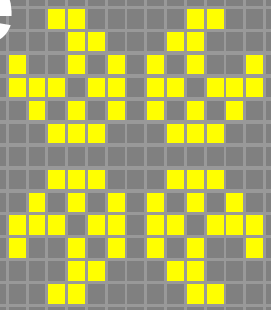
Hillclimbing & Game of life

Vypracovala: Martina Práznovská

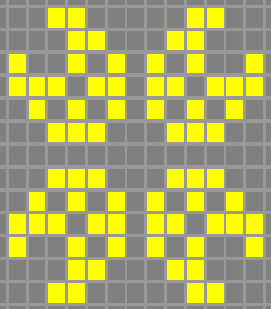


Zadanie:

Pomocou hillclimbingu nájdite taký najmenší (teda používajúci najmenší počet živých buniek) netriviálny obrazec, ktorý pri hre Life kompletne zanikne už po prvej iterácii.

Netriviálnosť spočíva v tom, že aspoň jedna bunka musí vymrieť na základe “prehriatia“, teda preto, lebo má 4 alebo viac susedov.

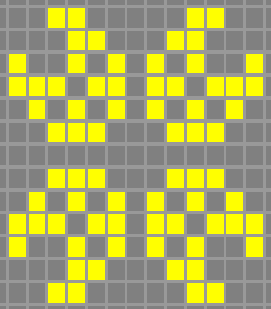
Game of Life – Hra Život



Pravidlá

2. **STABILNÝ STAV: Keď má daná bunka dvoch živých susedov (nech už je sama živá či „mŕtva”), ostáva v rovnakom stave ako predtým aj v ďalšej generácii.**
2. **RAST: Keď má bunka presne troch živých susedov, bude v ďalšej generácii živá bez ohľadu na jej momentálny stav.**
3. **SMRŤ: Keď má bunka počet susedov 0, 1, 4 - 8, bude v nasledujúcej generácii „mŕtva”. Bunka teda „zomrie na podchladenie alebo na prehriatie, keď má primálo alebo priveľa susedov”.**

Reprezentácia obrazca

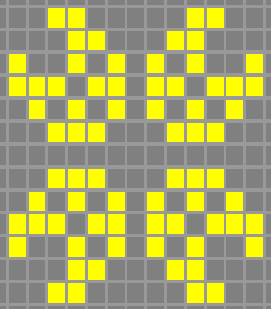


•Obrazec je prezentovaný ako štvorcová matica jednotiek a núl

0 0 0 1 0
0 0 1 1 1
1 1 1 0 1
0 0 1 0 0
1 1 0 1 1

• stav 1 – živá bunka
• stav 0 – mŕtva bunka
• iterácie sú vypočítavané na základe pomocnej matice „susedstva“

Iterácia



Matica

1	0	1	0
1	1	1	0
0	1	1	1
1	0	1	0

Matica „susedstva“

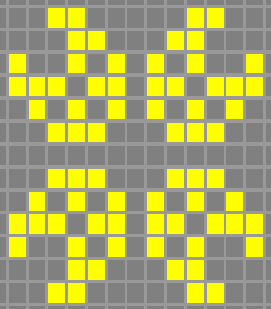
2	5	2	2
3	6	5	4
4	6	5	3
1	4	3	3

Krok iterácie

Nová matica sa vytvorí nahradením polí v pôvodnej matici:

- V poli s výskytom susedov 0,1,4 a viac -> stavom 0
- Stav 1 vznikne v poli s výskytom 3 susedov
- Pri susedstve „2“ sa stav poľa nezmení

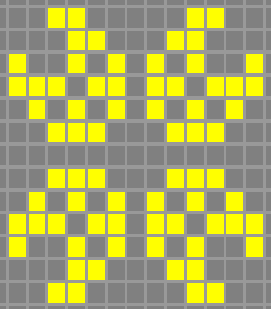
Pseudokód



```
for početExperimentov
{
    A=generujmaticu(rozmer);náhodne vygenerovaná štvorcová matica s nulami a jednotkami;
    najfitness = 105 cas = 0;

    while cas < MaxCas
    {
        aktFitnes = vypoFitness(A) vypočíta fitness matice
        [najMatica,fitnes]=prehladajOkolie(A)          funkcia prehľadá okolie a vráti najlepšieho jedinca a jeho fitness
        if( fitnes najlepsieho v okoli  >  aktFitnes )
        {
            A = najMatica;
        //najlepšie riešenie nájdené v okolí, použije sa ako stred nového okolia
        }
        if(fitnes == 105 ) { break }
    }
}
```

prehladajOkolie(A)

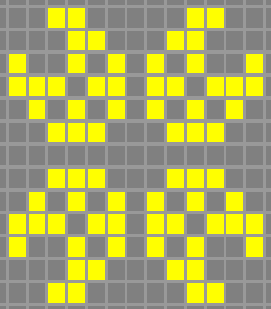


```
fitness=vypoFit(matica);

for k=1:velkostOkolia
    zmutovana=mutuj(matica,pmut);          %zmutuje vstupnu maticu
    fitZmutovanej=vypoFit(zmutovana);      %vypocita fitness zmutovanej
        if fitZmutovanej > fitness
            matica=zmutovana;   %zapamata si zatial najlepsie riesenie
            fitness= fitZmutovanej;
        end
end
```

- V matici zmeníme náhodne jednotlivé bity na základe pravdepodobnosti mutácie (napr. 0.05)
- Vypočítame fitness novovytvorenej matice a porovnáme ju z fitness pôvodnej matice
- Po prehľadaní okolia, vyberieme maticu s najvyššou fitness, ktorú potom berieme ako stred nového okolia

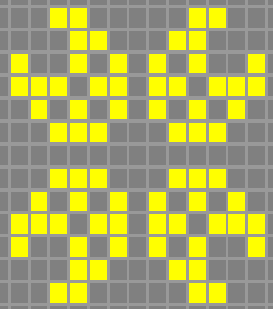
vypoFit(A)



- Fitness sa skladá z viacerých častí:
- Počet živých na začiatku
- Počet krokov, kým jedinec vyhynie
- Počet buniek, ktoré zomreli na prehriatie

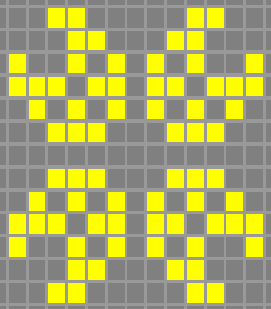
- Fitness=100/krokUplnehoVyhynutia – živýNaZačiatku + MrtvolyNaPrehriatie + konštanta
- Konštanta vyjadruje primeraný počet živých na začiatku (napr. 9)

Vzniknuté problémy



- Spôsob výpočtu fitness
- Veľkosť prehľadávaného okolia
- Náhodná zmena bitových premenných – malá zmena v matici, spôsobí veľkú zmenu vo fitness
- Nevýhoda horolezeckého algoritmu – po určitom počte iteračných krokov sa vracia k lokálnemu optimálnemu riešeniu

Lokálne optimá

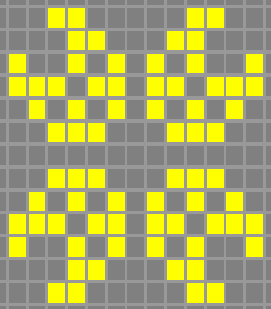


Časté uviaznunie v lokálnom optime
= jedinec zomrie po druhej iterácii

0 0 0 0 0
0 1 1 0 0
0 0 1 1 1
0 0 1 0 0
1 0 0 0 0

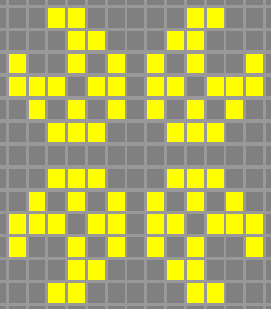
Fitness 57

Najlepšie riešenie



0 0 1 0 0
0 0 1 0 0
1 1 1 1 1
0 0 1 0 0
0 0 1 0 0

- Fitness = 96



- Ďakujem za pozornosť