VYSOKÉ UČENÍ TECHNICKÉ
V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

CELULÁRNÍ AUTOMATY (GAME OF LIFE)

CELULLAR AUTOMATA (GAME OF LIFE)

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

MAREK TOMAŠTÍK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. RADOMIL MATOUŠEK, PH.D.

BRNO 2010

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2009/2010

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Marek Tomaštík

který/která studuje v **bakalářském studijním programu**

obor: **Aplikovaná informatika a řízení (3902R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Celulární automaty (Game of Life)

v anglickém jazyce:

Cellular automata (Game of Life)

Stručná charakteristika problematiky úkolu:

Daná BP se bude zabývat v rešerní části rozborem a možnostmi tzv. Celulárních automatů (CA). Vysvětlí jejich princip a využití v oblasti simulace. Praktická čáast práce bude realizovat celulární automat typu „game of life“ v programovém prostředí Java.

Cíle bakalářské práce:

- Rešerše a základní popis problematiky celulárních automatů (CA).
- Tvorba aplikace implemenující CA ve formě tzv. Conway automatu – Game of Life.
- Flexbilní rozšíření aplikace automatu pro realizaci nových pravidel.
- Testování aplikace zaměřené na pravděpodobnost vzniku stabilních struktur.
- Vytvoření e-dokumentace k vytvořené aplikaci.

Seznam odborné literatury:

[1] Berlekamp, E. R.; Conway, John Horton; Guy, R.K. (2001 2004), Winning Ways for your Mathematical Plays (2nd ed.), A K Peters Ltd, ISBN 978-1-56881-130-7; ISBN 156881142X; ISBN 1568811438; ISBN 1568811446

[2] "Elementary Cellular Automaton". Wolfram Mathworld. http://mathworld.wolfram.com/ElementaryCellularAutomaton.html. Retrieved July 12, 2009

Vedoucí bakalářské práce: Ing. Radomil Matoušek, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2009/2010.

V Brně, dne

L.S.

Ing. Jan Roupec, Ph.D. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc. Děkan fakulty

Licenční smlouva

Abstrakt

Daná BP se v rešeršní části rozborem a možnostmi tzv. Celulárních automatů (CA). Vysvětluje jejich princip a využití v oblasti simulace. Praktická část realizuje celulární automat typu „Game of life“ v platformě Java.

Abstract

In this bacheloir theses was written basic informations about cellular automata, it´s function and usage in simulation. Second part of this work is aimed on celular automata Game of life in Java platform.

Klíčová slova:

Celulární automat, hra život, S. Wolfram

Key Words:

Celullar automata, Conway´s Game of life, S. Wolfram

Obsah:

1. Úvod

Obecně lze celulární automaty charakterizovat jako fyzikální modely určené k simulaci dané problematiky. V praxi se často jedná o 2D CA tj. 2D pole – mřížku - , jejíž elementy mohou nabývat různých hodnot. Nejčastěji se setkáme s hodnotovým rozsahem typu boolean tj. pouze 2 možné hodnoty – buňka je mrtvá nebo živá (stav buňky).

Stav buněk v následující generaci (reprezentaci diskrétního časového okamžiku) vypočte na základě zadané funkce, která je naprosto stejná pro všechny buňky dané mřížky. Funkce pracuje s hodnotami dané buňky a jejího okolí tj. hodnoty buněk v sousedství.

Celulární automaty nepatří mezi nejnovější poznatky v oblasti modelování, ovšem stále můžeme naleznout odvětví, kde je jich využíváno. Obecné části problematiky bude věnována rešeršní část této práce. [1]

2. Teoretický rozbor

Obecně je problematiky celulárních automatů poměrně rozsáhlá, což je způsobeno dobou vývoje. Bez ohledu na verzi je celulární automat dynamický systém, diskrétní v hodnotách, prostoru i čase. [1]

2.1 Historie

Vývoj CA začal v polovině 20. století. První, kdo se začal zabývat CA byl ve 40. letech minulého století maďarský matematik John von Neumann. Pro svou práci potřeboval nový dynamický model pro použití v biologii – reprodukce mikroorganismů. Vytvořil pravidelnou mřížku buněk, přičemž každá buňka byla považována za samostatný automat, celá mřížka za organismus.[1]

V roce 1970 navázal na Neumannovu práci britský matematik John Conway [2], když našel pravidlo – lokální přechodovou funkci, platnou pro všechny buňky – vedoucí ke komplexnímu chování. V souvislosti s následným vznikem HPP modelu mřížového plynu (název je zkratka jmen objevitelů – Hardy, Pomeau, de Pazzis) získaly celulární automaty schopnost zachovat mimo stavů buněk také pohyblivost, čímž se stali vhodným pro modelování pohybů tekutin případně dynamických látek.

V 80. letech byly rozvíjeny především jednodimenzionální CA, a to zásluhou britského matematiky a fyzika Stephena Wolframa. Dokázal (stejně jako jeho současníci Frisch, Hasslacher a Pomeau), že pří aplikaci omezujících podmínek se CA chová v souladu s Navierova-Stokesova rovnicemi tj. rovnicemi o proudění nestlačitelné tekutiny a je tedy možné využít CA k modelování fyzikálních problémů.[1]

Další se známých CA je Codův automat. Pracoval s osmi stavy a s neumanovským okolím. Čtyři stavy určovaly strukturu:[3]

0 - prázdná buňka(prvek)
1 - jádro signálové cesty
2 - obal signálové cesty
3 - speciální použití, např. pro hradlo
4,5,6,7 - byly signálové.

Základní prvek automatu byl realizován pomocí kombinované dvojce signálové a prázdné buňky. Každá další generace znamenala posun dvojce o jednu pozici na signálové cestě. Coddův automat byl teoreticky schopen emulovat Turingův stroj (teor. model počítače) a též vytvořit svou vlastní kopii.

Posledním zástupcem vycházejícím z Codova automatu jsou Langtonovy Q-smyčky. Jsou zde doplněny pro úplnost, ale této práce se již netýkají.

2.2 Typy CA

Z dříve popsané historie vyplývá několik typů celulárních automatů. Jako hlavní kritérium pro dělení se nabízí počet dimenzí mřížky. Celulární automaty tedy můžeme dělit takto:

Nejjednodušším typem je 1D dvojstavový CA podle vzoru Stephena Wolframa, který se v literatuře často vyskytuje pod názvem *Elementary Celullar Automata* – Elementární CA.[4]

Elementární CA resp. každá jeho buňka (příp. prvek) nabývá dvou hodnot a to 0 nebo 1. Hodnota dané buňky je přímo určena hodnotami tří buněk v jeho sousedství. Díky této zákonitosti můžeme určit vývoj dané buňky v následující generaci pomocí stavů buňky samotné a buněk sousedních tj. buňky zleva a zprava. Po ověření možných stavů získáme 2^3 možných stavů pro tuto trojici.

0 0 0 1 1 1 1 0

Obr1. Možné stavy buňky pro 1D CA lze vidět zde.

Celkový počet modifikací stejného automatu pomocí změny pravidel je 2^8=256, díky čemuž je v rámci automatu možno používat indexaci pomocí 8 bitového čísla. Pravidla pro změny stavů se nastavují dle potřeby. Nastavení, které je uvedeno na Obr.1 je pouze ilustrativní. Jedná se o jedno z možných pravidel.

.

Druhým rozšířeným typem je 1D k-stavový CA označovaný jako *Totalistic Cellular Automata* – „Souhrnný“ CA. [5]

Souhrnný CA spatřil světlo světa roku 1983, jeho tvůrcem byl také S. Wolfram a je svým principem velmi podobný předchozímu typu. Klíčové je stejně jako u předchůdce v jakém stavu se nachází tří buňky v sousedství, tedy daná buňka a soused zleva a zprava. Při použití Souhrnného automatu se však hodnoty trojce zprůměrují, což je dovoleno k-stavovou vlastností.

Počet stavů (barev) je popsán jako 3k – 2. V rámci automatu lze využívat počet stavů k indexaci velmi podobným způsobem jako u předchozího typu. V případě, že určíme k=3, indexace bude provedena pomocí kódu, jehož hodnota odpovídá stavům po převedení do číselné soustavy o základu 3. Potom kód $777=1001210_3$ je znázorněn na Obr2.

1 0 0 1 2 1 0

Obr2. Stavy při kódu $777 = 1001210_3$

Dalším velice rozšířeným typem je 2D CA obvykle dvoustavový. Nejvíce známým zástupcem této skupiny je *Game of Life* [6] vynalezená Johnem Conwayem a veřejnosti přiblížena ve sloupku Scientific American od Martina Gardenera v říjnu 1970.

Game Life tvoří v základu 2D mřížka, obsahující daný počet vyplněných (živých) buněk. V diskrétním časovém okamžiku tj. „nástupu další generace“ se mění stav každé buňky dle buněk v okolí. V tomto případě okolím nazýváme Moorovo okolí – tj. všechny, které se „dotýkají“, resp. mají společný alespoň bod – 8 buněk. U dané buňky je vždy překontrolováno všech osm okolních buněk, zda sou 0 nebo 1. Živé buňky (stav=1) se počítají a tento čítač následně porovnán s danými pravidly:

1) smrt	(stav=0)	-	čítač < 2 nebo čítač > 3	– buňka zahyne
2) přežití	(stav=1)	-	čítač = 2 nebo čítač = 3	- buňka přežívá
3) zrod	(stav=0/1)	-	čítač u mrtvé buňky = 3	- ožije

Pro inicializaci mřížky se kromě „ručního plnění“ používá vzorová předloha. K možnostem nastavení Game of Life bude věnována praktická část řešení problému. Nejzajímavější oblastí Game of Life je oblast stabilních struktur, vznikajících při průběhu hry. Této oblasti bude věnována následující kapitola.

Pomocí zavedení jiných pravidel vzniklo mnoho různých verzí této hry. Za zmínku stojí jedny z nejznámějších – HighLife, Hexlife, Hashlife. HighLife má namísto klasického pravidla 23/3 pravidlo 23/36, Hexlife se vyznačuje mřížkou ve tvaru šestiúhelníku, kdežto Hashlife je schopen díky specifické implementaci hashovacích tabulek přejít o mnoho generací vpřed.

3. Stabilní struktury

Jejich vznik je velmi podstatný pro výsledky práce programu. Stabilní struktury vznikají v mřížce náhodně v závislosti na původní předloze. Lze si to představit v souvislosti s mikroorganismy, kdy při pokusu rozložíte po desce resp. sklíčku mikroskopu, různé koncentrace mikroorganismů, vlivem vzájemného působení na určitých místech vznikají kolonie, které jsou stabilní, v čase se nemění. Pokud je v blízkosti této stabilní kolonie jiná nestabilní pravděpodobně dojde k narušení rovnováhy. Stabilní struktury u CA reaguje stejně.

Na tvar a typ struktur mají přímý *vliv pravidla automatu a počet generací*. Blíže bude nejlépe danou problematiku ukázat na automatu, který je pro tuto práci určen.

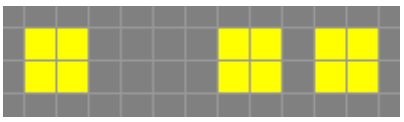
V původní Game of Life s pravidlem 23/3 se můžeme setkat se třemi základními typy stabilních struktur. Jedná se o tzv.: [7]

1) Still lives („stále živé“)
2) Oscillators (oscilátory)
3) Spaceships

Podrobně budou popsány v samostatných podkapitolách včetně ilustrací pro názornost.

3.1 Still lives („stále živé“) [8]

Prvním z této skupiny jsou blocks ,tedy bloky. Jedná se vlastně o čtyři buňky tvořící čtverec. Můžeme se také setkat s tzv. bi-blocks, což jsou dva bloky horizontálně nebo vertikálně na stejné úrovni oddělené mezerou.

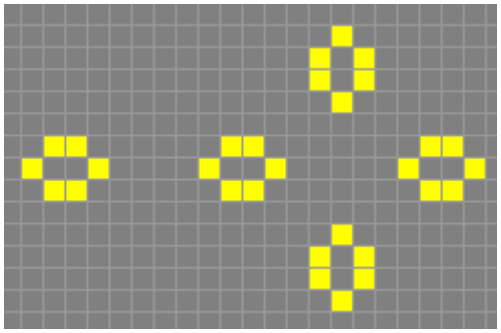


Obr3. Block a bi-blocks.

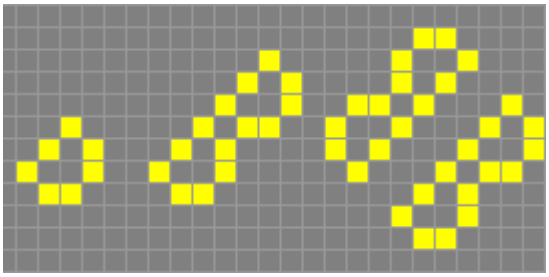
Dalším zástupcem této skupiny jsou hives neboli úly. Vyskytují se samostatně, případně ve skupině, která je v literatuře uváděna jako „honey farm“ tedy medová farma. Velmi podobná struktura zařaditelná společně s úly je loaf, tedy bochník. Je to statický zástupce stále živých buněk a jako předchozí dva typy tvoří větší skupiny – „bi-loaf“ a „bakery“ (pekárnu). Rozdíl je snadno rozeznatelný na následující ilustraci.

Velmi zajímavými zástupci ze skupiny Still life jsou také tube (a barge), boat a ship. Zajímavost této trojce spočívá především ve velké podobnosti jednotlivých členů a schopností neomezeně měnit svoji délku, aniž by se jednalo o jiný typ resp.

druh stabilní struktury. Jednotlivé struktury ze skupiny se od sebe postupně liší pouze jednou živou buňkou navíc. Viz. Obr4.



Obr5. Shive a „honey farm“.



Obr6. Loaf, bi-loaf a „bakery“.

Pro úplnost je třeba uvést, že je možno potkat například lodě nebo čluny v těsné formaci u sebe, což si můžeme představit jako člun tažený pomocí vlečného lana. Krom mnou zmíněných formací se lze setkat i s různými jinými, které zde, ale uvádět kvůli rozvětvenosti není vhodné.[8]

3.2 Oscillators (Oscilátory)

Jak napovídá samotný název podkapitoly druhým základním typem resp. formou stabilních struktur, se kterými se lze setkat jsou oscilátory, tedy takové struktury, které se s počtem generací přeměňují a posléze dosáhnou výchozího tvaru. Počet generací potřebných ke kompletní proměně závisí na „složitosti“ dané struktury. Na stránkách zdrojů lze v podobě animací vidět mnoho odlišných oscilátorů, které velmi dobře zobrazují dynamiku ve hře život. Pro zajímavost bych uvedl blinker, který je asi nejjednodušším zástupcem dané skupiny – jedná se pouze o rotující linii – a dále Kok´s galaxy (Kokovu galaxii), která je výrazně komplexnější a pro kompletní průběh potřebuje osm generací. [9]

3.3 Spaceships (vesmírné lodě) [10]

Poslední ovšem neméně významným (možná ještě významnějším než předchozí zmíněné) zástupcem stabilních struktur v Game of life jsou tzv. „vesmírné lodě. Podobně jako oscilátory, vesmírné lodě mění svůj „tvar“, resp. orientaci, v daném počtu generací. U vesmírných lodí v cyklech zůstává zachována orientace

s tím rozdílem, že její pozice je jiná, či že tento druh struktur může cestovat napříč mřížkou – mění svoji pozici. Počet generací k takovému kroku se obecně nazývá jako perioda.

V souvislosti s loděmi vesmírného typu se dostaneme k definici rychlosti. Obecně se CA používá přenesena rychlost světla z fyziky (tj. v CA posun o buňku za jednu generaci), přičemž obvykle mají různé lodě dělitele rychlosti světla (dle platných pravidel je součinitel značen jako c) a vyjádření rychlosti je tedy ve formě c/n – n je libovolné celé číslo. Například z nejznámějších zástupců, glinder má rychlost c/4, light-weight spaceship (lehko-tonážní vesmírná loď) c/2,tedy pro pohyb lodi o 2 je třeba čtyř generací. Obecně lze rychlost vyjádřit vzorcem:

$$v = \max\frac{(|x|,|y|)}{n}.c$$

, kde x a y vyjadřují maximální rozlohu mřížky, n zastupuje počet generací potřebných pro pohyb dané lodi a v neposlední řadě c bylo vysvětleno již dříve.

Tento formát rychlosti je praktickým důsledkem faktu, že posun lze hodnotit až při dosažení stejného tvaru a orientaci, tedy na posun o jednu buňku mřížky je zapotřebí více generací.

Jednotlivé lodě je možno vzájemně napojovat, jejich využití je v přenosu informace a při vytváření odzkoušených předloh jako jsou kupříkladu „reflektor“ – obrací dané lodě do jiných směrů, glinder gun – vystřeluje glinder formace. Jejich využití zasahuje i do jiných modifikací CA než je námi řešená Game of life.

Celkové povědomí o stabilních strukturách je důležité zejména k pochopení komplexního chování CA včetně Game of life, a tím i k možnostem využití, kterým je věnována následující kapitola.

4. Využití CA

Představit si možnosti celulárních automatů v prostředí simulování (resp. modelování) jistě není nijak těžké. K přesnému zaměření a použití nám tato představa ovšem nestačí. Doufám, že nastínění hlavních oblastí a jejich principů povede ke zlepšení představy o využití CA.

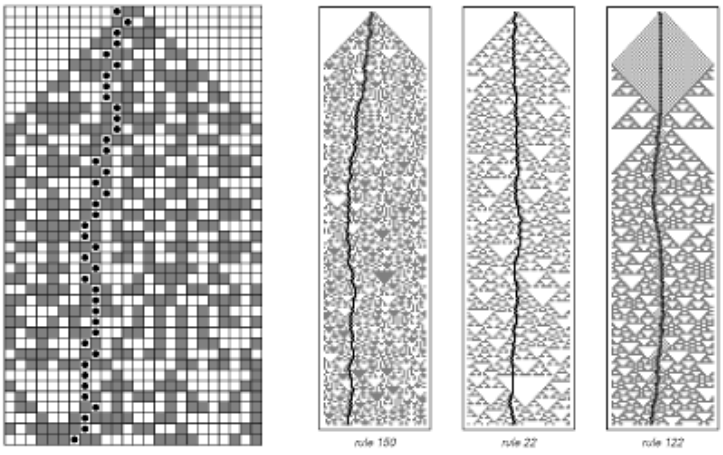
4.1 Růst krystalu

S růstem krystalu se můžeme nejčastěji setkat u sněhových vloček. Při bližším pohledu – obecně na mikroskopické úrovni – je zřejmé, že vločky vznikají podle pravidelné mřížky, se kterou se setkáme i CA. Ke vzniku krystalu dochází při ochlazení plynu a tekutin krystalu k bodu mrazu. Počátek (střed) tvoří zrnko prachu, na nějž se postupně nabalují další atomy. Převedením tohoto známého principu na celulární automat mřížkové pole (obvykle hexagonální tvar) kolem středu reprezentuje pevné částice (černé neboli linie) a plyn nebo tekutiny (bíle pole-mrtvé). Rozšiřování krystalu je v tomto případě reprezentováno přechod buňky do živého stavu.

Zvolené pravidlo automatu v důsledku ovlivní růst krystalu, tak že krystal dosahuje „stromového“ tvaru nebo naopak tvaru s hladkou hranou. Toto je v přírodě způsobeno interakcemi při růstu krystalu, tedy tlakem a změnami teploty při přeměně páry v led. U automatů je dosaženo stejných výsledků pomocí jednoduchého pravidla, kdy buňka oživne v závislosti na počtu živých sousedů v předchozí generaci, což je stejné ovšem buňky, které oživnou, již neumírají. Tím při různých inicializačních stavech vznikají různé typy, což je shodné s různorodostí sněhových vloček v přírodě. [11]

4.2 Porušování materiálu

V souvislosti s celulárními automaty a jejich simulační schopností si pod pojmem porušování materiálu představujeme inicializace a šíření trhliny, kdy po dosažení kritické velikosti dojde k lomu. Zde je poněkud méně jasné, jak lze komplexní problematiku jako porušování materiálu popsat za pomoci jednoduchých pravidel CA. Na základní rozlišovací úrovni to ovšem lze pouhou změnou pravidel.



Obr7. Průběh lomu v materiálu simulovaný CA.[12]

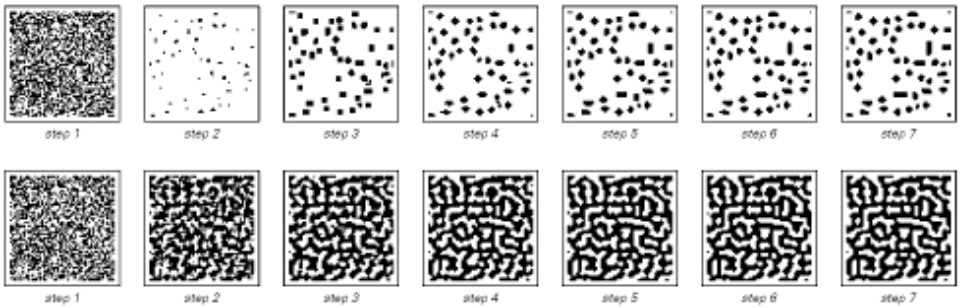
Průběh lomu v automatu je vyjádřen (viz Obr7.) Černé buňky vyjadřují vzniklé dislokace, černá tečka označuje pozici trhliny. Aplikací určitého pravidla jsou vyhodnocovány spojování dislokací tj. stejně jako při vzniku mikrotrhliny dochází k propojení dislokací a vzniku resp. rozrůstání nejvýznamnější. Speciality jednotlivých materiálu zahrnuje základní pravidlo v podobě hustoty jinak náhodných dislokací.

4.3 Elementární přínosy v biologii

Na biologické systémy je často pohlíženo jako na ukázku velmi komplexního chování a tento způsob pohledu obrovskou měrou znesnadňuje pochopení a vůbec zjištění, jak daný biologický systém funguje. Proto některé přístupy nahlížení na tuto problematiku, jakoby začínali z druhého konce, tj. při pohledu na elementární části se lze vymanit z komplexní smyčky a pochopit základní funkční principy, ze kterých se ve skutečnosti skládá komplexní chování systému.

Příkladem zmíněného přístupu mohou být genetické algoritmy např. včelí algoritmus používaný pro optimalizaci. Převzetím základních pravidel v komunikaci a schopnosti vyhledat potravu u každého jedince, vzniká komplexní chování celého hejna, na které již pohlížíme jako na komplexní chování skupiny, aniž bychom potřebovali vnější řízení. Podobný přístup se využívá také u CA.

Ukázkovým a přesto jednoduchým příkladem využití CA je modelování barevného pigmentu mušlí. Pigment vylučovaný živočichem sklouzavá po mušli a její nová barva vzniká na základě předchozího stavu a barvy v okolí tj. stavu sousedů. Přesněji řečeno vývoj takového automatu při každém kroku závisí na váženém průměru výskytu okolních barev (pro okolí+3). Nejbližší okolí má váhu 1, přičemž okolí +2 má -0,4 a +3 pouze -0,2 [13], pokud je vážený průměr kladný, buňka zčerná, v opačném případě zbělá. Z původní předlohy rychle vzniká pigment různého tvaru se kterým se setkáváme u různých organismů, a jehož vývoj je naznačen na Obr8.



Obr8. Postupně dochází k zaplnění do vzoru dle startovních pravidel.[13]

5. Aplikace a její tvorba

Praktickou a hlavní součástí této práce bylo vytvoření aplikace v programovacím jazyku Java, na základě CA známého pod názvem „Game of Life“, rozšířené o možnosti, které budou rozebrány později a následné testování vytvořeného programu s důrazem na „četnost“ stabilních struktur, jejichž druhům a možnostem byla věnována pozornost v předchozích kapitolách.

5.1 Třídy aplikace

Pro správné pochopení funkce daného programu budou nastíněny hlavní třídy aplikace, jejich metody a důležité parametry.

- *Cell.java*
- *GameCellGrid.java*
- *CanvasGrid.java*
- *Interface.java*

Jak bylo zmíněno už dříve, celulární automat typu „Game of Life“ je obvykle reprezentován 2D mřížkou, přičemž každá „mezera“ vyhrazená touto mřížkou zastupuje nám již známou buňku. Logicky je tedy první třída Cell.java, která definuje parametry každé buňky, jakou jsou například souřadnice, status (živá/mrtvá), status v další generaci – důležitý pro správné zobrazování a dva další číselné parametry, označující počet aktivních sousedních buněk a indikátor resp. čítač změny buňky v dané generaci.

Druhá z uvedených tříd, tedy GameCellGrid.java, obstarává deklaraci dvojrozměrného pole instancí předchozí třídy – pole objektů – dle současné velikosti mřížky resp. plátna (český název dílčí komponenty programovacího prostředí). Instance třídy GameCellGrid.java je vytvořena při každé změně velikosti okna programu, ať se jedná o roztažení okna nebo využití předefinovaných rozměrů, určených poměrem sloupců a řádků. Aktuální velikost se zobrazuje na panelu ovládání. Tyto počty jsou získány prostým vyděleným rozměrů komponenty Canvas, tento přístup byl zaveden pro zjednodušení a správnou reakci při jakékoli změně, ale má jisté omezení, které je skryto ve využití celočíselného dělení – toto bude ozřejměno v návodu na použití aplikace, jelikož se nejedná o závažný problém.

Třetí v pořadí je třída, odpovědná za většinu funkcí vytvořeného programu. CanvasGrid.java má základ v třídě Canvas (plátno pro kreslení) v programovém prostředí Netbeans, používaného mimo jiné pro tvorbu aplikací v jazyce Java. Původní třída byla zásadně rozšířena, metody a podstatné atributy si probereme blíže.

Metody pro inicializaci zodpovídají za nastavení rozměrů (sem patří i nastavení počtu sloupců a řádků mřížky pro pozdější vytvoření instance třídy GameCellGrid.java) a vykreslení mřížky na podkladovou komponentu, za pomoci přímek v barvě popředí. Dané dvě metody jsou následně volány defaultní metodou paint(), což zaručuje pozdější automatické přenastavení – metoda paint(), je vázána

na grafickou knihovnu Java.awt.Graphics a proto je volána při každé změně podkladové komponenty (změna rozměrů, aktualizace atd.).

Následující dvojce metod s názvy Vykresli/Vykresli2 mají, jak již sám název napovídá za úkol vykreslit buňky v mřížce v závislosti na stavu jejich atributu live. První uvedená obsluhuje změny provedené myší, druhá je volána při samotném běhu aplikace.

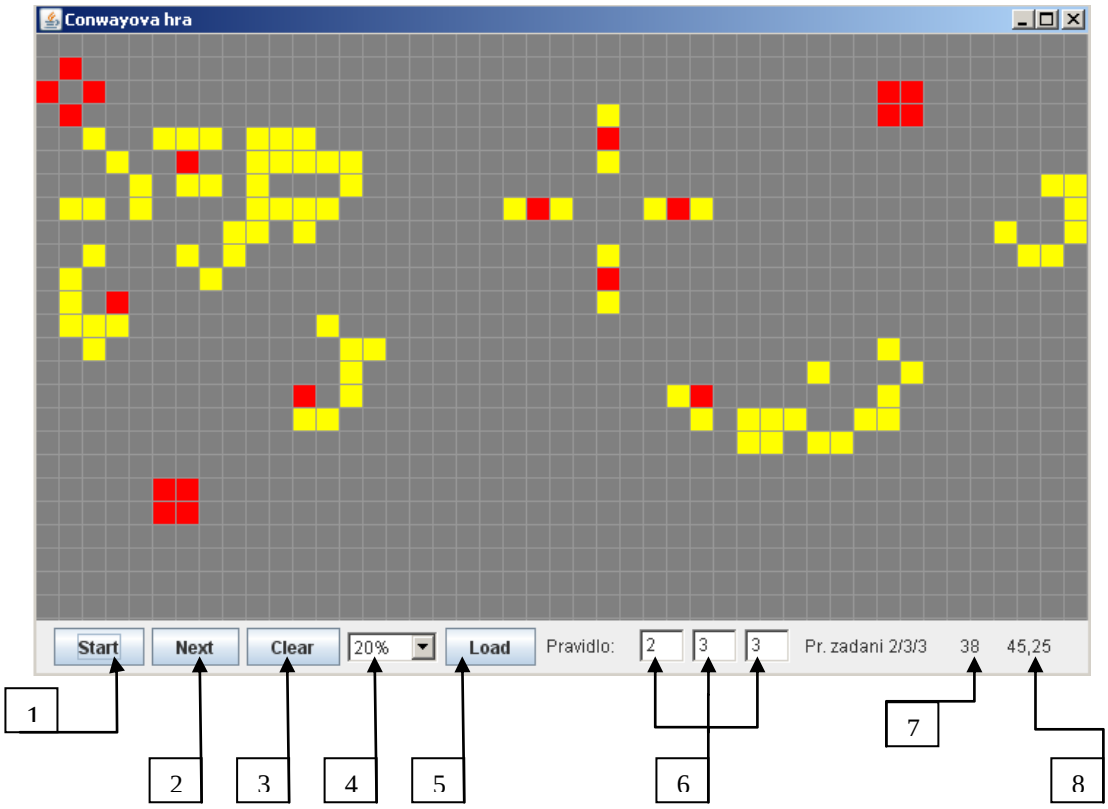
Dostáváme se k metodě Start(), která obsahuje kontrolu okolí. Zde jsou aplikována samotná pravidla. Stav zkoumané buňky je při průchodu zaznamenán do atributu liveNew, který slouží jako indikátor stavu buňky pro průběhu generaci. Po průchodu celé mřížky je zavolána výše uvedená grafická metoda.

Zbývající metody svými jmény odpovídají tlačítkům ovládacího panelu, tj. Clear, Load a neposlední řadě komponentě Choice (roletka), která obsluhuje náhodné generování živých buněk mřížky. Generování probíhá s využití instance Netbeans třídy Random. Čísla v počtu závislém na zvolené procentuální hustotě ukládají do dynamicky alokovaných jednorozměrných polí ze kterých jsou následně vybrány buňky, které změní svůj status. Ukládání jinak náhodně získaných hodnot dvojce souřadnic je výhodné i dále, využití lze nalézt při volání metody Load(), stejnojmenným tlačítkem, kdy je mřížka smazána a její stav navrácen na poslední vygenerovanou hustotu rozložení náhodných živých buněk (elementů).

Nejvýše postavenou třídou resp. instancí této třídy chcete-li je třída Interface.java, která již obsahuje instance tříd výše zmíněných. Hlavní funkcí této třídy je vytvoření grafického uživatelského prostředí (GUI), včetně inicializace použitých komponent a nastavení „posluchačů“ (listeners), každé z komponent. Zde je taktéž umístěna podtřída rozšiřující systémovou třídu Thread (vlákno), potřebnou pro cyklický běh programu.

Mimo zmíněné třídy a jejich metody, aplikace obsahuje také rozhraní pro „naslouchání“ uživatelských vstupů (listeners), nutných pro zajištění funkčnosti ovládacího panelu.

5.2 GUI – Uživatelské prostředí



Obr9. GUI aplikace a jeho popis.

V tomto okamžiku jsme již seznámeni s třídami vytvořené aplikace. Před samotným návodem k použití je třeba v rámci elektronické dokumentace doplnit popis uživatelského prostředí (GUI), se kterým se při užívání aplikace setkáme.

Jednotlivé položky panelu jsou očíslovaný pro lepší přehlednost na Obr9. Postupně je projdeme.:

1. *Start:* Volání vlákna pro cyklický chod programu. Využívá *se i pro ukončení běhu* programu!

2. *Next:* Postup o jednu generaci dopředu. Toto tlačítko má využití také při volbě „hustoty života“ – bod 4, popř. bod 5 – *pokud nedojde* okamžitě po volbě *k aktualizaci mřížky.*

3. *Clear:* Resetuje mřížku a běhové atributy.

4. *Výběr „hustoty života“:* Slouží pro vygenerování náhodně určených pozic a množství živých buněk v procentuálním poměru k velikosti mřížky.

5. *Load:* Po resetu mřížky vyvolá zpět poslední konfiguraci z bodu 4.

6. *Pravidlo(Rule):* Každé z textových polí obstarává jednu sekvenci pravidla CA. Př. Základní pravidlo Conwayovy hry je 2/3/3 resp. 23/3. Spojitost je tedy snadná. *Pro změnu* pravidel je nezbytné změněnou *hodnotu potvrdit stiskem Enter!* (požadavek textových polí).

7. *Generace:* Indikátor pořadí daného generace.

8. *Rozměry:* Průběžně zobrazovaná velikost mřížky v podobě *sloupce, řádky*. Je aktualizována při každé změně velikosti. Defaultní rozměry, které jsou patrné na Obr9, budou použiti při zkoumání vzniku stabilních struktur.

5.3 Upozornění pro obsluhu

Pro správnou funkci programu jsem se rozhodl přidat k dokumentaci upozornění pro správnou obsluhu aplikace. V zásadě se jedná o několik doporučení, jejichž část již byla zmíněna při popisu uživatelského prostředí. Důrazně doporučuji dodržovat tyto doporučení, pokud budete při obsluze postupovat v rozporu s uvedenými upozorněními, nemohu ručit za správnou a bezproblémovou funkčnost programu!

- tl. *Next*, jak již bylo uvedeno, slouží k postupu po generaci a případné aktualizaci mřížky (př. opakovaně změníte procento pokrytí v bodu 4,tj. výběr hustoty „života“, nebo tlačítko Load a nic se nezobrazí, potom *jedním stiskem tlačítka Next aktualizujete mřížku*.

- Vysunovací lišta *Výběr* při novém výběru dokáže pouze přidávat buňky, proto při několika-násobné změně ve Vámi zvolené hodnotě prosím nejprve stiskněte tlačítko *Clear*, nebo položku lišty *„Empty“* pro promazání mřížky. Teprve poté zvolte novou hustotu pokrytí dle Vašeho přání!

- Změnu pravidel potvrďte stiskem Enter v daném poli.

- Při potřebě jiných rozměrů mřížky prosím dbejte na minimální počet sloupců (+-45), nutných pro správné zobrazení panelu ovládání. Samotný „Resize“ probíhá za pomoci myši klasickým roztažením okna. Je třeba dávat pozor na pokud možno rozšíření o celé sloupce nebo řádky, v jiném případě je bude program ignorovat. Tato vlastnost souvisí s výpočtem rozměrů a byla rozebraná dříve. Zda rozměry odpovídají požadovaným, se můžete přesvědčit pomocí indikátoru na pravé straně ovládacího panelu.

Mřížka je pevně zadána jako ohraničená plocha, okraje jsou tedy brány programem jako konec vymezeného prostoru, tj. okrajové buňky mají méně sousedů. Pokud dojde k přepnutí okna programu do neaktivního stavu („hození na lištu, aktivizace jiného okna Windows, ..), zadané informace budou ztraceny.

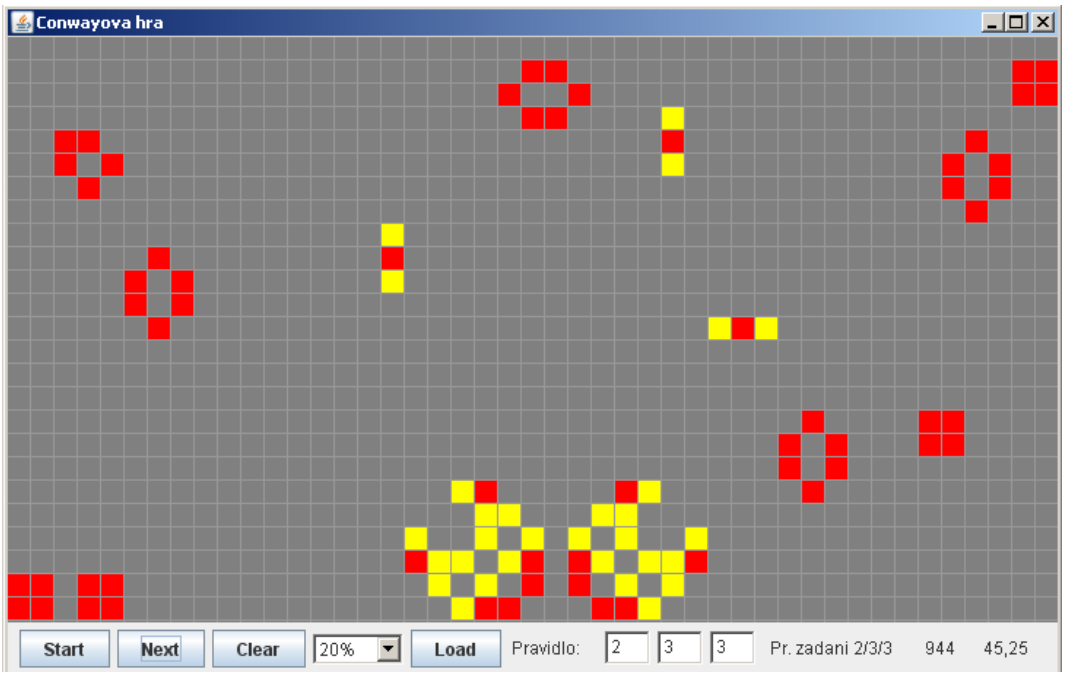
6. Testování okolností vzniku stabilních struktur

Pro test pravděpodobnosti stabilních struktur a následné vyhodnocení určených parametrů byla použita 2D mřížka defaultních rozměrů 45/25, náhodně generovaná hustota živých buněk v mřížce o hodnotách 5, 10, 20,35 %.

Účelem měření bylo zjistit generaci, při které vznikne první stabilní struktura v jinak nestabilní mřížce a také kolik stabilních struktur zůstane v mřížce po nekonečný počet generací, tj. v případě, že mřížka je kompletně stabilní.

Každé pokrytí (5, 10, 20, 35%) bylo proměřeno stokrát a ze získaných hodnot vypočten aritmetický průměr, směrodatná odchylka a medián daného výběru. Všechna data a několik ukázek měření je připojeno níže.

Vzhledem k nestálosti mřížky, při hledání výskytu první stálé struktury, byla tato hledána pozorovatelem. Po jistém ustálení mřížky, tj. asi 7 generací beze změny v okolí mění struktura uznaná jako stabilní barvu na červenou. Tento přístup plně pokrývá nepohyblivé objekty, pohyblivé detekuje pouze částečně, což ovšem stačí k rozpoznání i méně znalému uživateli. Dobře je tato skutečnost vidět na obr10.



Obr10. Blinkery a ½ pulsaru jsou detekovány částečně, ale účinně.

Soupis pozorovaných hodnot a propočtených charakteristik nerozlišuje, zda se jedná o dynamickou nebo statickou strukturu, i když jejich využití se liší (např. „glinder“ pro přenos informací). Veškeré informace jsou uvedeny níže.

5%											
p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞
1	0	0	51	0	0	26	0	0	76	4	2
2	0	0	52	0	0	27	7	1	77	0	0
3	2	1	53	1	1	28	1	1	78	2	5
4	0	0	54	0	0	29	0	0	79	0	0
5	2	6	55	0	0	30	2	2	80	2	2
6	6	1	56	5	1	31	0	0	81	2	1
7	0	0	57	2	1	32	2	1	82	24	1
8	0	0	58	0	0	33	2	1	83	5	1
9	0	0	59	6	1	34	2	1	84	0	0
10	6	2	60	0	0	35	2	1	85	0	0
11	0	0	61	0	0	36	3	1	86	11	4
12	0	0	62	0	0	37	0	0	87	2	1
13	3	2	63	2	1	38	0	0	88	2	2
14	0	0	64	2	1	39	0	0	89	0	0
15	2	1	65	7	1	40	0	0	90	11	4
16	3	1	66	3	1	41	2	1	91	1	1
17	0	0	67	2	1	42	0	0	92	2	1
18	3	1	68	2	2	43	2	1	93	0	0
19	4	2	69	1	1	44	2	1	94	3	1
20	0	0	70	0	0	45	3	2	95	2	1
21	2	1	71	0	0	46	2	6	96	0	0
22	11	4	72	2	2	47	1	1	97	2	1
23	3	1	73	3	1	48	0	0	98	0	0
24	4	2	74	2	1	49	2	2	99	3	2
25	4	2	75	7	1	50	56	2	100	0	0

Σ	2,7	1,0
σ	6,3	1,2
Me	2,0	1,0

Tab1, 2. První sloupec reprezentuje 1. SS, druhý SS ∞ (mřížka beze změn)

10%											
p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞
1	4	2	51	2	2	26	2	10	76	2	3
2	6	11	52	2	2	27	2	5	77	3	12
3	2	3	53	3	2	28	3	5	78	4	4
4	2	3	54	5	2	29	3	7	79	3	4
5	2	3	55	23	4	30	3	8	80	5	9
6	3	4	56	2	9	31	4	3	81	2	2
7	7	2	57	2	6	32	4	2	82	2	4
8	3	3	58	3	8	33	3	7	83	2	10
9	7	3	59	5	12	34	2	2	84	2	3
10	2	9	60	4	6	35	3	10	85	3	14
11	3	2	61	2	4	36	3	5	86	4	3
12	4	3	62	2	7	37	6	2	87	4	7
13	3	6	63	2	5	38	2	7	88	1	6
14	3	4	64	2	6	39	3	5	89	2	6
15	3	3	65	2	2	40	2	2	90	1	6
16	2	4	66	3	3	41	2	5	91	2	3
17	7	1	67	5	1	42	2	5	92	4	7
18	1	3	68	2	4	43	3	8	93	1	2
19	9	13	69	1	5	44	2	9	94	5	13
20	1	7	70	5	2	45	2	4	95	2	4
21	2	2	71	3	6	46	3	4	96	1	5
22	3	8	72	2	7	47	4	3	97	2	8
23	4	7	73	2	5	48	5	8	98	2	9
24	2	4	74	75	5	49	2	4	99	3	3
25	16	3	75	4	1	50	2	7	100	2	5

Σ	4,0	5,2
σ	7,7	3,0
Me	3,0	4,5

Tab3, 4. První sloupec reprezentuje 1. SS, druhý SS ∞ (mřížka beze změn)

20%											
p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞
1	10	6	51	3	7	26	3	10	76	2	5
2	4	7	52	2	10	27	4	8	77	1	7
3	3	9	53	1	6	28	2	13	78	2	9
4	1	6	54	1	13	29	1	9	79	2	9
5	2	3	55	2	6	30	2	8	80	2	4
6	3	9	56	4	7	31	1	12	81	3	5
7	3	6	57	3	12	32	2	13	82	1	6
8	3	7	58	3	11	33	1	8	83	2	4
9	6	12	59	3	11	34	4	7	84	1	3
10	1	8	60	1	3	35	5	9	85	1	8
11	12	7	61	3	7	36	3	4	86	2	8
12	2	9	62	2	3	37	2	9	87	1	8
13	3	5	63	1	10	38	2	8	88	2	8
14	2	5	64	2	5	39	6	13	89	2	8
15	4	9	65	2	10	40	4	7	90	1	11
16	1	2	66	2	5	41	1	7	91	3	3
17	2	9	67	1	8	42	2	10	92	3	16
18	2	8	68	1	11	43	4	14	93	2	6
19	1	17	69	1	5	44	2	9	94	4	13
20	2	11	70	4	6	45	3	6	95	1	9
21	1	9	71	2	11	46	4	8	96	1	10
22	3	11	72	1	16	47	2	5	97	1	6
23	1	9	73	5	3	48	8	7	98	1	6
24	3	11	74	1	10	49	2	4	99	2	8
25	1	8	75	2	11	50	2	10	100	3	16

Σ	2,5	8,2
σ	1,8	3,1
Me	2,0	8,0

Tab5, 6. První sloupec reprezentuje 1. SS, druhý SS ∞ (mřížka beze změn)

35%											
p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1. SS	SS ∞	p.	1 SS	SS ∞
1	4	8	51	2	10	26	4	6	76	1	12
2	3	9	52	3	11	27	3	4	77	12	4
3	3	9	53	6	8	28	7	9	78	4	6
4	6	9	54	1	13	29	5	8	79	8	7
5	1	5	55	3	2	30	2	8	80	7	10
6	3	9	56	5	12	31	4	11	81	3	9
7	4	12	57	1	8	32	7	12	82	5	8
8	3	6	58	3	8	33	5	7	83	6	10
9	14	10	59	1	10	34	2	4	84	1	8
10	1	10	60	6	6	35	5	6	85	7	6
11	3	6	61	2	10	36	1	7	86	6	7
12	6	10	62	1	12	37	5	6	87	4	13
13	4	10	63	4	7	38	2	10	88	4	12
14	1	7	64	1	8	39	2	8	89	5	6
15	4	10	65	6	3	40	7	7	90	3	7
16	2	8	66	4	8	41	6	7	91	4	8
17	11	8	67	2	7	42	4	10	92	1	9
18	1	11	68	6	11	43	8	12	93	3	14
19	2	10	69	16	9	44	8	12	94	5	7
20	10	8	70	7	5	45	5	9	95	4	8
21	3	20	71	2	11	46	2	11	96	4	7
22	6	6	72	5	12	47	5	7	97	5	6
23	3	8	73	5	8	48	5	5	98	2	10
24	8	7	74	1	8	49	7	13	99	9	11
25	12	14	75	5	13	50	1	8	100	5	12

Σ	4,5	8,7
σ	2,9	2,7
Me	4,0	8,0

Tab7, 8. První sloupec reprezentuje 1. SS, druhý SS ∞ (mřížka beze změn)

Z daného rozboru charakteristik vyplývá, že k výskytu první struktury, která je ovšem v zápětí obvykle přepsána druhou až pátou generací v závislosti na zvolené hustotě a náhodném rozložení. Při nižších koncentracích dochází k velké směrodatné odchylce, lepé je tedy dle mediánu vyhodnotit jako vznik první stabilní struktury vůbec druhou až čtvrtou generaci.

Hodnoty SS ∞, již ukazují zvyšující se potenciál v závislosti na předchozí hodnotě. Je ovšem třeba si uvědomit, že zároveň dochází k rostoucímu počtu generací potřebných k úplné stabilizaci mřížky. Ze zkušeností s měřením vychází přibližně tyto rozmezí: (generace potřebné ke stabilizaci nebyly zaznamenány, proto se jedná o hrubý odhad na základě nedávného měření)

- hustota 5% - několik desítek až 100 generací
- hustota 10% - 100 až 300 generací
- hustota 20% - obvykle 200 až 600 generací, ale může přesáhnout i 1000
- hustota 35% - obvykle 800 až 1500 nebo i 2000 generací

Skrze fakt, že daný počet generací nebyl zaznamenáván a je tedy pouze orientační, toto zvýšení logicky souvisí s vyšší hustotou zaplnění mřížky.

7. Závěr

Rešeršní část práce se zabývala především rozsáhlejším teoretickým základem pro pozdější vytvoření aplikace „Game of life“ v programovacím prostředí Java.

Vytvořená aplikace byla rozšířena o flexibilní možnosti volby (hustota pokrytí, pravidla, změna rozměrů, postup o jednu generaci) a nastavena do defaultních hodnot vhodných pro pozorování tvorby stabilních struktur.

Z naměřených hodnot po statistickém rozboru vychází, že nehledě na hustotu pokrytí mřížky živými buňkami dochází ke vzniku první stabilní struktury v druhé až čtvrté generaci. Pro nekonečný počet generací, tj. mřížka je stabilizována, dochází k závislosti počtu stabilních struktur na úvodní hustotě mřížky. Zároveň, je třeba si uvědomit, že dochází k rostoucímu počtu generací potřebných k úplné stabilizaci mřížky. Z měření lze vytvořit hrubý odhad generací potřebných pro danou hustotu pokrytí.

- hustota 5% - několik desítek až 100 generací
- hustota 10% - 100 až 300 generací
- hustota 20% - obvykle 200 až 600 generací, ale může přesáhnout i 1000
- hustota 35% - obvykle 800 až 1500 nebo i 2000 generací

Skrze fakt, že daný počet generací nebyl zaznamenáván a je tedy pouze orientační, toto zvýšení logicky souvisí s vyšší hustotou zaplnění mřížky.

Použitá literatura

[1] ----.*Cellular automaton - Wikipedia, the free encyclopedia*.[online]. 2010-4-19,[cit.2010-05-04]. Dostupné z http://en.wikipedia.org/wiki/Cellular_automaton

[2] Weisstein, Eric W. .*Cellular Automaton -- from Wolfram MathWorld*.[online]. 2002-07-23,2006-03-28,[cit.2010-05-04]. Dostupné z http://mathworld.wolfram.com/CellularAutomaton.html

[3] ----.*Cellular automaton - Wikipedia, the free encyclopedia*.[online]. 2010-4-19,[cit.2010-05-04]. Dostupné z http://cs.wikipedia.org/wiki/Celul%C3%A1rn%C3%AD_automat#Codd.C5.AFv_automat

[4] Wiesstein, Eric W. . *Elementary Cellular Automaton -- from Wolfram MathWorld*[online].2002-04-09,2006-04-12,[cit.2010-05-05]. Dostupné z http://mathworld.wolfram.com/ElementaryCellularAutomaton.html

[5] Wiesstein, Eric W. . *Totalistic Cellular Automaton -- from Wolfram MathWorld:* [online]. 2002-04-11,2003-01-21,[cit. 2010-05-05]. Dostupné z http://mathworld.wolfram.com/TotalisticCellularAutomaton.html

[6] Wiesstein, Eric W. . *Life -- from Wolfram MathWorld:* [online]. 2002-11-05,2003-01-21,[cit. 2010-05-05]. Dostupné z http://mathworld.wolfram.com/Life.html

[7] ----,*Conway's Game of Life - Wikipedia, the free encyclopedia*.[online].2010-05-05,[cit. 2010-05-06]. Dostupné z http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life#Examples_of_patterns

[8] ----, *Still life (cellular automaton) - Wikipedia, the free encyclopedia*[online].2010-04-21[cit. 2010-05-06]. Dostupné z http://en.wikipedia.org/wiki/Still_life_%28CA%29

[9] ----,*Oscillator (cellular automaton) - Wikipedia, the free encyclopedia*[online].2010-01-01,[cit. 2010-05-10]. Dostupné z http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

[10]----,*Spaceship (cellular automaton) - Wikipedia, the free encyclopedia*[online].2009-12-25[cit. 2010-05-10]. Dostupné z http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

[11] A NEW KIND OF SCIENCE, Stephen; Wolfram. Champaign, IL, Wolfram Media, 2002-04-14,1197p., 1-57955-008-8

[12] ----, *[Page 375] Stephen Wolfram: A New Kind of Science | Online*[online],[cit. 2010-05-11]. Dostupné z http://www.wolframscience.com/nksonline/page-374

[13] ----,*[Page 427] Stephen Wolfram: A New Kind of Science | Online*[online],[cit. 2010-05-11]. Dostupné z http://www.wolframscience.com/nksonline/page-427