EVOLUTIONARY DESIGN USING REWRITING SYSTEMS

Jiří Hýsek
Master Degree Programme, FIT BUT
E-mail: xhysek02@stud.fit.vutbr.cz

Supervised by: Michal Bidlo
E-mail: bidlom@fit.vutbr.cz

ABSTRACT

This work is an introduction to evolutionary algorithms and an evolutionary design. It also describes disadvantages of direct encoding of a genotype to a phenotype and a method of nontrivial encoding which tries to solve these problems. We are particularly talking about nonscalability of evolved solutions, hence our main goal is to propose a technique for the design of arbitrarily large instances of a target problem. The proposed technique is demonstrated on sorting networks. Sorting networks and some conventional ways of their construction are discussed as well.

1 ÚVOD

V technické praxi často vznikají požadavky na návrh konstrukcí, obvodů či systémů různých vlastností nebo velikostí. Ruční návrh může být časově náročný, bude vyžadovat vysoce kvalifikovaného odborníka a pokud mluvíme o rozsáhlých systémech, může být velmi obtížné vytvořit řešení s dostatečně dobrými vlastnostmi, o škálovatelnosti výsledného řešení ani nemluvě. Zde se začínají uplatňovat automatizované techniky založené na evolučních algoritmech. Těmi se tato práce zabývá, jejím cílem je návrh evoluční techniky schopné konstruovat libovolně rozsáhlé struktury, které mohou svými vlastnostmi konkurovat konvenčním postupům.

2 EVOLUČNÍ ALGORITMY

Evoluční algoritmy (EA) [1] reprezentují množinu biologií inspirovaných algoritmů, které mají společný základ – využívají hledání nejlepšího řešení *přirozeným výběrem*. Přirozený výběr je klíčový proces evoluce a zajišťuje, že se prosadí ti nejsilnější. Slabší se buď přizpůsobí nebo vymřou. Další důležitý efekt evoluce je *genetický drift*, který se projevuje tím, že z genotypu mizí neúspěšné geny, vhodné vlastnosti postupně převládají a v konkurenčním boji o přežití se dále zlepšují. Třetím hlavním rysem je *proces reprodukce* – nová generace jedinců vzniká z původní populace, po které dědí část genetického materiálu. EA pracují s množinou kandidátních řešení, každé z nich má přiřazenu hodnotu *fitness*, která určuje kvalitu daného řešení. Dokud není splněna podmínka pro ukončení výpočtu, vytvářejí se další generace řešení. Přirozený výběr a genetický drift zajišťuje, že se v populaci jednotlivá řešení zlepšují. Jedinci s vyšší

fitness mají větší šanci stát se rodiči a vyprodukovat tak více potomků, které mohou zdědit jejich kvalitní genetický materiál. Podmínkou pro ukončení výpočtu bývá nejčastěji nalezení řešení o dostačující kvalitě, nebo dosáhnutí stanoveného počtu generací. EA se s výhodou používají pro řešení problémů, na které nestačí dnešní matematický aparát, výpočetní síla, nebo problém dostatečně neznáme a nedokážeme jej analyticky řešit.

3 EVOLUČNÍ NÁVRH

Evoluční návrh se zabývá konstrukcí složitějších struktur z jednoduchých stavebních bloků s využitím EA. Populární je např. evoluční návrh logických obvodů na programovatelných hradlových polích.
EA často mívají kandidátní řesení (genotyp) zakódována do struktury, jejíž velikost je přímo úměrná velikosti požadovaného cílového objektu (fenotyp). Čím chceme mít komplexnější fenotyp, tím samozřejmě musí být rozsáhlejší i genotyp. Z toho však vyplývá několik problémů.

- Se zvyšující se velikostí požadovaného fenotypu se zvyšuje i množství dat, které je v evolučním procesu nutné zpracovat a tedy i prostor, ve kterém se řešení nachází. Velmi snadno se tak může stát, že evoluční algoritmus nezvládne efektivně s dostupným výpočetním výkonem takový prostor prohledat.

- Řešení nejsou škálovatelná. Vyevolvované řešení odpovídá pouze jednomu konkrétnímu genotypu. Pokud chceme řešení stejné funkce, ale jiné velikosti, je potřeba celý evoluční proces opakovat znovu s genotypy odpovídajícího rozsahu.

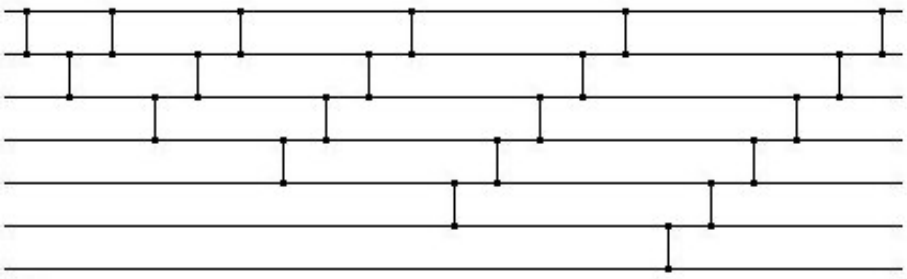
Tyto problémy jsou v evolučním návrhu mnohem palčivější než u běžných optimalizačních úloh, kde jde o nalezení správné kombinace číselných parametrů. Existuje několik přístupů jak je řešit, obecně jde však o použití genotypů, které neobsahují přímo zakódovaný fenotyp, ale instrukce nebo pravidla pro jeho vytvoření. Opakovaným prováděním těchto instrukcí získáváme rozsáhlejší fenotypy. Vyhodnocení fitness funkce jedince pak znamená výrobu fenotypu o požadované velikosti aplikací instrukcí obsažených v genotypu a výpočet jeho kvality.

4 ŘADICÍ SÍTĚ

Strukturou, na které budeme testovat kvalitu evolučního návrhu, jsou řadicí sítě [2, 3]. Řadicí síť (ŘS) je kombinační obvod s N vstupy a N výstupy. Jejím výstupem jsou vstupní hodnoty seřazené do neklesající posloupnosti. Základním blokem řadicí sítě je *komparátor*. Komparátor má 2 vstupy a 2 výstupy. Menší ze vstupních hodnot předá na výstup 0, větší naopak na výstup 1. ŘS jsou obvykle navrhovány pro předem daný pevný počet vstupů. Existuje však i několik konvenčních postupů konstrukce libovolně velkých ŘS. Jde např. o princip *vkládání*, který se využívá např. v algoritmu insert sort – každý $(m+1)$-tý vstup se vloží na správnou pozici do seřazené posloupnosti předchozích m vstupů. Tím se postupně vytváří seřazená posloupnost od nejnižších indexů vstupní datové sekvence. Takovouto síť zkonstruujeme přidáním dalšího vstupu a posloupnosti komparátorů na konec sítě (za poslední komparátor) tak, že porovávají sousední vstupy postupně od vstupů s nejvyšším indexem až po první vstup.

5 NÁVRH ŘEŠENÍ

Podívejme se ve zkratce na návrh EA, kterým budeme daný problém řešit. Genotypem je soubor přepisovacích pravidel, které jsou v každé iteraci tvorby výsledného řešení použity na všechny komparátory, pokud to jejich umístění dovoluje. Pravidla umožňují měnit zapojení komparátorů, posouvat je vertikálně i horizontálně, klonovat je, dávat komparátorům stejnou podobu, jakou má sousední, apod. Evoluci napomáháme také zařazením složených pravidel, jako je např. převzetí podoby sousedního komparátoru a výměna s druhým sousedem.
Je využit genetický algoritmus, kde genotyp obsahuje seznam pravidel zakódovaných jako binární řetězec. Na obr. 1 vidíme jeden z výsledků, kterého se podařilo dosáhnout. Síť je totožná se sítí vytvořenou metodou založenou na principu vkládání. Výhodou generovaných sítí je zejména jejich škálovatelnost. Zatím sice nepředčila konvenční způsob konstrukce, jde ovšem i o velmi ranné experimenty. Řešení se s vyšším počtem experimentů a následných úprav algoritmu a pravidel budou pravděpodobně dále zlepšovat.



Obrázek 1: Příklad ŘS pro 7 vstupů vygenerované pravidly získanými evolucí

6 ZÁVĚR

V článku jsme si nastínili problémy přímého zakódování kandidátního řešení a ukázali, že je možné je řešit netriviálním zakódováním genotypu. Řadicí sítě se pomocí evoluce běžně navrhují pro konkrétní počet vstupů, takový přístup však trpí popsanými problémy. V článku jsme demonstrovali automatizovaný návrh škálovatelných (tedy libovolně velkých) sítí s využitím netriviálního zakódování genotypu. Další výzkum v této oblasti bude zaměřen na zobecnění pravidel přepisovacího systému tak, aby bylo možné evolucí vytvářet úplně nová pravidla a tím dát algoritmu větší prostor k nalezení nových kvalitních řešení.

REFERENCE

[1] Kvasnička V., Pospíchal J., Tiňo P., Evolučné algoritmy, STU Bratislava, 2000.

[2] Knuth, D.: The Art of Computer Programming, Vol. 3 - Sorting and Searching. Addison-Wesley, 1973.

[3] Sekanina, L., Bidlo. M: Evolutionary Design of Arbitrarily Large Sorting Networks Using Development. In Genetic Programming and Evolvable Machines, pages 319 – 347, 2005.