Case study z Evolučných algoritmov
1999/2000

Standing ovations

Peter Kolenič
MFF UK

Hrubá špecifikácia problému:
Modelovanie situácie po skončení prednášky, keď niektorých poslucháčov príjme vnútorné rozpoloženie k postaveniu sa, a tí príjmu k tomuto činu iných. Modelovanie vzájomného ovplyvňovania sa jedincov v spoločnosti.

Model (predbežné info):
Jedinec tlieska istou silou, ktorá je závislá od jeho presvedčenia o kvalite prednášky, zároveň vie, že je slušné istý čas po skončení prednášky tlieskať (toto donútenie okolnosťami budeme v ďalšom nazývať bias (podobne ako "budenie" neurónu)). Tým zvyšuje hluk v miestnosti, a ovplyvňuje tak presvedčenie všetkých v nej (podľa modelu "Iný si myslia, že bola dobrá, tak asi bola ..."). Naviac priamo ovplyvňuje svojich susedov. Ak tlieska príliš silno, postaví sa. To priamo ovplyvňuje jeho susedov po druhý raz: ak stojí istý počet ľudí okolo nejakého jedinca, postaví sa aj on.

Jedinec je vlastne celulárny automat (keďže ale súčasťou charakterizácie jeho stavu je aj presvedčenie o kvalite prednášky (v našom reálne číslo), počet jeho stavov je veľký a ak by sme opomenuli len konečný počet reálnych čísel zapísateľných v počítači, tak je nekonečný, preto to "vlastne"), ktorý je ale ovplyvňovaný nielen bezprostrednými susedmi, ale aj globálnou situáciou (preto to "vlastne" po druhý krát).
Globálne ovplyvňovanie je priemerný potlesk + bias.

Takto zjednodušene možno modelovať chovanie sa ľudí v rôznych situáciách, napríklad možno interpretovať jedinca ako potencionálneho zákazníka, hlasitosť potlesku ako jeho presvedčenie, že práve náš výrobok je kvalitný, a jeho postavenie sa ako rozhodnutie náš výrobok používať. Na funkciu bias-u sa potom možno pozerať ako na náklady na reklamu v danom čase. Celkove sa potom môžeme snažiť nájsť funkciu maximalizujúcu zisk

$$zisk = \sum_t \sum_a a(t) \rightarrow isStanding == True$$

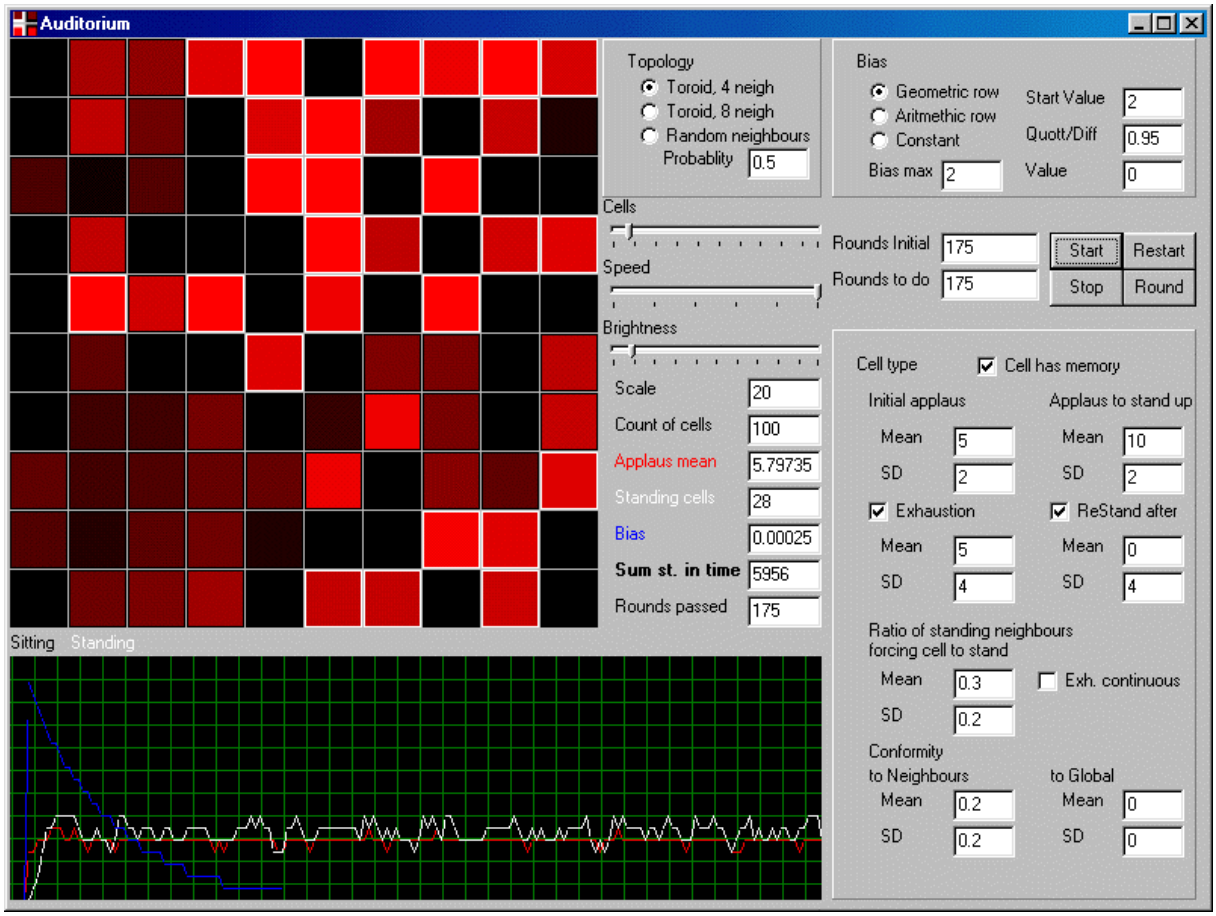
a minimalizujúcu náklady

$$náklady = \sum_t bias(t)$$

kde $a(t) \rightarrow isStanding$ je true, ak jedinec a v čase t stojí, false, ak nie, a $bias(t)$ je bias v čase t.

Popis programu (brutto)**:**
Držiteľom informácie o stave publika je objekt **TAuditory**. Jednu bunku (t.j. jedinca) reprezentuje objekt **TCell**. TAuditory udržuje TCell-y v zozname. Na uskutočnenie kola má TAuditory metódu **round**. Táto vykoná jedno kolo programu, ktoré spočíva v prerátaní pre každú bunku (TCell::recalculate) a nastavení nových hodnôt (TCell::set). Metóda recalculate spolu s bias funkciou určuje vlastné chovanie systému. Prístupné sú rôzne topológie susednosti, tiež rôzne algoritmy chovania sa buniek.

Práca s programom:
Obrazovka je rozdelená na 3 hlavné časti: vlastné hľadisko, graf zachycujúci niektoré parametre v čase a ovládacie prvky:



Hľadisko je štvorcová sieť buniek. Každá bunka je zobrazená červeným štvorčekom, ktorého jas zodpovedá sile aplauzu bunky. Na prispôsobenie farieb je regulátor jasu **Brightness**, ktorý posúva hranicu aplauzu, pri ktorom má bunka maximálny jas. Ak bunka "stojí", má biely okraj, ak "sedí", čierny. Potenciometer **Cells** nastavuje počet buniek, od 9 (štvorec 3 x 3) po 1000 (štvorec 100 x 100). **Speed** nastavuje rýchlosť simulácie. Tá sa spúšťa tlačítkom **Start**, prerušuje tlačítkom **Stop**. **Round** vykoná jedno kolo simulácie. **Restart** znovu reštartne hľadisko podľa údajov v ovládacích prvkoch. Riadok **Rounds to do** určuje, po koľkých krokoch sa simulácia zastaví. Ak sa po zastavení hodnota zmení, ďalej sa pokračuje, kým sa táto hodnota nedosiahne. Pri reštarte sa hodnota nastavi na **Rounds initial**.

Legenda:

Count of cells	počet buniek v hľadisku
Applaus mean	priemer aplauzu buniek *(v tomto momente)*
Standing cells	počet stojacich buniek *(v tomto momente)*
Bias	momentálna hodnota biasu *(v tomto momente)*
Sum st. in time	funkcia *zisk* z vyššie, suma za každú stojacu bunku v každom čase + 1
Rounds passed	počet kôl, ktoré zatiaľ prebehli

V **grafe** sú zakreslené hodnoty **Applaus mean, Standing cells, Bias,** pričom farba zodpovedá farbe v legende. Pri reštarte sa vynechá krátka medzera, na odlíšenie. **Scale** nastavuje mierku grafu.

Topológie:
Nastavujeme ich prepínačmi. **Toroid, 4 neigh** je toroid (mriežka, ktorej krajné prvky sú navzájom cyklicky spojené – ľavý krajný s pravým v tom istom riadku, podobne horný s dolným v tom istom stĺpci) so 4 susedmi, **Toroid, 8 neigh** s 8 susedmi, Random je topológia, kde sa každému prvku náhodne vyberú susedia podľa algoritmu:

```
Výber susedov pre bunku i:
while (random<probablity)
      vyber náhodne prvok, pridaj do zoznamu susedov i
```

kde `probablity` je pravdepodobnosť nastavená v **Probablity**, random je generátor s rovnomerným rozdelením, random je z intervalu (0,1). Probablity sa teda dá interpretovať ako pravdepodobnosť, že sa k nejakému prvku ešte pridá sused.

Bias:
Funkcie biasu sa nastavujú prepínačmi, kde **Geometric row** je geometrický rad, **Arithmetic row** je aritmetický rad a **Constant** je konštantná funkcia, parametre radov sú **Start value** (štart. hodnota), **Quott/Diff** (v prípade geometrického radu kvocient, v prípade aritmetického rozdiel – pre klesajúcu funkciu teda rozdiel < 0), parametrom konštantnej funkcie je **Value**, čo je jej hodnota. **Bias max** určuje hodnotu, ku ktorej budú porovnávané hodnoty aktuálneho biasu v grafe.

Predtým, než popíšem nastavenie typu bunky, považujem za potrebné zmieniť sa o generátore náhodných čísel. Používam normálne rozdelenie, podľa algoritmu:

```
float Gauss()
{
    float a,b;
    if(gauss_help==0)
    {
        a=sqrt(-2.0*ln(random));
        b=2*PI*random;
        gauss_savegaussdev=a*cos(b);
        gauss_help=1;
        return a*sin(b);
    }
    else
    {
        gauss_help=0;
        return gauss_savegaussdev;
    };
};
```

ktorý generuje hodnoty s rozdelením N(0,1), ak potrebujem N(Mean,SD), používam `Gauss*SD+Mean`.

Parametre buniek:
(vždy je použité normálne rozdelenie N(Mean, SD), kde Mean, SD sú hodnoty z riadkov prislúchajúcich parametrom)

- ***Initial applaus*:**
  bunky sa na začiatku inicializujú s aplauzom s normálnym rozdelením so zadanými parametrami;
- ***Ratio of standing neighbours forcing cell to stand:***
  určuje, pri akom pomere (*počet stojacich susedov bunky*) / (*počet susedov bunky*) sa bunka tiež postaví;
- ***Conformity:***
  aplauz bunky sa v kole vyráta ako vážený priemer aplauzu samotnej bunky, súčtu priemerného aplauzu v minulom kole a biasu a priemerného aplauzu bezprostredných susedov:

  $$newApplaus = (1 - confToNeigh - confToGlobal) * applaus + confToNeigh * meanApplausOfNeigh + confToGlobal * (meanApplausOfAll + Bias)$$

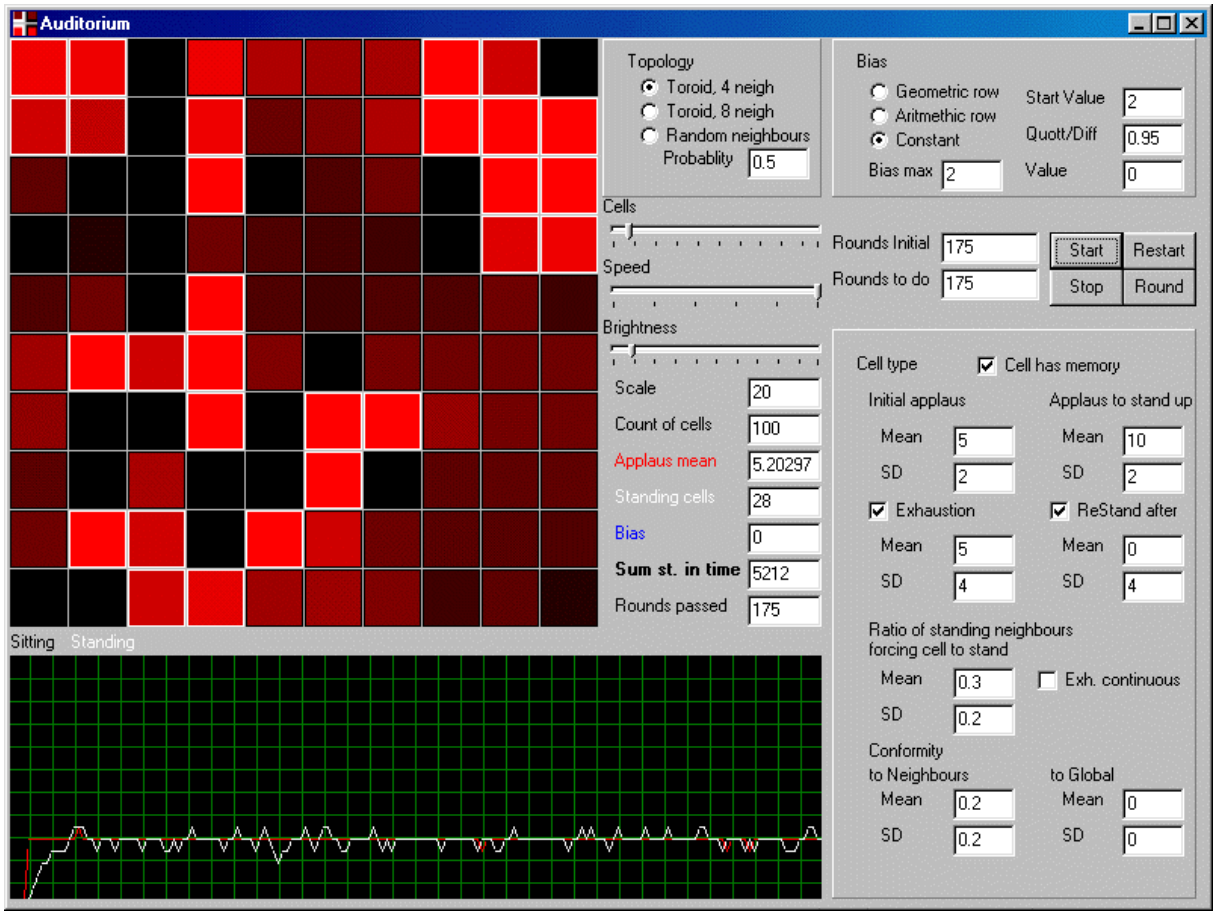
  kde $confToNeigh$ je konformita s bezprostrednými susedmi, $confToGlobal$ je konformita so všeobecným hlukom, $applaus$ je aplauz v minulom kole, $meanApplausOfNeigh$ je priemerný aplauz susedov v minulom kole, $meanApplausOfAll$ je priemerný aplauz všetkých v minulom kole, $Bias$ je budiaca funkcia v tomto kole; parametrami rozdelenia konformít teda modelujeme ovplyvniteľnosť bunky alebo jej zotrvačnosť;
- ***Applaus to stand up:***
  ak bunka tlieska aplauzom aspoň ako tento parameter, postaví sa; ak sa postaví kvôli

tomu, že stoja jej susedia (Ratio of standing neighbours forcing cell to stand), tak ak by podľa predchádzajúceho vzorca mala tlieskať slabšie ako je táto hodnota, bude tlieskať aspoň silou veľkosti Applaus to stand up;

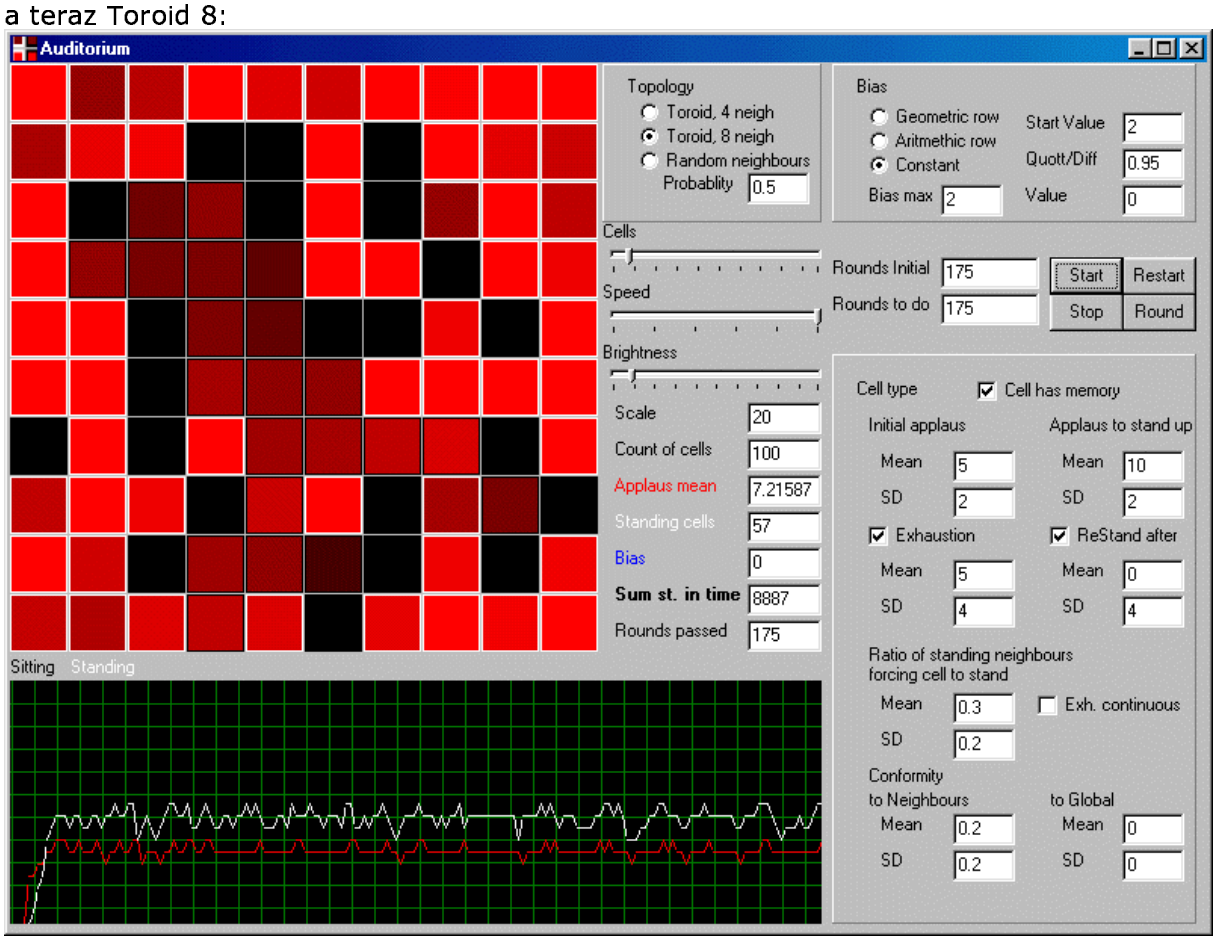
- ***Exhaust:***
  zaškrtnutím tohto parametra sa určuje, či je aktívny alebo nie; ak je, tak bunka, ktorá stojí počet kôl aspoň ako hodnota tohto parametra, sa vyčerpá, sadne si a úplne prestane tlieskať, a pokiaľ nie je zaškrtnutý ďalší parameter ReStand, už sa nepostaví;
- ***Exh. continuous:***
  ak je zaškrtnutý, tak nato, aby sa bunka vyčerpala, musí stáť súvisle, ak nie je zaškrtnutý, stačí stáť celkovo aspoň Exhaust kôl;
- ***ReStand after:***
  ak je aktívny a je aktívny aj Exhaust, tak sa bunka pri sadnutí si kvôli vyčerpaniu po sedení počas počtu kôl ReStand znovu zapája do potlesku; ak je Exhaust vypnutý, tento parameter nemá žiadny vplyv;
- ***Cell has memory:***
  bunky rozdeľujeme na bunky s pamäťou a bez nej; tie s pamäťou dostanú na začiatku parametre (s normálny rozdelením s parametrami Mean a SD), ktoré si už počas výpočtu nemenia (teda ak nejaká bunka s pamäťou dostane hodnotu parametra Ratio of standing neighbours forcing cell to stand 0,2, tak sa táto hodnota pre ňu počas ďalších výpočtov nezmení; bunka bez pamäte dostane parametre rozdelenia, a v každom kole si určí hodnotu tohto parametra znova náhodne s daným rozdelením); pamäť sa týka všetkých parametrov, okrem Initial applaus; ak je tento parameter zaškrtnutý, bunky majú pamäť;

Experimentálne výsledky a ich analýza:

Rozdiely v topológii:
- sú podstatné, tu je porovnanie (najprv krátky grafický argument):
Toroid 4:



a teraz Toroid 8:



Pri Toroide 8 sú hodnoty **Sum st. in time** vyššie (8887 ku 5212). Poďme teda skúmať tento jav bližšie. Použijeme Random topológiu (tá umožňuje plynule zvyšovať počet veľkosť susedstva). Odvoďme najprv strednú hodnotu počtu susedov pri hodnote parametra Probablity *p*, *i* je počet susedov.

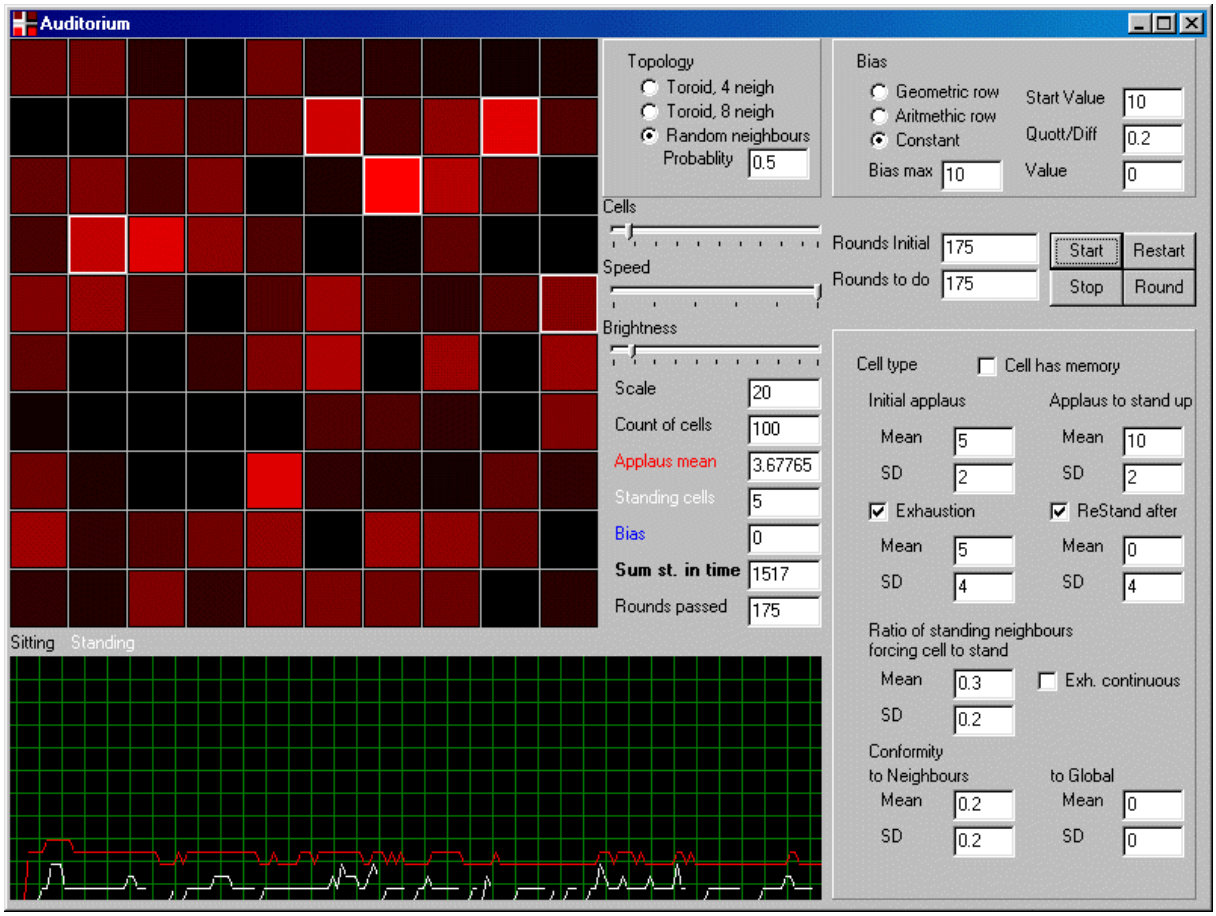
$$P(i=0)=(1-p)$$
$$P(i=1)=p(1-p)$$
$$P(i=j)=p^j(1-p)$$

Zrátajme strednú hodnotu funkcie počtu susedov *S*.

$$E(S)=\sum_{i=0}^{\infty} i\cdot p^i(1-p)=\frac{p}{1-p}$$

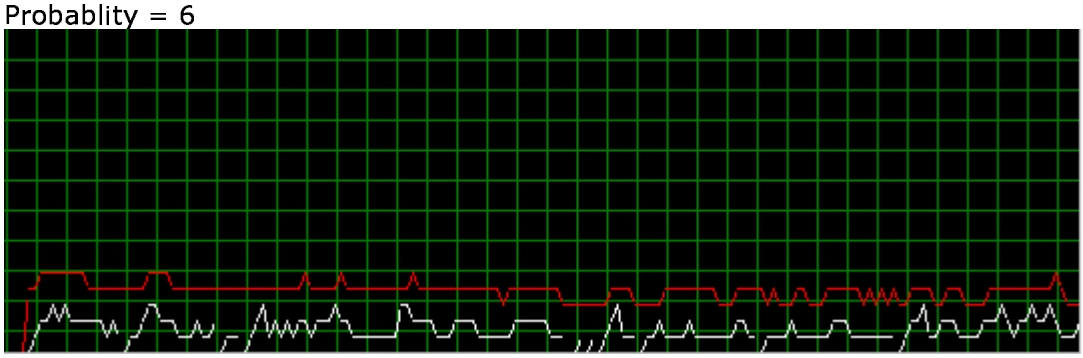
(zrátané cez $\left(\sum_{i=0}^{\infty} p^i\right)'=\left(\frac{1}{1-p}\right)'$).

Zvyšovali sme **Probablity** od 0.5 (každý má priemerne jedného suseda) po 0.95 (priemerne 19 susedov) po 0.05. Pre každú hodnotu 3 merania **Sum of st. in time**, z nich aritmetický priemer. Najprv nastavenie parametrov bunky:

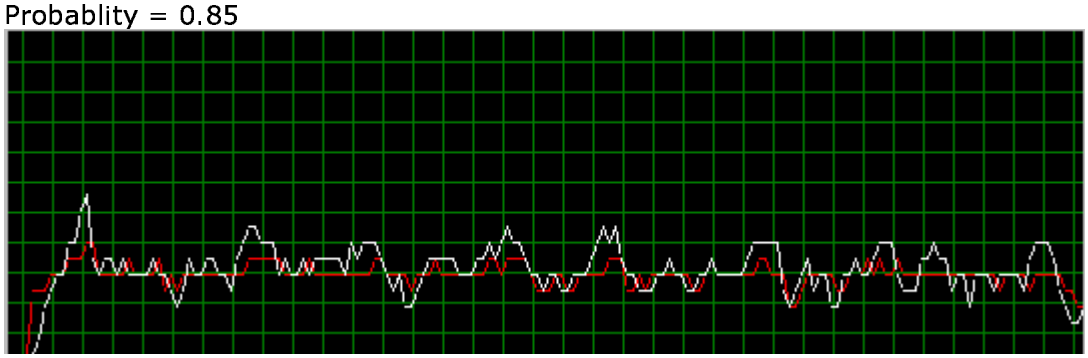


príklady niektorých grafov počas výpočtu:

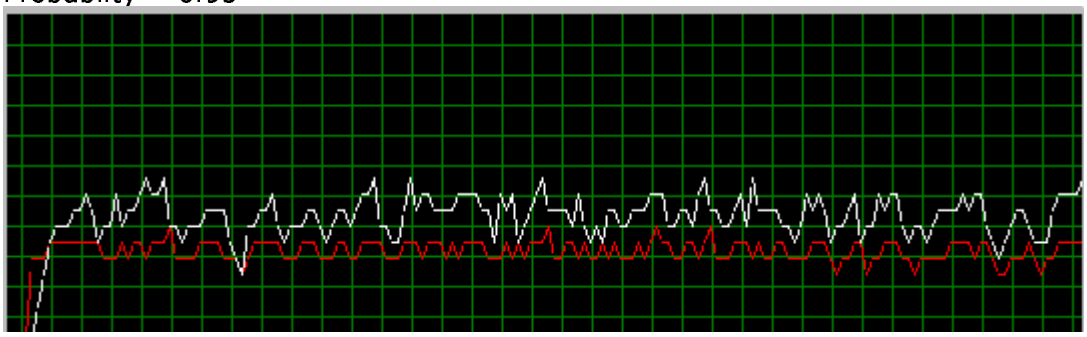
Probablity = 6



Probablity = 0.85



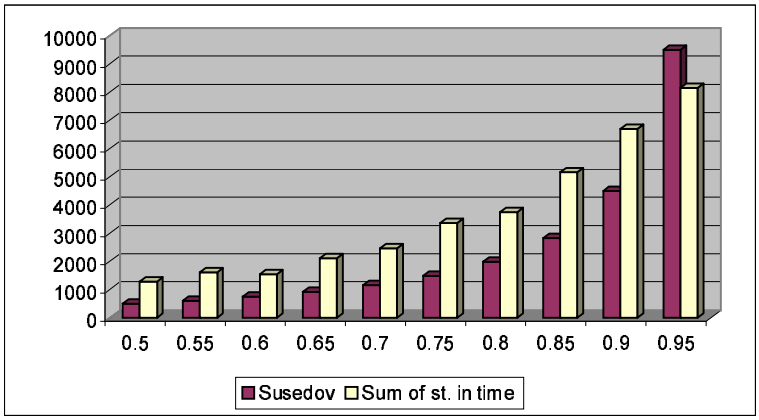
Probablity = 0.95



tabuľka meraní:

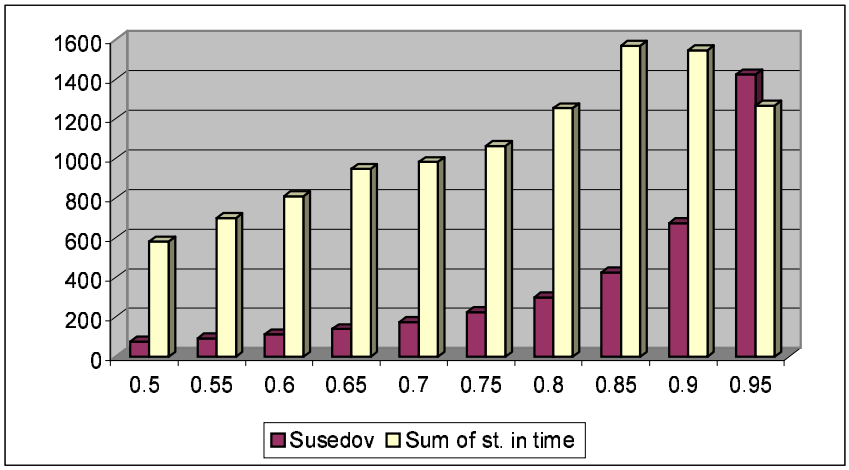
Porovnanie vplyvu počtu susedov na chovanie modelu						
Probablity	Priem. susedov	meranie **Sum. st. in time**				priemer
		1	2	3	4	
0.5	1.00	1527	1533	1202	864	1281.5
0.55	1.22	1752	1118	1554	2062	1621.5
0.6	1.50	1901	1459	1118	1749	1556.75
0.65	1.86	2493	2039	2104	1860	2124
0.7	2.33	3127	2035	1747	2990	2474.75
0.75	3.00	2470	3632	3298	4057	3364.25
0.8	4.00	3409	4186	4310	3065	3742.5
0.85	5.67	5543	4844	4541	5706	5158.5
0.9	9.00	5680	6244	7367	7457	6687
0.95	19.00	8271	8293	8132	7912	8152

graf:



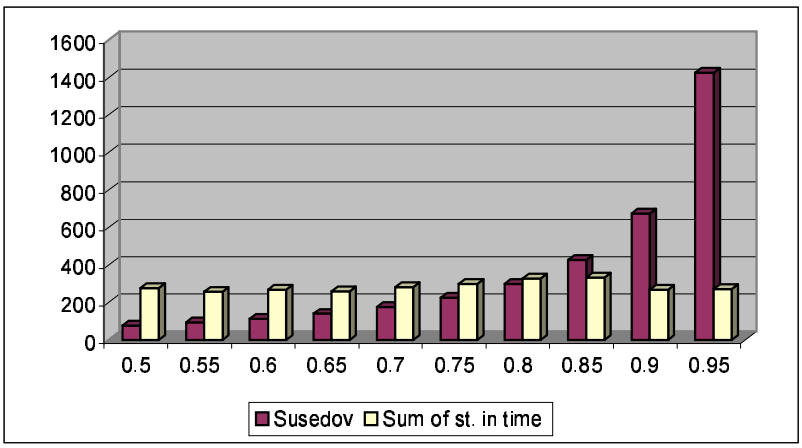
Na základe týchto údajov možno usudzovať, že väčší počet susedov spôsobí väčší počet postavení. Skúsme však zmeniť **Ratio of standing neighbours forcing cell to stand** na Mean = 0.4 (ostatné parametre ako v predch. prípade):

Porovnanie vplyvu počtu susedov na chovanie modelu 2						
Probablity	Priem. susedov	meranie **Sum. st. in time**				priemer
		1	2	3	4	
0.5	1.00	793	417	427	692	582.25
0.55	1.22	719.00	607	736	736	699.5
0.6	1.50	564	1055	766	859	811
0.65	1.86	863	833	874	1220	947.5
0.7	2.33	1112	620	1100	1102	983.5
0.75	3.00	1021	1120	1124	996	1065.25
0.8	4.00	1206	886	1313	1613	1254.5
0.85	5.67	1244	1356	2154	1524	1569.5
0.9	9.00	1160	1887	1357	1782	1546.5
0.95	19.00	1332	1273	1391	1069	1266.25

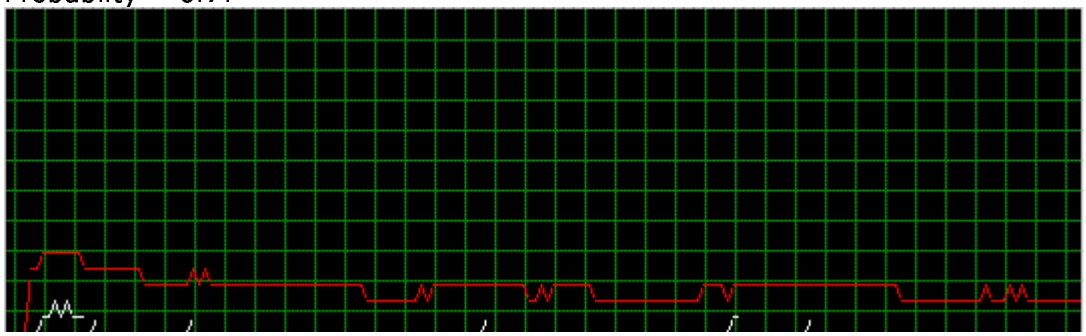


A ešte zopár meraní pre Ratio Mean = 0.5:

Porovnanie vplyvu počtu susedov na chovanie modelu 3						
Probablity	Priem. susedov	meranie **Sum. st. in time**				priemer
		1	2	3	4	
0.5	1.00	301	245	322	241	277.25
0.55	1.22	245	249	283	246	255.75
0.6	1.50	320	295	211	244	267.5
0.65	1.86	256	241	269	268	258.5
0.7	2.33	325	233	330	244	283
0.75	3.00	367	249	322	256	298.5
0.8	4.00	276	301	428	301	326.5
0.85	5.67	419	268	244	404	333.75
0.9	9.00	297	250	257	267	267.75
0.95	19.00	224	195	336	325	270



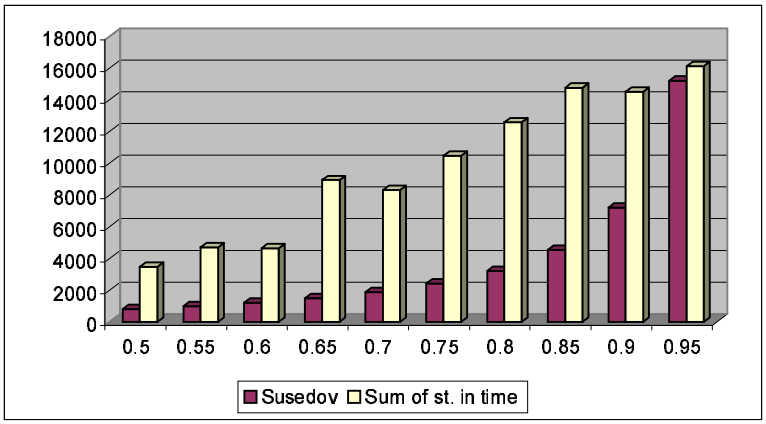
Probablity = 0.7:



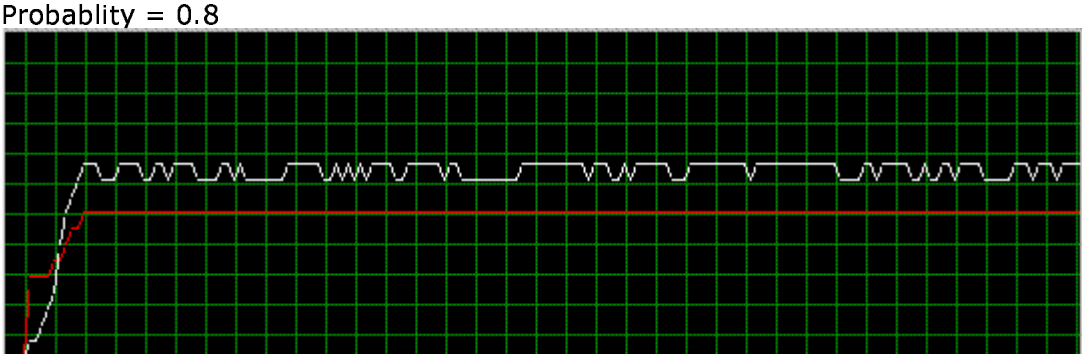
Zhrňme pozorovanie slovne: pre Ratio Mean 0.3 vidíme jasnú závislosť medzi počtom susedov a **Sum of st. in time** – ak zvýšime počet susedov, zvýši sa suma. Pri Ratio Mean = 0.4 sa model chová podobne po 0.85, potom však suma z rastúcim počtom susedov klesá. Pre Ratio Mean = 0.5 sa model nechová výrazne odlišne pre rôzne počty susedov, pričom platí, že hlavná časť sumy sa nasčíta už na začiatku (viď graf pred týmto odstavcom – veľký biely peek na začiatku, potom už len sporadické malé peeky). Pokúsme sa tieto výsledky analyzovať a interpretovať:
Uvažujme skupinu K blízkych buniek, majúcich medzi sebou L prepojení. Používame bunky bez pamäte, tzn. na začiatku kola sa im zase zvolia hodnoty závislosti od susedov a hodnota ATS (Applaus to stand up). Teda v istom počte prípadov ATS klesne pod aplauz bunky, a táto sa postaví. Pozrime sa však na druhú možnosť postavenia sa - a to postavenie sa preto, lebo stojí istý počet mojich susedov. Ak mám len jedného suseda, a ten stojí, potom stojí 100% mojich susedov a teda sa určite postavím pri každom nastavení pomeru R (Ratio of standing neighbours forcing cell to stand, jeho Mean). Čím mám viac susedov, pri tom istom R, tým viac sa vďaka náhodnosti spojení pomer mojich stojacich susedov rovná pomeru stojacich v celom hľadisku. A teda pri R=0.3 napríklad stojí v celom hľadisku >30%, a ja mám veľa susedov, tak je málo pravdepodobné, že moji sú práve tí, čo nestoja. A teda sa postaví viac buniek (tie zase spôsobia postavenie ďalších atď). Ako však vysvetliť, že pre R=0.4 už maximum sumy nie je pre P (Probablity) = 0.95 ? Kľúčom je parameter E (Exhaustion, jeho Mean). Ak totiž vypneme únavu, výsledky vyzerajú takto:

R=0.4

Porovnanie vplyvu počtu susedov na chovanie modelu 4						
Probablity	Priem. susedov	meranie **Sum. st. in time**				priemer
		1	2	3	4	
0.5	1.00	4126	3279	3561	2809	3443.75
0.55	1.22	4740	5780	4545	3682	4686.75
0.6	1.50	3895	6434	3581	4676	4646.5
0.65	1.86	8945	8956	8839	9008	8937
0.7	2.33	7525	9891	7013	8819	8312
0.75	3.00	10011	11835	9627	10472	10486.25
0.8	4.00	13629	10696	11456	14578	12589.75
0.85	5.67	14654	15051	14843	14463	14752.75
0.9	9.00	13985	13078	15837	15106	14501.5
0.95	19.00	16159	15568	16460	16356	16135.75



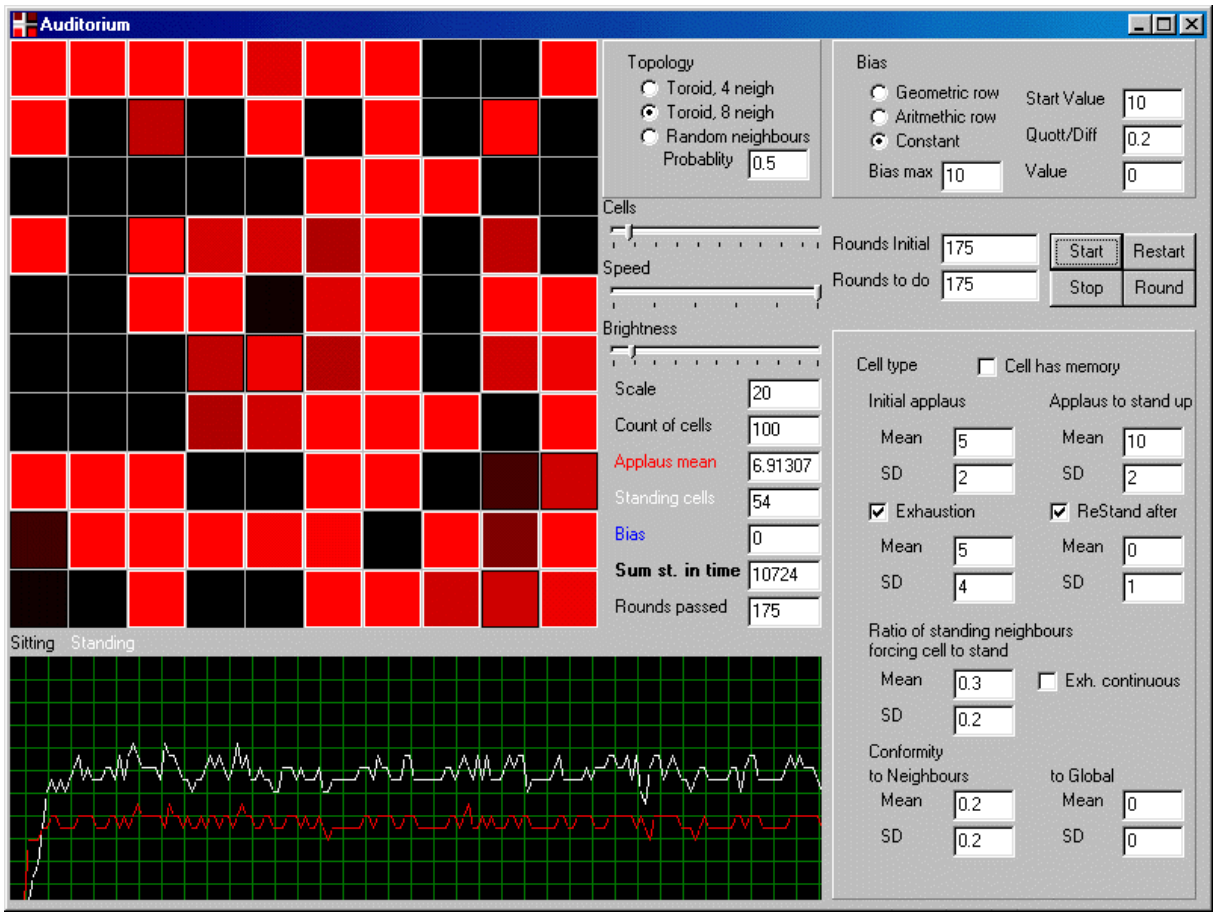
Probablity = 0.8



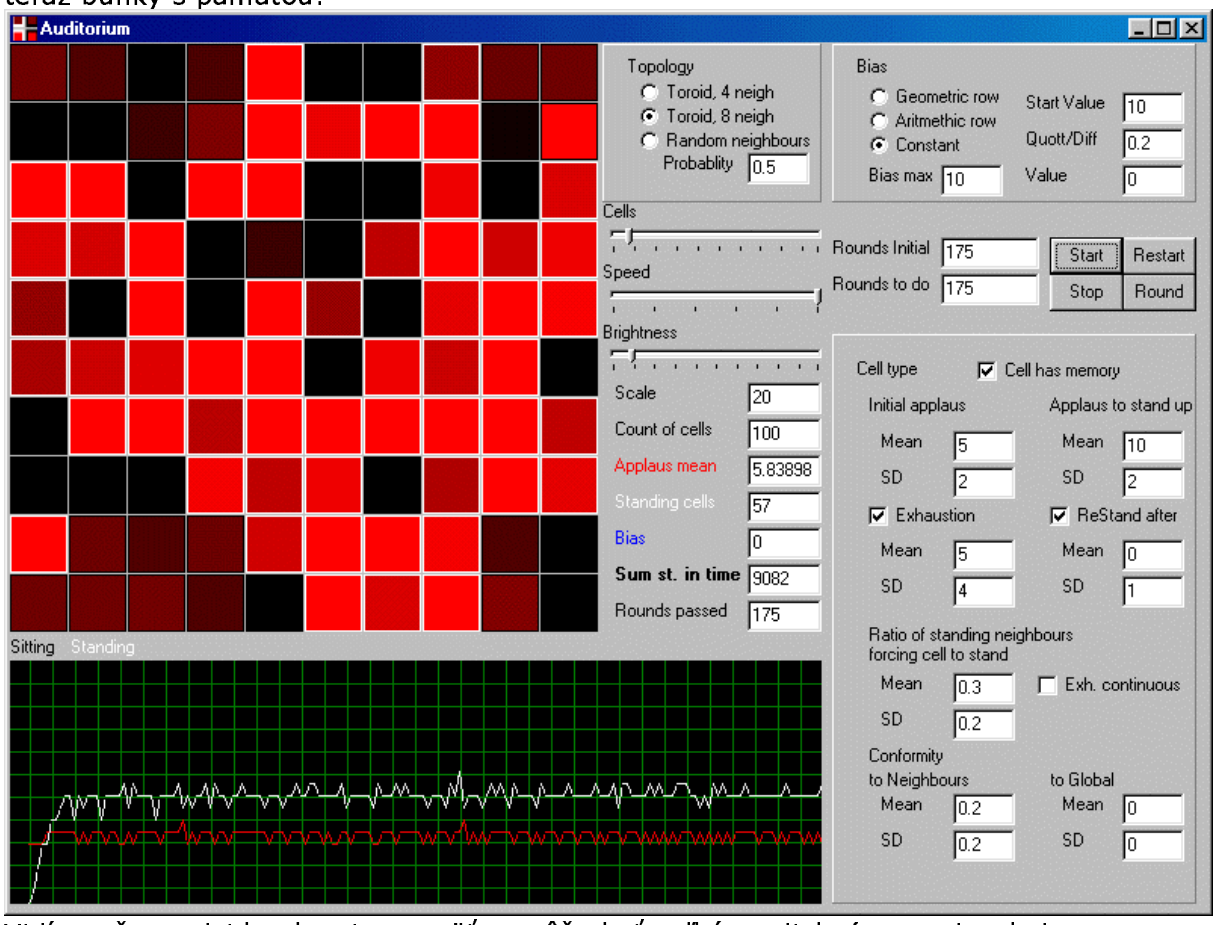
(pozn.: pri 100 bunkách a 175 kolách je suma z hora ohraničená 17500)

Teda, naše vysoké P=0.95 spôsobí, že sa naraz postaví príliš veľa buniek, ktoré sa naraz vyčerpajú, a tým spomalia proces.

Pamäť:
Skúmajme teraz vplyv pamäte (tzn. konštantnosti parametrov buniek) najprv bunky, ktorých parametre sa menia (tzn. bez pamäte):

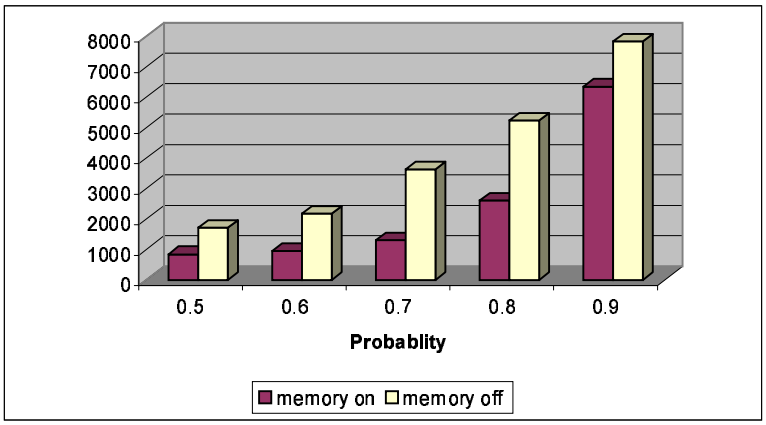


teraz bunky s pamäťou:



Vidíme, že medzi bunkami s pamäťou môžu byť veľké rozdiely (na predposlednom obrázku vidno "studenú olasť" po okrajoch, ktorá sa neroztlieska počas celého výpočtu), bunky bez pamäte sa chovajú omnoho uniformnejšie (je málo pravdepodobné, aby mala nejaká časť dlho rovnaký charakter).

Pri pohľade na tento graf *(topológia Random, menili sme Probablity, 4 merania pre každú hodnotu, z nich priemer (ostatné parametre ako vyššie))*:

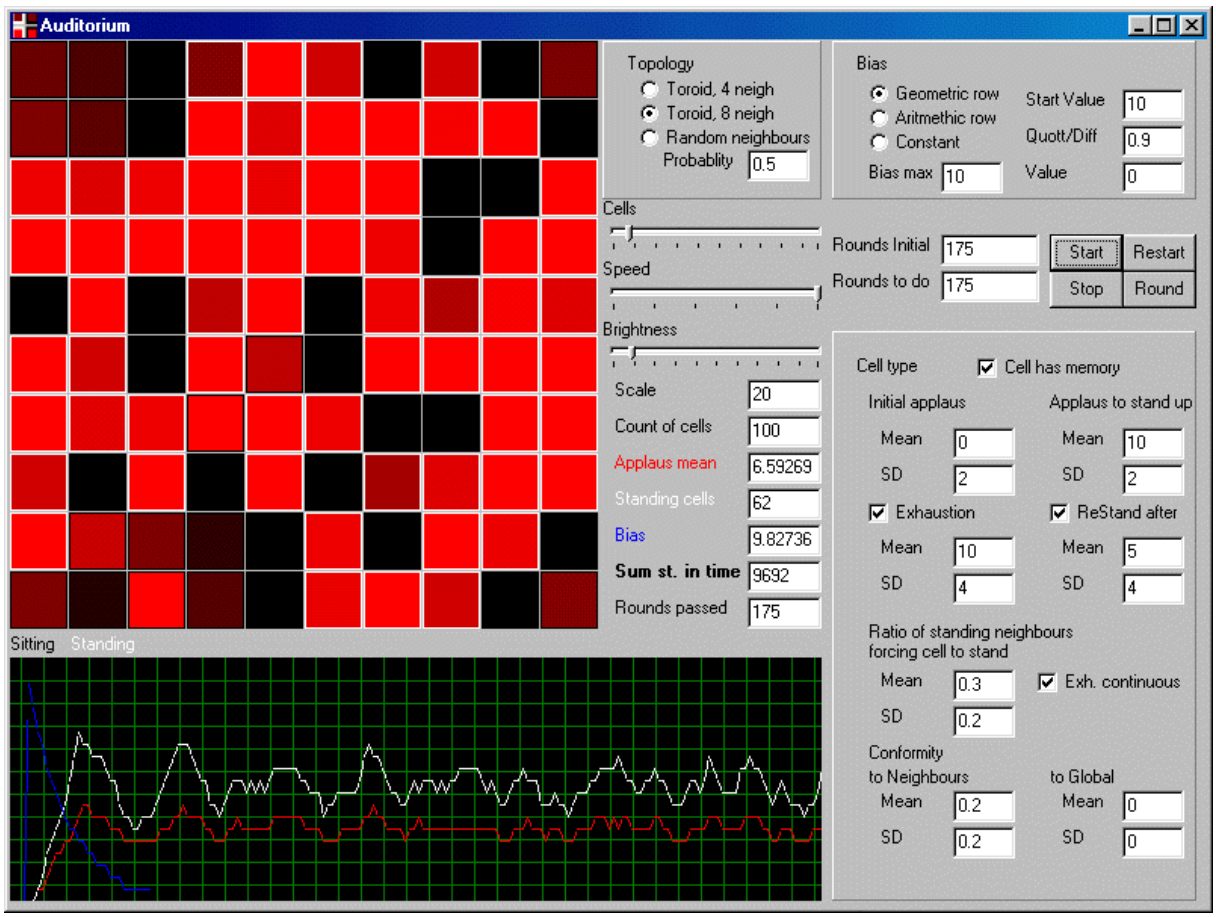


môžeme pozorovať oprávnenosť našich tvrdení. Bunky s pamäťou sa postavia častejšie, pričom môžeme pozorovať, že pre väčšie množstvo susedov sa rozdiel znižuje. Zdôvodnením rozdielu medzi pamäťami je: ak sa bunke s pamäťou na začiatku priradí vysoká hodnota ATS, a bias =0, počas výpočtu sa už nepostaví (keďže jej aplauz sa ráta ako vážený priemer vlastného aplauzu a aplauzu okolia, nebude vyšší ako maximum z týchto hodnôt). Naproti tomu v náhodnom modeli k tomuto nedochádza – parametre bunky sa stále menia, a teda nevznikajú chladnejšie oblasti. Čo sa týka znižovania rozdielu medzi zapnutou a vypnutou pamäťou, zdôvodnením môže byť náhodnosť modelu – náhodným výberom susedov zamedzíme vzniku oblasti s hranicou z netlieskajúcich

buniek. Ak som sám ochotný tlieskať, ale mám tichých susedov, nebudem tlieskať – a ak ich mám len 2, šanca že sú obidvaja tichý je vyššia, ako keď ich je 20.

Exhaust continuous:
Na otázku, či **Exh. continuous** spôsobí badateľnú zmenu môžeme odpovedať kladne:

nastavenie parametrov buniek:



Porovnanie vplyvu parametra súveslej/nesúvislej vyčerpanosti						
	Meranie					priemer
	1	2	3	4	5	
súvislá	11500	10102	10162	10907	10541	10642.4
nesúvislá	9815	10352	8327	9292	8953	9347.8

Je to ľahko vysvetliteľné – pri nesúvislej vyčerpanosti si bunka sadne aj v prípadoch, keď si pri súvislej nesadne, a ak si sadne pri súvislej, určite si sadne aj pri nesúvislej.

Záver:
Postavili sme model na simulovanie úlohy. Pozorovali sme vplyv zmien charakteristík buniek na konvergenciu k masívnej standing ovation so záverom – konvergencia je pravdepodobnejšia a rýchlejšia v prípade meniacich sa buniek. Rovnako pôsobí vyšší počet susedov.

Možné rozšírenia:
Je možné hľadať bias funkciu s minimálnymi nákladmi a maximálnym ziskom pomocou evolučných algoritmov.