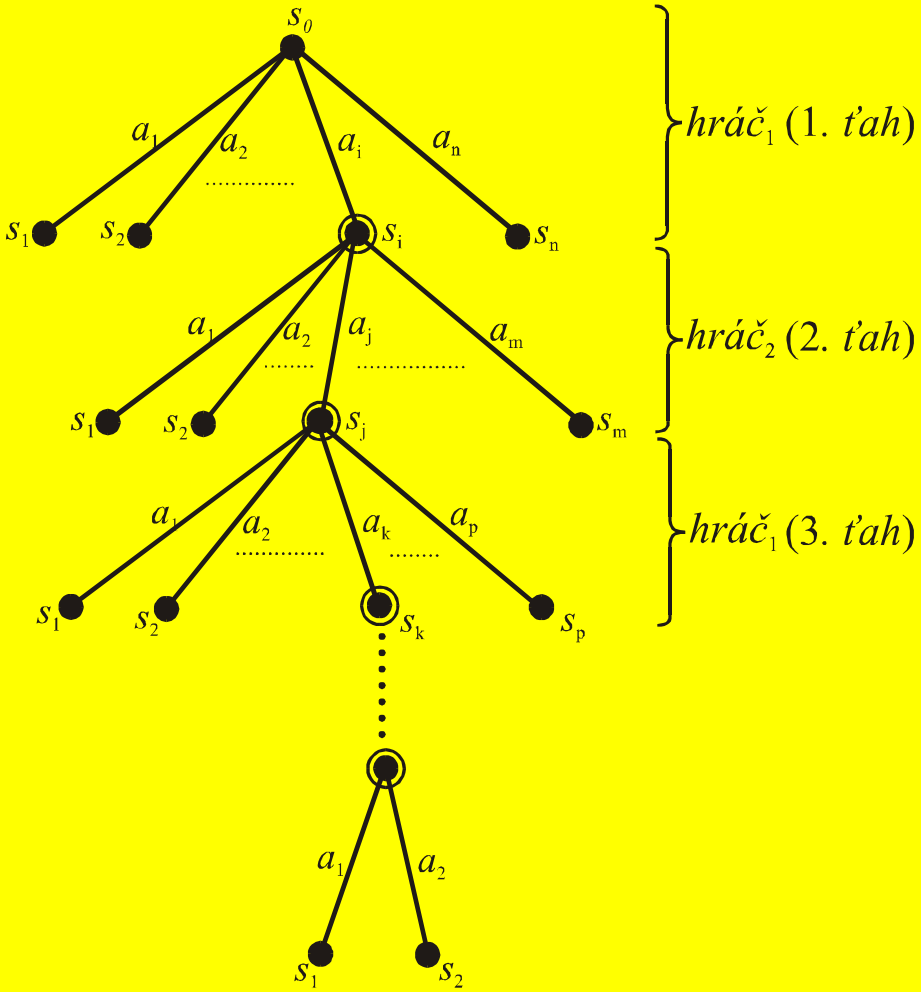
Obrázok. Znázornenie priebehu hry pomocou modelu, hra skončila remízou.

Prehľadávanie stromu riešení

Formalizácia našich úvah o priebehu hry piškvorky:

- Rozloženie znakov X a O na hracej doske sa nazýva ***stav hry***.
- Ťah hráča, t.j. pridanie znaku X alebo O na hraciu dosky do konkrétnej volnej polohy, sa nazýva ***akcia***.
- Pre daný stav s má hráč k dispozícii **množinu prípustných akcií** $\mathcal{A}(s)=\{a_1, a_2,..\}$.
- Hovoríme, že akcia a ***transformuje*** stav s na nový stav s' , alebo $s' = a(s)$.
- Množina všetkých možných stavov $\mathcal{S}=\{s_1, s_2, \ldots\}$ sa nazýva ***stavový priestor*** hry.
- Pomocou akcií sa môžeme ***pohybovať v stavovom priestore***. Nech s_0 je počiatočný stav odpovedajúci prázdnej hracej doske, prvý hráč na vybranú pozíciu priloží znak X, dostaneme pozíciu s_1, potom druhý hráč na stav vykoná akciu (priloží na voľnú polohu znak O), takto dostaneme stav s_2, atď. Priebeh hry je popísaný sekvenciou stavov

$$s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \ldots\ldots \xrightarrow{a_k} s_k \Rightarrow (s_0 s_1 s_2 \ldots s_k)$$

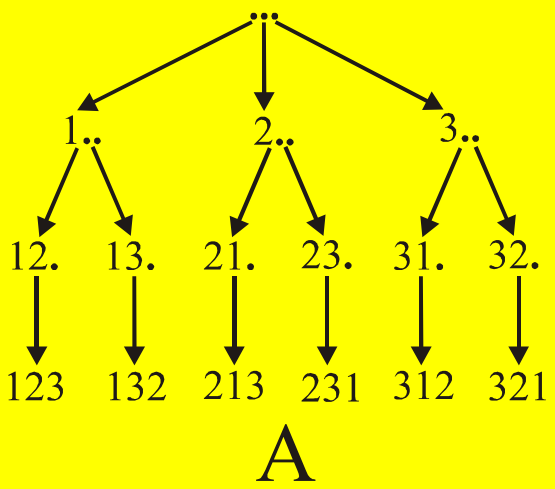


Obrázok. Prvý hráč vytvorí z počiatočnej pozície (vrchol stromu) všetky možné nasledujúce (1-ťahové) pozície s_1, s_2,…. Na základe určitých (racionálnych) úvah vyberie pozíciu s_i. Druhý hráč z pozície s_i vytvorí nové (2-ťahové) pozície, z nich vyberie pozíciu s_j ako výsledok svojho ťahu – akcie. Oba hráči tento postup opakujú, končí sa vtedy, ak niektorý hráč vyhral, alebo obaja hráči remizovali.

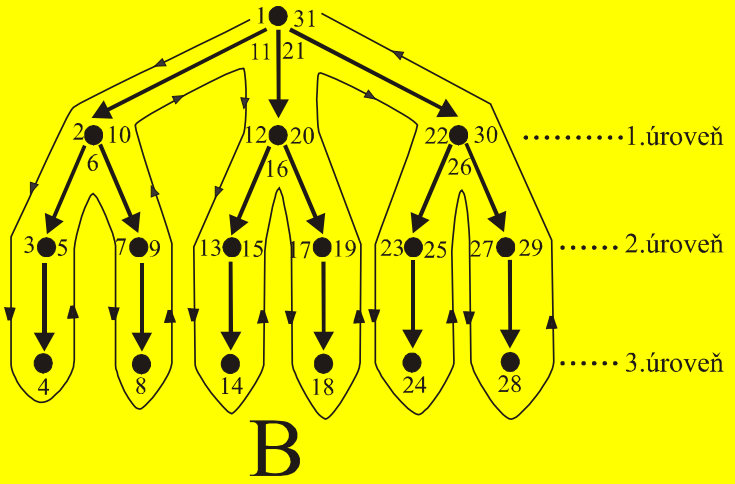
Spätné prehľadávanie

Metóda je spätného prehľadávania je ilustrované konštrukciou všetkých možných 6 permutácií troch objektov 1, 2 a 3:

- Metóda je inicializovaná prázdnym riešením (**...**), na prvej úrovni je prázdny symbol ‘**.**’ nahradený postupne všetkými možnými číslicami 1, 2 a 3.
- V druhej úrovni je druhý prázdny symbol ‘**.**’ nahradený prípustnými číslicami 2, 3 (alebo 1, 3 resp. 1, 2).
- Tento jednoduchý postup “predlžovania” riešení sa opakuje až do vyčerpania prípustných symbolov.
- Potom sa metóda musí vrátiť späť o jednu úroveň vyššie a celý postup sa opakuje.



A



B

Metóda spätného prehľadávania pre piškvorky

Veľkosť stavového priestoru (t.j. počtom rôznych stavov - pozícií hry) je určená vzťahom

$$N = \sum_{p=1}^{5} \binom{9}{p} \left[\binom{9-p}{p-1} + \binom{9-p}{p} \right] - 2 \times 6 \times 13 = 5889$$

Použitím metódy spätného prehľadávania môžeme zostrojiť kompletný strom riešení hry piškvorky. Zistili sme, že má nasledujúci počet koncových pozícií, ktoré sú charakterizované takto

Počet	Typ
131184	víťazstvo hráča X
77904	víťazstvo hráča O
46080	remíza hráčov X a O
255168	celkový počet

Ďalší dôležitý problém pred ktorým teraz stojíme (predpokladáme, že už poznáme úplný strom riešení hry piškvorky) je zistiť víťaznú stratégiu pre prvého hráča. Existujú tieto tri možnosti:

(1) ***existuje víťazná stratégia pre prvého hráča***, to znamená, že druhý hráč pri použití tejto stratégii musí prehrať,

(2) ***existuje víťazná stratégia pre druhého hráča***, to znamená, že prvý hráč pri použití tejto stratégii musí prehrať,

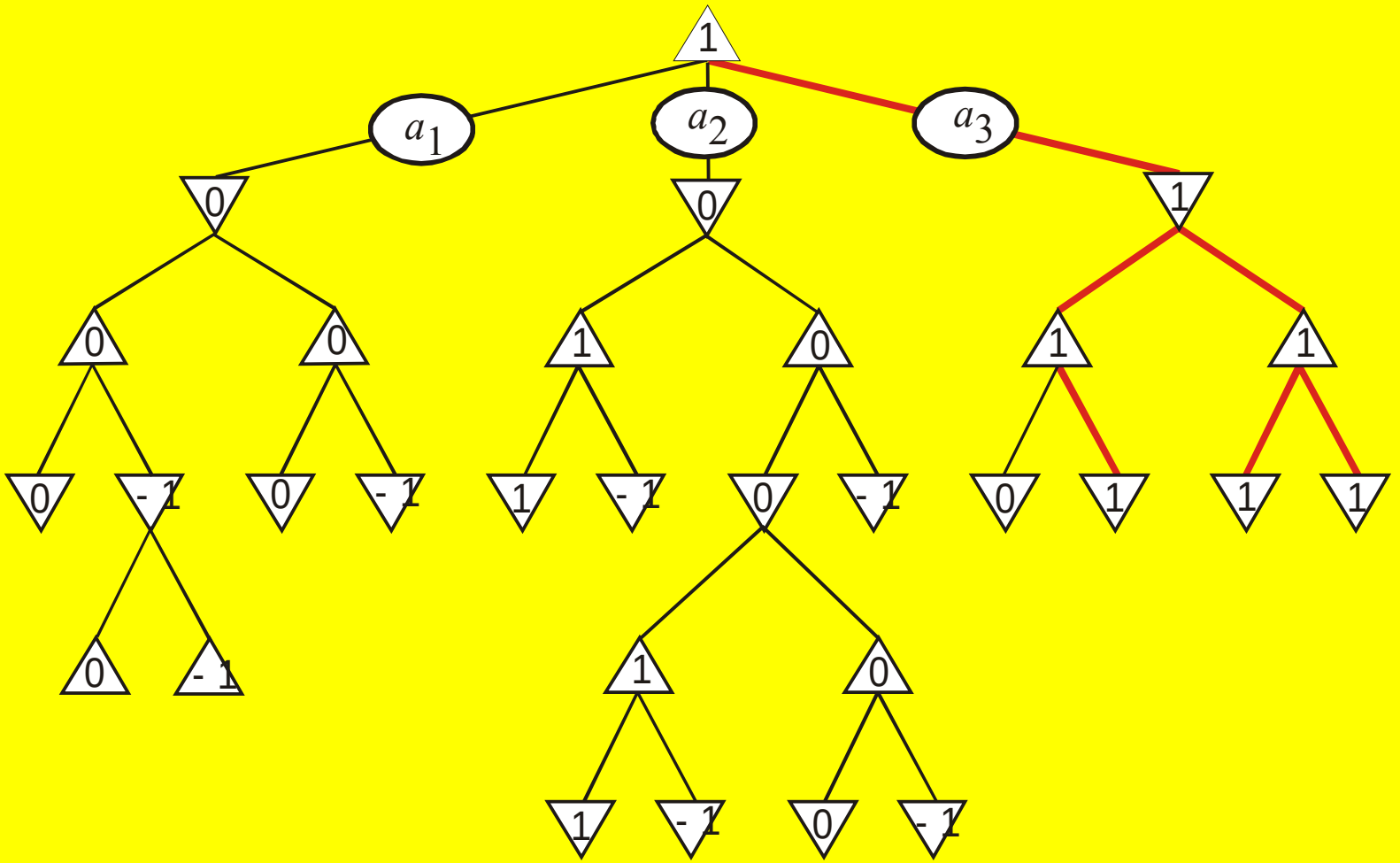
(3) ***neexistujú víťazné stratégie***, optimálna stratégia poskytuje obom hráčom len remízu.

Jednoduchá modifikácia metódy spätného hľadania pre konštrukciu stromu riešení nám umožňuje zistiť optimálnu stratégiu pre oboch hráčov. Základný princíp je veľmi jednoduchý, vychádza zo skutočnosti, že každý hráč volí taký ťah, aby maximalizoval svoj zisk a minimalizoval zisk súpera, preto ho nazývame „*minimax princíp*“.

Prvý hráča (označený max) chce vyhrať, to znamená, že bude vyberať také ťahy, aby maximalizoval svoj zisk a minimalizoval zisk súpera (druhého hráča, označeného min).

Druhý hráč sa bude správať podobne, ako prvý hráč, chce maximalizovať svoju zisk, čo je ekvivalentné že minimalizuje zisk prvého hráča (preto je druhý hráč označený min).

Strom riešení a minimax princíp



1. hráč (max)

2. hráč (min)

1. hráč (max)

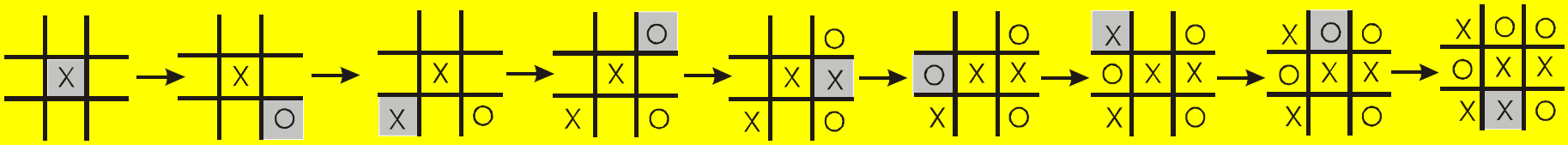
2. hráč (min)

1. hráč (max)

2. hráč (min)

Pomocou tohto jednoduchého postupu založeného na minimaxovom princípe môžeme riešiť veľkú triedu hier s dvoma hráčmi, ktorí striedavo vykonávajú ťahy, pričom hlavným strategickým zámerom každého hráča je maximalizovať svoj zisk (t.j. minimalizovať zisk súpera). Hlavný problém s použitím tohto algoritmu spočíva v tom, že strom riešení pre zložitejšie hry (napr. šach) má enormnú veľkosť, takže systematické prechádzanie po všetkých jeho vrcholoch-stavoch je nerealizovateľné.

Ak aplikujeme metódu spätného hľadania kombinovanú s minimax princípom, potom zistíme, že pri obojstranne korektnej hre hráči môžu v najlepšom prípade dosiahnuť len remízu. To znamená, že vrchol stromu riešení je ohodnotený na záver 0, jedna z týchto optimálnych partií je postupnosť ťahov reprezentovaná „permutáciou“ (597364128).



Obrázok. Znázornenie postupnosti (597364128) pomocou jednotlivých ťahov hry piškvorky. Jednoducho sa môže skontrolovať, že táto postupnosť ťahov vyhovuje modelu hry piškvorky, ktorý bol diskutovaný v prvej časti tejto kapitoly.

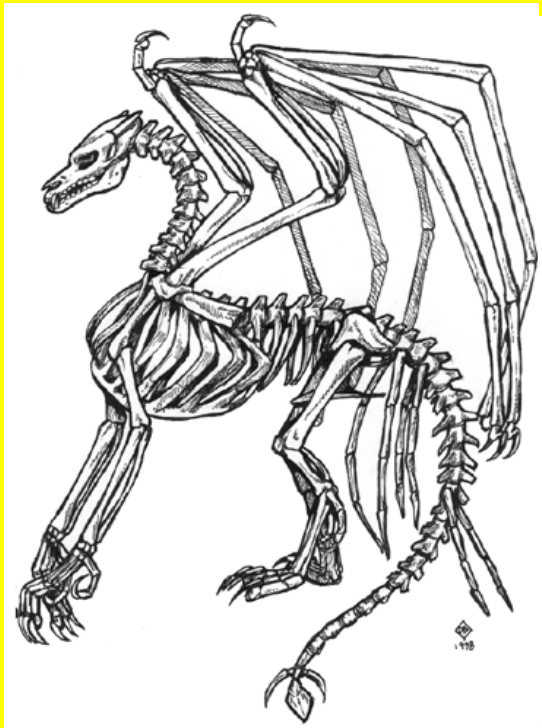
Veta A. Optimálna stratégia hry piškvorky vedie k remíze, neexistuje taká stratégia, pomocou ktorej by jeden hráč vyhral a druhý prehral.

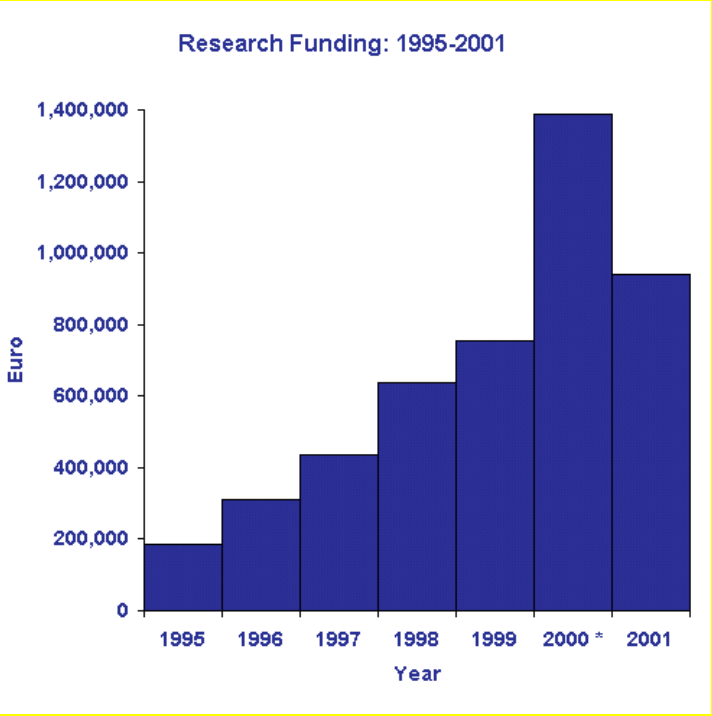
Tento dôležitý záver vyplýva z výsledkov získaných metódou spätného hľadania spolu s „minimax“ princípom. Tým, že metóda prekontrolovala celý strom riešení, môžeme tento výsledok pokladať za konečný a nemenný.

Veta B. Optimálna stratégia hry piškvorky sa pre prvého hráča riadi pomocou modelu hry s hierarchickými pravidlami.

Podobne, ako v predchádzajúcom prípade, aj táto veta je dôsledkom počítačových simulačných výpočtov. Skontrolovali sme optimálne postupnosti ťahov dĺžky 9, zistili sme, že vo všetkých prípadoch model hry je splnený.

Kostra grafu je strom

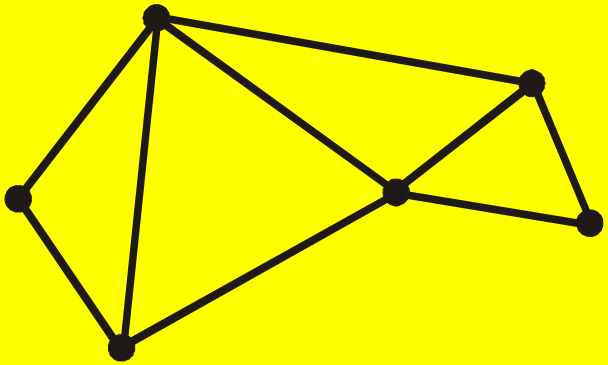




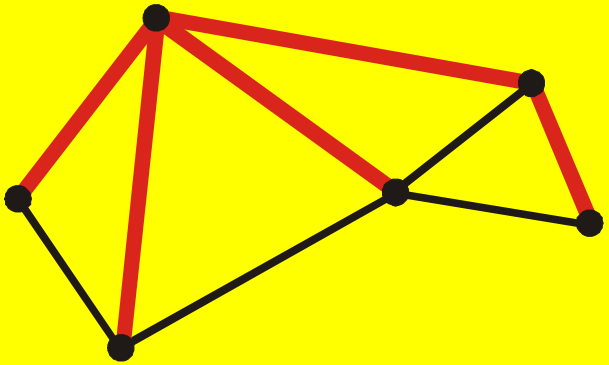
=

Kostra (spanning tree) obyčajného grafu *G* je podgraf grafu *G*, ktorý je stromom obsahujúcim všetky vrcholy z *G*.

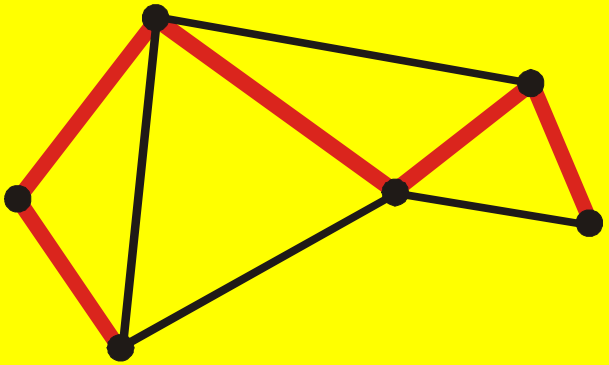
Predstavte si snehovú kalamitu, kedy chceme sprevádzkovať silnice prejdením snehovým pluhom tak, aby boli prepojené všetky mestá, ale aby sme pritom museli čistiť čo najmenej kilometrov. Ako to urobiť? (Ekvivalentná úloha: najlacnejšie prepojenie počítačovej siete)



graf siete

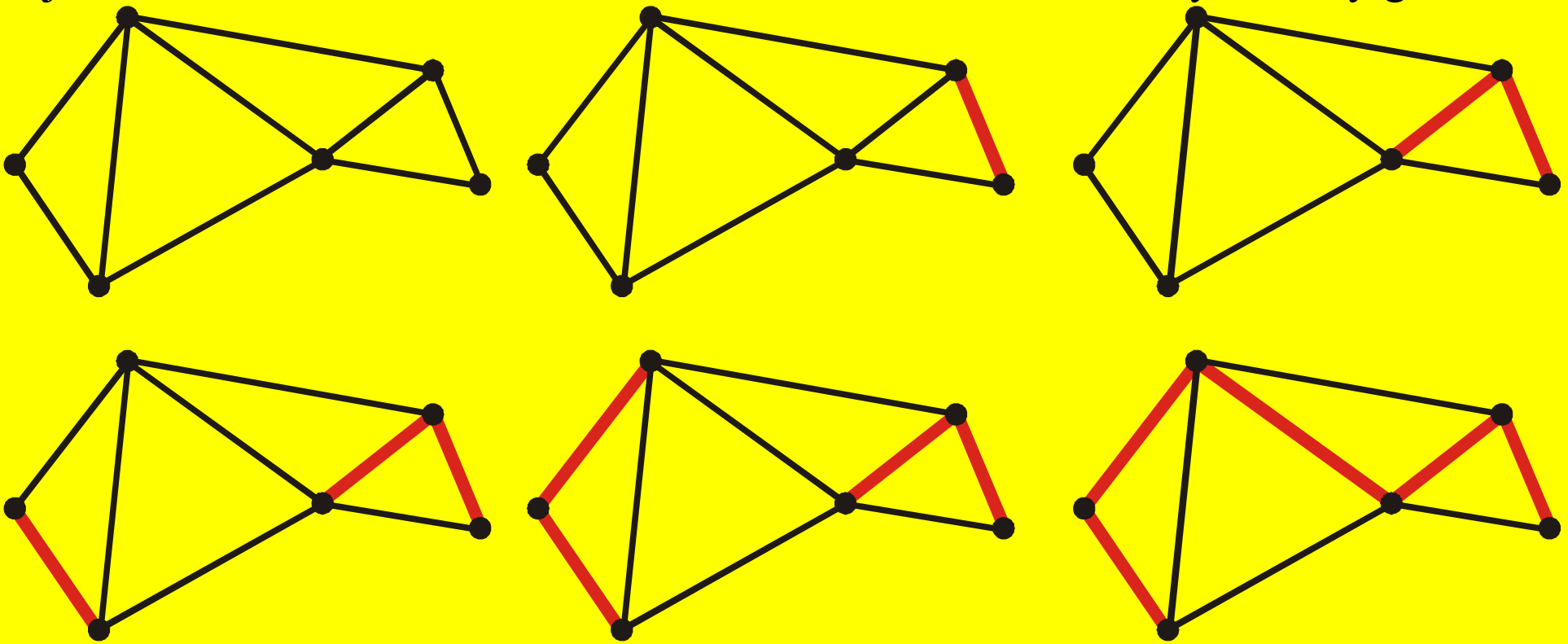


kostra



minimálna kostra

Ako vyrobiť kostru? Vyberáme do kostry najmenej ohodnotené hrany, tak aby v nej nevznikla kružnica, dokiaľ nebude kostra obsahovať všetky vrcholy grafu.



Je to greedy (pažravý) algoritmus, ale v tomto prípade funguje optimálne, nájde (nejakú) minimálnu kostru a časová náročnosť je $O(|E| \log |E|)$