Experimenty s celulárnymi reprezentáciami v evolučnom designe

Jana Hlavačiková, Pavel Petrovič

Katedra aplikovanej informatiky, FMFI UK
Mlynská dolina, 842 48 Bratislava
janahlava@gmail.com, ppetrovic@acm.org

Abstrakt

Evolučný design sa zaoberá automatickým návrhom morfologickej štruktúry objektov prostriedkami evolučných výpočtov. Pri evolučnom návrhu je kľúčovým problémom to, ako zvoliť reprezentáciu príslušného evolvovaného tvaru. Do úvahy prichádzajú dve základné skupiny reprezentácií: priama a nepriama. V priamej reprezentácií je tvar predmetu priamo zakódovaný v genotype ("blueprint"), zatiaľ čo v nepriamej reprezentácii je genotypická reprezentácia jedinca odlišná od jeho fenotypu, ktorý sa konštruuje na základe iteratívneho algoritmu, počas ktorého vzniká výsledná morfologická štruktúra tvaru. Celkom odlišný je však prístup celulárnych reprezentácií, ktoré sú založené na samoorganizácii buniek v mriežke stavového automatu. Predmetom evolúcie je teda prechodová funkcia stavového automatu, ktorú môže tvoriť napríklad umelá neurónová sieť s dopredným šírením.
Jeden z vhodných evolučných algoritmov, ktoré poskytuje dobré výsledky pri evolúcii takých sietí je CMAES [1]. V článku experimentujeme s evolúciou funkcií pre celulárne automaty, ktoré svojou samoorganizáciou vytvárajú dvojrozmerné vzory podľa predlohy. Skúmame vplyv počiatočných podmienok na výsledný tvar. Experimentujeme s celulárnymi automatmi s nepravidelnou mriežkou.

1 Úvod

Na rozdiel od rôznych prístupov a metód zavedených v minulosti je dnes všeobecne akceptované poznanie, že *distribuovanosť* informácií, procesov a javov je kľúčová pre všetko živé alebo inteligentné. Môžeme ju pozorovať napríklad vo vývoji poznania ľudstva, kde sú poznatky distribuované naprieč ľudskými populáciami, vo vývoji genotypov druhov organizmov, kde prirodzená evolúcia prehľadáva priestor prípustných genetických reťazcov široko distribuovanou druhovou diverzitou, či v organizácii informácie v centrálnych nervových sústavách vyšších organizmov, kde sú údaje distribuované v sieťach prepájajúcich milióny neurónov. Distribuovanosť je zároveň základnou vlastnosťou najvyspelejších a najobsiahlejších umelých systémov. Spočíva a) v duplikovaní a teda redundancii informácie a vďaka tomu omnoho väčšej robustnosti, b) priestorovej dislokácii, ktorá umožňuje vytvárať účinnejšie grafové štruktúry vyjadrujúce vzájomné závislosti jednotlivých aspektov, c) vo vysokom stupni paralelizmu spracovania, vďaka ktorému celok môže pozostávať z pomerne jednoduchých elementov a napriek tomu dosahovať obrovskú výpočtovú alebo reprezentačnú silu a d) lokálnej autonómnosti elementov, ktorá dovoľuje vysoký stupeň adaptívnosti a flexibilnosti štruktúry podľa aktuálnej potreby.

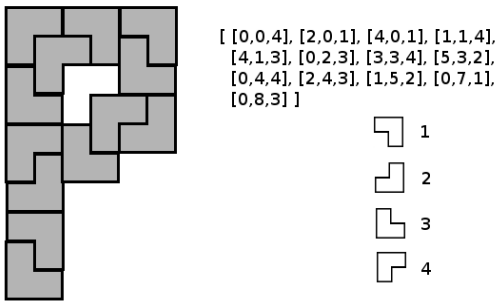
1.1 Distribuovanosť v evolučných algoritmoch

Typickým príkladom osvojenia myšlienky distribuovanosti v informatike sú evolučné algoritmy, ktoré reprezentujú aktuálne napredovanie počas prehľadávania stavového priestoru pomocou distribuovanej populácie riešení. Aktuálna populácia genetického algoritmu obsahuje množstvo potenciálnych čiastočných riešení, pričom každé z nich sa v ďalších iteráciách/generáciách môže stať zárodkom pre hľadané cieľové riešenie. Napriek tomu tieto algoritmy reprezentujú hľadané riešenie spravidla ako monolitický celok – bitový alebo číselný reťazec, strom, graf, či lineárny program. Na druhej strane, aj v takýchto štruktúrach môže dochádzať k určitému stupňu distribuovanosti, keď genotyp obsahuje časti, ktoré nie sú v riešení aktívne. Tieto časti kódu slúžia ako genetická pamäť z predchádzajúcich generácií, časti kódu, ktoré môžu byť opätovne použité operátormi kríženia v neskorších generáciách. Vo väčšine prípadov však ide o zhustenú a centralizovanú (nedistribuovanú) informáciu zaznamenanú v genotype, kde sa jednotlivé časti priamo mapujú na časti výsledného riešenia – fenotypu. Príklad priamej reprezentácie je na obrázku 1.

1.2 Nepriame reprezentácie

Zväčšenie odstupu medzi genotypom a fenotypom dosahujú nepriame reprezentácie, kde sú jednotlivé črty výsledného fenotypu určené iteratívnym procesom z genotypu na základe postupnosti transformácií. Hlavnou motiváciou pre zavedenie nepriamych reprezentácií je kompresia údajov: kompaktná komprimovaná reprezentácia môže určovať širokú

rôznorodosť formy výsledných fenotypov, podrobne sa zamerať na tie časti fenotypu, ktoré treba v cieľovom riešení špecifikovať s veľkým stupňom detailu a naopak, jednoliate uniformné časti fenotypu vyjadriť malým množstvom reprezentačných primitív. Podobne, operátory rekombinácie a mutácie na genotype s nepriamou reprezentáciou môžu ovplyvňovať celú sústavu analogických aspektov výsledného riešenia v jedinom kroku. Možnosť vhodného návrhu reprezentácie a operátorov je však pri nepriamych reprezentáciách veľmi závislá od konkrétnej domény, stavia prevažne na subjektívnej skúsenosti a zväčša vyžaduje empirické overenie.

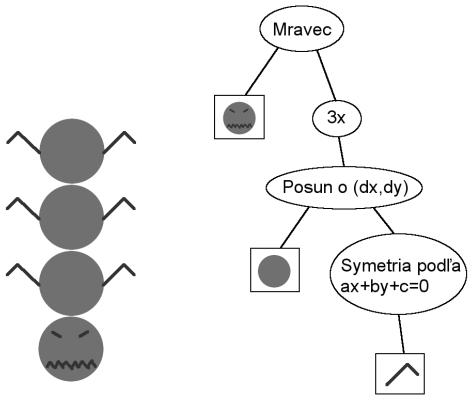


Obr. 1. Príklad priamej reprezentácie: tvar písmena "P" vytvorený z ľubovoľne natočených útvarov v tvare L – jednotlivé útvary sú umiestnené na súradniciach, ktoré sú priamo uložené v genotype spolu s ich orientáciou.

Vzhľadom na iteratívny proces vytvárania fenotypu môžu nepriame reprezentácie dosahovať istý stupeň distribuovanosti. Konkrétny aspekt výsledného tvaru môžu ovplyvňovať mnohé aspekty genotypu. Na druhej strane však takto nie je pokrytá vlastnosť redundancie, ktorá je tiež nevyhnutnou súčasťou distribuovanosti. Príklad nepriamej reprezentácie je na obrázku 2. Výsledný tvar objektu sa vytvára z elementárnych prvkov postupnosťou zhodných zobrazení zoradených v hierarchickom strome: tranformácia v uzle ovplyvňuje všetky listy vo svojom podstrome.

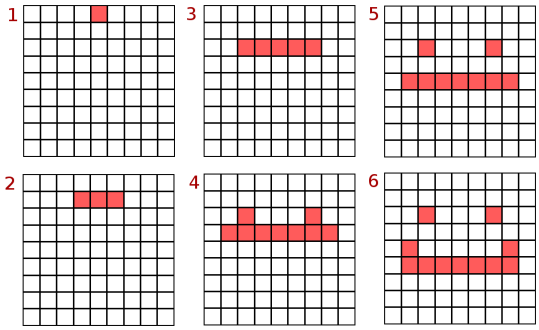
1.3 Celulárne reprezentácie

Kvalitatívne odlišným druhom sú celulárne reprezentácie, ktoré sú založené na celulárnych automatoch, čiže dvojrozmernej mriežke buniek, kde každá z nich vykonáva rovnakú jednoduchú činnosť v každom kroku iterácie automatu, čiže aktualizáciu svojho stavu. Okrem toho, že si bunky automatu udržujú svoj vnútorný stav, komunikujú s bunkami vo svojom bezprostrednom okolí. Výsledné riešenie – fenotyp je produktom výpočtu automatu po jeho skonvergovaní (stav, v ktorom sa vnútorný stav buniek naďalej nemení), alebo po dosiahnutí požadovaných parametrov (hoci konvergencia je hodnotnejší výsledok vzhľadom na jeho väčšiu robustnosť). Celulárne reprezentácie teda zdieľajú vlastnosti priamych i nepriamych reprezentácií: v stave konvergencie priamo zodpovedajú fenotypu a zároveň prechádzajú iteratívnym vývinom.



Obr. 2. Príklad nepriamej reprezentácie: uzly obsahujú transformačné operátory, makrooperátory, alebo ich kombinácie. Aplikovaním operátorov na všetky listy podstromu a kompozíciou získavame výsledný fenotyp.

Genotyp v tomto prípade spravidla reprezentuje predpis pre činnosť buniek automatu – môže to byť pravidlový systém, konečno-stavový automat, stromový program, alebo dopredná neurónová sieť. Celulárne reprezentácie dosahujú vysoký stupeň distribuovanosti: priestorová dislokácia vyplýva z rozmiestenia buniek v mriežke, vysoká redundancia z toho, že každá bunka sa riadi rovnakým predpisom (prechodovou funkciou) a vysoký stupeň paralelizmu zo spôsobu výpočtu – v každej bunke automatu prebieha výpočet nezávisle od ostatných. Tieto vlastnosti napovedajú, že riešenia reprezentované celulárnymi automatmi by mali byť robustné. Praktické experimenty túto teóriu potvrdzujú – po pridaní šumu k stavom buniek automat konverguje k rovnakému výsledku. Obr. 3 zobrazuje príklad celulárnej reprezentácie dvojrozmerného obrázku.



Obr. 3. Príklad cellulárnej reprezentácie. Výsledný tvar vznikne po 6 iteráciách interakcií buniek v mriežke.

2 Evolučný design

Evolučné výpočty boli v minulosti úspešne použité na optimalizáciu morfológie dvoj- alebo troj-rozmerných objektov [2]. Účelová funkcia spravidla testuje jedno alebo viacero optimalizačných kritérií, ktorým má výsledný tvar vyhovovať. Algoritmus prehľadáva rozsiahly priestor možných riešení bez toho, aby bol zaťažený (biased) tradičnými inžinierskymi postupmi a tak môže viesť k inovatívnym riešeniam, ktoré prekonávajú (outperform) konvenčné riešenia.

Evolučný design má veľký potenciál a význam pre aplikácie umelého života, kde hľadanie inteligentného jedinca zahŕňa nielen jeho správanie ale aj jeho morfológiu. Telo agenta hrá kľúčovú úlohu pri získavaní vnemov a informácií z prostredia (embodiness).

Okrem voľby druhu optimalizačného algoritmu je dôležitá voľba vhodnej reprezentácie jednotlivých tvarov a operátorov, ktoré definujú lokálne okolie v prehľadávanom priestore. Najčastejšie využívané sú priame reprezentácie, v ktorých sa prehľadávací priestor najjednoduchšie definuje a analyzuje. Nepriame reprezentácie sľubujú viacero výhod:

- majú premenlivú rozlišovaciu schopnosť, ktorú prispôsobujú podľa potreby v jednotlivých častiach evolvovaného tvaru, čiže
- dokážu sa zamerať na vybrané časti tvaru
- umožňujú kompresiu veľkého množstva dát do kratšej reprezentácie, veľkosť prehľadávacieho priestoru sa podľa potreby dynamicky mení
- dokážu vyžiť vnútornú štruktúru tvaru: symetrie, opakovanie, modularitu
- majú lepší potenciál uniknúť z lokálneho optima v prípade predčasnej konvergencie, keďže operátory dokážu spôsobiť malú i veľkú zmenu vo výsledku
- výsledky môžu mať lepšie estetické vlastnosti (vyplýva to z využitia vnútornej štrúktúry)
- výsledky sa viac podobajú na analytick popisy riešenia a preto môžu byť jednoduchšie analyzovateľné

Dosiahnutie uvedených výhod v praxi však často ostáva mimo dosah konkrétnej implementácie, voľba vhodných evolučných operátorov je ťažký problém. V prácach, ktoré skúmajú vlastnosti jednotlivých reprezentácii (napr. [2]) niekedy reálne kritériá účelovej funkcie nahrádzame umelými. Napr. ak chceme prevereť vhodnosť kombinácie zvolenej reprezentácie a evolučného algoritmu a ich schopnosť konvergovať k tvaru s požadovanými reálnymi vlastnosťami, môžeme namiesto toho overiť ako je pre túto kombináciu náročné konvergovať k nejakému fiktívnemu zvolenému a presne stanovenému výslednému tvaru. Dá sa predpokladať, že tie reprezentácie, ktoré ľahšie konvergujú k rozličným tvarom, podobným pre typické riešenia v danej doméne, majú väčšiu reprezentačnú silu a budú vhodnejšie aj pri použití reálnych optimalizačných kritérií.

Doposiaľ najmenej využívaným a preskúmaným druhom sú celulárne reprezentácie. V ich prípade je výsledný tvar priamo „nakreslený“ v posledných iteráciach celulárneho automatu. Doterajšie práce sa zameriavali predovšetkým na schopnosť automatu vytvoriť výsledný predpísaný dvojrozmerný tvar [3]. V tomto príspevku nadväzujeme na uvedené štúdie a pokúšame sa o ďalší prieskum možností a obmedzení celulárnych automatov.

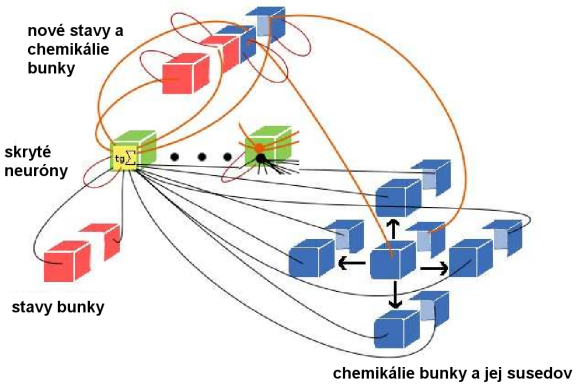
3 Problém vlajky

Úloha je inšpirovaná známou informatickou úlohou “Dutch National Flag Problem” navrhnutou E.Dijkstrom. Holandská vlajka pozostáva z trikolóry červená – biela – modrá, pričom úlohou je zoradiť náhodne rozhádzané guličky troch farieb v lineárnom rade tak, aby tvorili vzor vlajky, pričom cena presunutia guličky je tak vysoká, že každá gulička môže byť presunutá iba raz. K danej úlohe existuje triviálne riešenie, ktoré je užitočnou etudou pri výuke programovania.

V našom prípade, vychádzajúcom z práce [3], tvorí cieľová vlajka dvojrozmerný vzor, ktorý má byť výsledným produktom iterujúceho celulárneho automatu: každému bodu vo výslednom vzore zodpovedá jedna bunka automatu. Hľadáme takú prechodovú funkciu pre bunky, ktorá z počiatočnej konfigurácie (nulové hodnoty) postupne vytvorí stabilný požadovaný vzor. Ten by mal byť atraktorom, čiže po lokálnych perturbáciách (pridanie Gaussovského šumu) automat opäť zaujme rovnaký vzor. Množstvo šumu vzhľadom na ktoré je automat robustný určuje kvalitu výsledku.

Vzhľadom na jeho dobré adaptívne vlastnosti je ako prechodová funkcia buniek použitý viacvrstvý perceptrón, pričom každá bunka pozostáva z dvoch druhov stavových premenných: stavov a chemikálií. Chemikálie sú inšpirované reakčno-difúznym modelom, odkaz v [3], v ktorom stav systému závisí od chemikálií, difundujúcich v prostredí a vstupujúcich do lokálnych reakcií. Stavy automatu sú vnútorné premenné, reagujú na chemickú komunikáciu s okolitými 4 bunkami. Chemikálie tvoria vonkajšie prostredie, a okrem komunikácie s okolitými bunkami sa aj difúzne šíria v prostredí – v našom prípade operátorom Gaussian Blur v dvoch prechodoch vertikálne a horizontálne. Difúzny krok sa strieda s reakčným. Stavy aj chemikálie bunky sú aktualizované podľa výstupu neurónovej siete, do ktorej vstupujú predchádzajúce stavy bunky a chemikálie bunky a okolitých buniek, obr. 4.

Počet stavov, chemikálií a skrytých neurónov sú parametre systému a ich správne hodnoty závisia od zložitosti evolvovaného vzoru.



Obr. 4. Aktualizácia stavov a chemikálií bunky.

Výslednú farbu bunky tvorí nelineárna váhovaná kombinácia stavov a chemikálií bunky. Ak c_i sú chemikálie, s_i sú stavy, farba bunky sa určí podľa vzťahu:

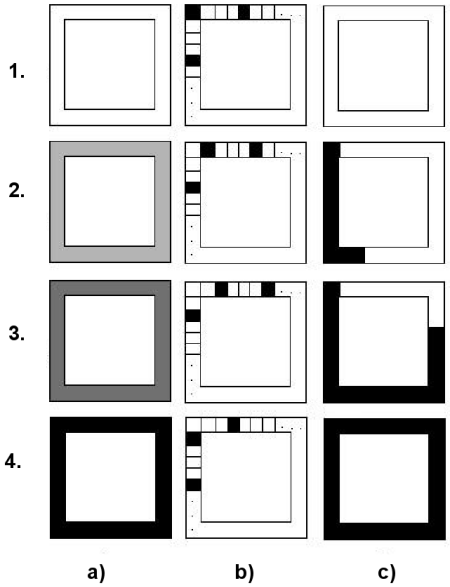
$$color_i = \frac{1 + \tan\left(w_{c_i} c_i + w_{s_i} s_i\right)}{2}$$

Činnosť automatu teda závisí od váh neurónovej siete a váh vo funkcii na výpočet farby a práve tie sú predmetom prehľadávania a optimalizovania. Automat sa zastaví, keď štandardná odchýlka jeho „energie“ klesne pod prahovú hodnotu počas W (W=8) iterácií, alebo ubehne maximálny počet iterácií (1024). Energia je stanovená ako súčet druhých mocnín všetkých stavových premenných všetkých buniek automatu. Počet iterácií sa tiež zohľadňuje v kvalite riešenia a uprednostnené sú tie riešenia, ktoré konvergujú po nižšom počte iterácií.

Podobná aktualizácia a reprezentácia bola použitá v práci [4].Miller v roku 1998 zaviedol metódu Cartiesian Genetic Programming. V jeho práci sa program v bunke rozhodne aké množstvo chemikálie vyprodukuje, či prežije, alebo sa zmení na iný typ bunky a ako porastie. V prípade rastu do inej bunky úplne prepisuje jej vlastnosti. Bunky, ktoré neprežijú prestávajú interagovať.

3.1 Riadenie vývinu nastavením okrajov

Okraje automatu boli v predchádzajúcej práci inicializované na nulové hodnoty. Naším cieľom bolo preskúmať, či je možné nájsť automat s dvomi alebo viacerými rôznymi atraktormi, ktoré by sa volili len nastavením pevných stavových hodnôt po okrajoch automatu. Obr. 5 zobrazuje viacero rôznych spôsobov nastavenia okrajov automatu, vľavo sú všetky okrajové bunky inicializované na hodnotu proporcionálnu poradiu cieľového vzoru, v strede je plne aktivovaná každá N-tá okrajová bunka, pričom okraj je fázovo posunutý podľa poradia vzoru, vpravo je plne aktivovaná časť okraja, ktorej veľkosť zodpovedá poradiu vzoru, ostatné bunky sú nulové. V prípade dvoch obrázkov sú prvý a tretí spôsob inicializácie ekvivalentné.



Obr. 5. Rôzne nastavenia okrajov automatu.

4 Evolučný algoritmus

Rovnako ako v predchádzajúcej práci [3] využívame na optimalizáciu váh perceptrónu evolučnú stratégiu CMAES (Covariance Matrix Adaptation Evolution Strategy), ktorá spomedzi evolučných stratégií dosahuje výborné výsledky, keďže je založená na matematickom modeli viacrozmerného normálneho rozdelenia pravdepodobnosti určeného priemerom a kovariančnou maticou [1]. Na rozdiel od klasických populačných metód si CMAES neudržuje informáciu o aktuálnej polohe vo forme populácie jedincov, ale v parametrickej forme viacrozmerného náhodného rozdelenia: *n*-rozmerný vektor **m** určuje priemer a štvorcová kovariančná matica **C** typu *n*x*n* určuje škálovanie a natočenie náhodného rozdelenia v priestore. V každej iterácii algoritmus vzorkovaním viacrozmerného náhodného rozdelenia vytvorí λ jedincov (*n*-rozmerných vektorov reálnych čísel), z ktorých vyberie μ najlepších rodičov, ktorí ovplyvnia parametre distribúcie v nasledujúcej generácii, rôznou váhou, podľa poradia ich „fitness“. Distribúcia náhodného rozdelenia sa takto v jednej generácii upraví. Priemer sa posunie do váhovaného priemeru vybraných rodičov a kovariančná matica sa upraví o malý krok tak, aby distribúcia bola „natočená“ v smere k lepším riešeniam, ktoré boli v príslušnej generácii objavené. Okrem toho CMAES berie do úvahy históriu posunov priemeru (tzv. *evolution path*), vďaka ktorej zmena distribúcie môže nabrať zotrvačnosť a prejsť krížom cez prehľadávací priestor s menším počtom ohodnotení jedincov. Táto história ovplyvňuje veľkosť kroku aj zmenu tvaru distribúcie. Podrobnosti CMAES možno nájsť v [1]. V našich experimentoch sme všetky

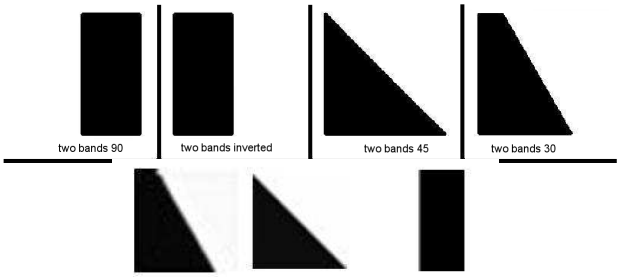
parametre evolučného algoritmu nastavili na štandardné hodnoty. Genotyp tvoril vektor 111 váh – reálnych čísel (107 pre aktualizáciu premenných a 4 pre výpočet farby). Veľkosť populácie λ=18 a počet vybraných rodičovských jedincov μ=9. Váhy rodičovských jedincov boli určené logaritmickou stupnicou: $w_i = log\ \mu + 1 - log\ i + 1$.

Účelová funkcia pre našu úlohu je stanovená ako súčet rozdielov farieb výsledného celulárneho automatu a predloženého vzoru. Súčet je lineárne normalizovaný do intervalu (0;1). Optimalizačný algoritmus hľadá riešenie s minimálnou „fitness".

5 Výsledky

V experimentoch sme použili bunky s jedným stavom, dvoma chemikáliami a 8 skrytými neurónmi. Testovali sme evolúciu relatívne jednoduchých čiernobielych vzorov v mierke 32x32 bodov, farba bodu je v intervale (0;1). Počiatočná inicializácia stavov buniek automatu na iné ako nulové hodnoty v nultej iterácii nepriniesla rozdiely vo výsledkoch.

Základný vzor „*two bands 90*" tvoria biely a čierny obdĺžnik umiestnené vedľa seba, ostatné vzory vznikajú otáčaním základného vzoru o 30 a 45 stupňov a inverziou, obr. 6. Najlepšie výsledky sme získali pre nastavenie okrajov podľa obr. 5c.



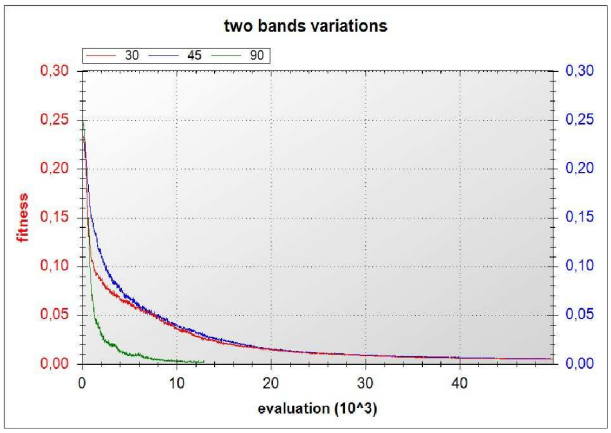
Obr. 6. Cieľové vzory vlajok (hore) a príklady troch výsledných skonvergovaných vzorov (dole).

Účelová funkcia v experimentoch s viacerými vlajkami uvažovala priemernú fitness všetkých vzorov vlajok, alebo započítala najhoršiu fitness, ktorú automat dosiahol na jednotlivých vzoroch, nezaznamenali sme rozdiel. Cieľom je minimalizovať fitness na všetkých vzoroch.

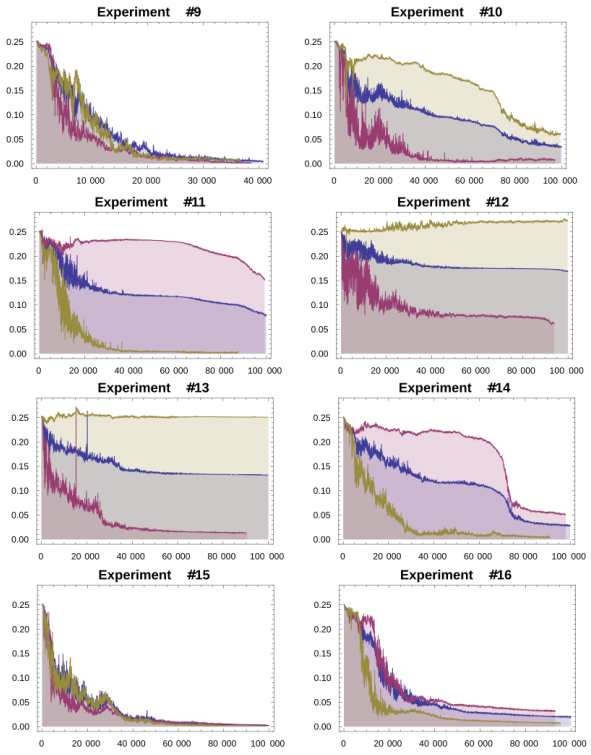
Najskôr sme zopakovali experimenty evolúcie jedného vzoru (s nulovým okrajom) podľa [3]. Obr. 7 zobrazuje priemer najlepšej fitness z 16 behov. Algoritmus našiel riešenia pre všetky vlajky, pričom automat konvergoval k minimálnej zmene energie približne po 400 iteráciách.

V ďalších experimentoch mal automat konvergovať k dvom rôznym vzorom v závislosti od nastavenia okraja. Vo všetkých skúmaných prípadoch sa nám podarilo automat naevolvovať, hoci konvergencia evolúcie nenastala v každom evolučnom behu. Obr. 8 zobrazuje postupnú konvergenciu evolučných behov, pre ostatné dvojice vzorov vyzerali priebehy analogicky.

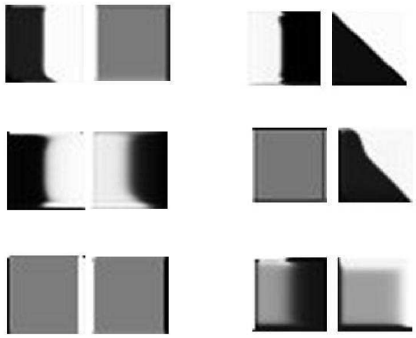
Realizovali sme i pokusy o evolúciu automatov so 4 atraktormi – pre 4 rôzne predlohové vzory a okraje automatu, avšak v týchto behoch sa nepodarilo nájsť automat, ktorý by vytváral čolen jediný z predložených vzorov.



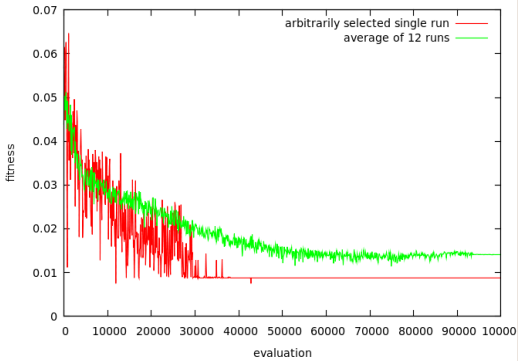
Obr. 7. Priebeh testovacích behov pre 1 cieľový vzor.



Obr. 8. Vývoj fitness najlepšieho jedinca v populácií vzhľadom na počet ohodnotení jedincov. Dve krivky určujú fitness pre požadované vzory (*twobangs90* a *invertovaný*), krivka v strede je priemer – teda celková fitness automatu. Grafy sú z polovice evolučných behov.



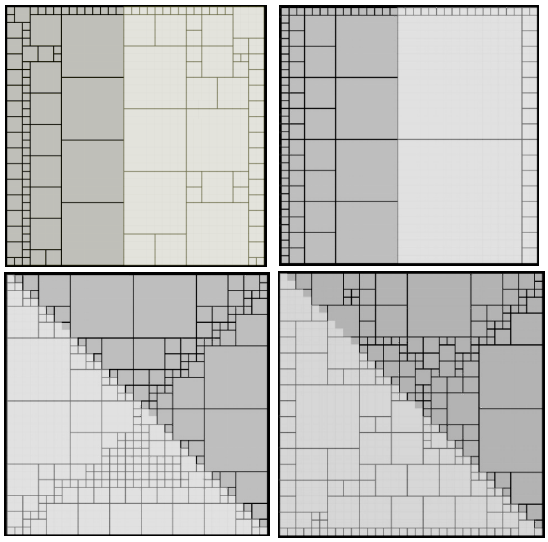
Obr. 9. Príklady naevolvovaných dvojíc atraktorov automatov pre dvojice vzorov *twobands90* a *invertovaný* (vľavo) a *twobands90* a *twobands45* (vpravo).



Obr. 10. Vývoj fitness pre premenlivé mriežky.

6 Experimenty s premenlivou mriežkou

Inšpirovaní elegantnosťou nepriamych reprezentácií sme ďalej skúmali odklon od pôvodnej myšlienky celulárnych automatov: namiesto rovnomernej mriežky sme dovolili bunkám zlučovať sa a rozdelovať sa na základe hodnoty jednej zo stavových premenných: pre hodnoty < -1/3 sa bunka rozdelila (ak bola väčšia, ako základná jednotka), pre hodnoty > 1/3 sa mala záujem zlučovať s ostatnými bunkami. Bunky teda vo svojom okolí mohli mať rôzny počet susedov, pričom príspevok každého suseda z tej-ktorej svetovej strany bol váhovaný dĺžkou spoločnej hranice. Najskôr sme chceli preskúmať, či takýto automat vôbec dokáže efektívne fungovať a prispôsobiť svoju štruktúru tak, aby zodpovedala predloženému vzoru, presnejšie, aby po priradení vzniknutých buniek k súvislým plochám vstupného vzoru ostalo čo najmenšia plocha, ktorá je priradená nesprávne a prečnieva. Okrem tejto zložky brala účelová funkcia do úvahy aj výsledný počet buniek, ktorý sa snažila minimalizovať. V experimente s *twobands90* skonvergovala väčšina evolučných behov, v experimente *twobands45* niektoré. Príklady výsledných vzorov sú na obrázku 11 a priemer najlepšej fitness z 12 evolučných behov i ukážka vybraného behu pre *twobands90* je na obrázku 10.



Obr. 11. Príklady výsledných premenlivých mriežok.

7 Záver

V prvej časti práce sme úspešne replikovali predchádzajúci experiment (ref.). V druhej časti práce sme preskúmali a potvrdili schopnosť celulárnych reprezentácií konvergovať do rôznych predurčených atraktorov na základe rôznych indícií v prostredí a schopnosť evolučnej stratégie CMAES skonštruovať takéto automaty. V záverečnej časti práce sme zaviedli (podľa našich vedomostí) nový druh celulárneho automatu s premenlivou štruktúrou mriežky, ktorá umožňuje zameranie sa na detaily len v potrebných miestach vzoru. V budúcej práci sa budeme zameriavať na využitie a konštrukciu automatov s premenlivou mriežkou, na automaty operujúce v trojrozmernom priestore a ďalšie aplikácie celulárnych reprezentácií v evolučnom designe.

Literatúra

[1] N. Hansen, *CMA Evolution Strategy: A Tutorial*, version March 7 2010 [online] prístupné na: `http://www.lri.fr/~hansen/cmaesintro.html`.

[2] J. Plavčan, *Reprezentácie v evolučnom designe*, diplomová práca, FMFI UK, jún 2007.

[3] A. Devert, *Building processes optimization*: *Toward an artificial ontogeny based approach*, Université Paris-Sud, June 2009, Ph. D. Thesis.

[4] J. F. Miller, Evolving a Self-Repairing, Self-Regulating, French Flag Organism, in *Genetic and Evolutionary Computation – GECCO 2004*, Springer-Verlag, 2004, 129-139.

[5] J. Hlavačiková, Celullar Embryogenic Representations for Evolutionary Design, diplomová práca pred obhajobou, FMFI UK, jún 2010.