COEVOLUTION OF SORTING NETWORKS AND TRAINING VECTORS

Luděk Žaloudek
Master Degree Programme(5), FIT BUT
E-mail: xzalou04@stud.fit.vutbr.cz

Supervised by: Lukáš Sekanina
E-mail: sekanina@fit.vutbr.cz

ABSTRACT

This paper deals with evolutionary design of sorting networks and concurrent coevolution of an incomplete set of training vectors. Genetic algorithms are used to achieve this goal. GAlib library for C++ was utilized for the implementation and evaluation of the experiments. The paper presents a method of designing simple sorting networks for selected number of inputs and implementation of the training vectors coevolution. By evolving the reduced number of these vectors, we achieve reduction of algorithm's execution time and/or improved results.

1. ÚVOD

Snaha o účinné řazení čísel (anglicky sorting) je jednou z klíčových oblastí na poli zefektivňování výpočetních postupů. Počítačoví výrobci uvádějí, že více než čtvrtina výpočetního času na jejich počítačích je využívána k řazení [1].

Stále vyšší nároky na výkon nutí konstruktéry hledat nové a rychlejší způsoby řazení. Jedním z takových způsobů je hardwarová realizace tzv. řadicích sítí, o kterých se podrobněji zmiňuji níže. Zatím bude stačit vysvětlení, že řadicí síť pracuje s předem známým počtem vstupů (řazených prvků), které jsou mezi sebou pospojovány komparátory, což jsou dvouvstupové prvky vracející minimum a maximum. I když jsou tyto prvky relativně levné na výrobu, snažíme se dále snížit cenu minimalizací jejich počtu. Nevýhodou ovšem je, že s rostoucím počtem vstupů sítě exponenciálně roste počet trénovacích vektorů (tj. kombinací čísel) nutných k jejímu otestování. Proto je výhodné se zabývat možnostmi, jak počet těchto vektorů co nejvíce snížit, aby se tím také snížila doba potřebná k vývoji řadicích sítí.

Cílem této práce je navrhnout evoluční metodou řadicí sítě, které budou srovnatelné s dosud nejlepšími nalezenými sítěmi počtem komparátorů. Druhým cílem je rozšíření zmíněné metody o koevoluci trénovacích vektorů a snížení jejich počtu z 2^N na co nejméně.

2. NAVRŽENÁ METODA

Nejprve bude vysvětlen princip jednotlivých částí problému a poté bude popsána navržená metoda.

2.1. NÁVRH ŘADICÍCH SÍTÍ S VYUŽITÍM GENETICKÝCH ALGORITMŮ A KOEVOLUCE

Řadicí síť (dále ŘS) je vlastně sekvence komparátorů propojených vodiči. Na vodiče se přivádí pevný počet vstupních proměnných. Komparátor funguje tak, že porovná obě proměnné na svých vstupech A a B a pokud je hodnota proměnné A větší než B, na výstupu jejich pořadí zamění.

Pokud chceme ověřovat ŘS, pomůže nám v tom následující teorém: *Pokud síť s N vstupy seřadí všech 2^N sekvencí nul a jedniček do neklesající posloupnosti, pak seřadí libovolnou sekvenci N čísel do neklesající posloupnosti* [1].

Evoluční návrh je stochastická metoda prohledávání stavového prostoru využívající tzv. *evoluční algoritmy*. Stavový prostor obsahuje možná řešení daného problému a my se pomocí algoritmů inspirovaných biologií snažíme najít to nejlepší z nich (nebo alespoň to, které nám dostatečně vyhovuje). Evoluční algoritmy pracují nad tzv. populací kandidátních řešení, na kterou jsou periodicky aplikovány genetické operátory produkující nové potomky (operace *křížení* a *mutace*). Potomci jsou poté vyhodnocováni *fitness funkcí* měřící jejich způsobilost a z nejlepších jedinců se tvoří nová populace. Postup se opakuje, dokud není dosaženo požadovaného výsledku nebo stanoveného počtu generací [2].

Základním a rovněž nejrozšířenějším typem *evolučních algoritmů* (EA) je tzv. *genetický algoritmus* (GA), který byl použit při popsaných experimentech. GA se drží postupu nastíněného výše s tím, že kandidátní řešení jsou zakódována řetězci binárních či celých čísel.

Z hlediska počítačového umělého vývoje je koevoluce dalším krokem přiblížení se přírodě. Je inspirována přirozeným soužitím živočišných druhů. Nejznámějším příkladem je systém dravec-kořist, kdy vývoj jednoho druhu vyvolává evoluční tlak na druhý a naopak.

Známým příkladem z oblasti umělé koevoluce je práce D. Hillise [3], který se zabýval problematikou ŘS o velkém počtu vstupů. Hillis v algoritmu použil několik populací „parazitů“, ve kterých se vyvíjely ty nejobtížněji seřaditelné trénovací vektory pro 16-vstupové ŘS. Moje práce se inspirovala Hillisovým počinem s tím, že vzhledem k omezeným výpočetním možnostem jsem implementoval jednodušší systém pro menší ŘS.

K implementaci experimentů s jednoduchým GA i s koevolucí jsem použil na FITu osvědčenou knihovnu GAlib pro C++.

2.2. METODA NÁVRHU

Pro návrh ŘS pomocí běžného GA bylo využito takové kódování kandidátních řešení, že jedinec byl reprezentován řetězcem celých čísel, kdy každá za sebou jdoucí dvojice čísel reprezentovala jeden komparátor a jeho připojení na příslušné vodiče ŘS. Jako genetické operátory byly použity operace jednobodového křížení s pravděpodobností 0,7 a mutace s pravděpodobností 0,01, kdy mutace znamená náhodnou změnu vodiče připojeného ke komparátoru. Velikosti populace se měnily se složitostí ŘS od 50 do 250 jedinců, délka jedince se měnila podle požadovaného počtu komparátorů. Fitness hodnota se rovnala počtu trénovacích vektorů, které dokáže ŘS správně seřadit, přičemž GA byl ukončen v případě dosažení fitness o hodnotě 2^N, kde N je počet vstupů sítě.

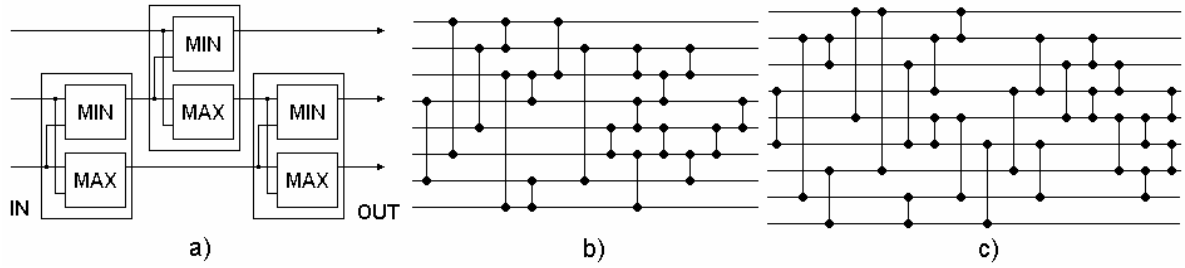
Metoda koevoluce pro 5-vstupové ŘS využívala stejné parametry jako předchozí návrh, navíc byla zavedena další evoluce množin trénovacích vektorů, která čítala obvykle 50 množin o délkách 12-24 vektorů. Fitness ŘS byla určována množstvím trénovacích vektorů, které dokáže síť seřadit a fitness množiny trénovacích vektorů závisela na tom, kolik ŘS nedokáže vektory v dané množině seřadit. Experimenty byly omezeny na 50 generací.

3. VÝSLEDKY

První částí popsaných experimentů byl evoluční návrh řadicích sítí pomocí obyčejného GA. Cílem bylo vyrovnat se nejlepším známým výsledkům pro řadicí sítě od velikostí 5 až 12 vstupů. Výsledky jsou shrnuty v tabulce 1. Ukázka nalezených sítí je na obrázku 1.

N	1	2	3	4	5	6	7	8	9	10	11	12
Nejlepší známé ŘS	0	1	3	5	9	12	16	19	25	29	35	39
Nalezené ŘS	n	n	n	n	9	12	16	19	25	30	36	n

Tabulka 1: Hodnoty nalezených řadicích sítí porovnané s dosud nejlepšími známými



Obrázek 1: (a) Struktura komparátorů v 3-vstupové ŘS zakódované v GA jako (1,2)(0,1)(1,2) a nejlepší nalezené ŘS pro (b) 8 a (c) 9 vstupů

Druhou částí práce byla snaha uplatnit koevoluci na vývoj řadicích sítí o velikosti 5 vstupů. Takto malé sítě byly zvoleny kvůli omezeným výpočetním možnostem, což znamenalo, že se neuplatnilo zmenšení výpočetního času užitím omezené množiny trénovacích vektorů. Nicméně se ukázalo, že aplikací koevoluce se mohou zlepšit výsledky EA. Pro 5-vstupovou řadicí síť se ukázaly vhodné množiny trénovacích vektorů s 16 jedinci. Výsledky jsou shrnuty v tabulce 2. Pro každý experiment bylo spuštěno 150-200 běhů evoluce.

	Velikost množiny trénovacích vektorů				Plná množina trénovacích vektorů (32)
	12	16	20	24	
Průměrná fit. ukončeného GA	31,74	31,84	31,8	31,79	31,76
Úspěšnost [%] (platných ŘS)	77,57	84,62	81,99	80,48	75,93

Tabulka 2: Porovnání výsledků vývoje 5-vstupové řadicí sítě s koevolucí a bez ní

4. ZÁVĚR

V práci se ukázalo, že evoluční návrh řadicích sítí konvenčním GA se s větším počtem vstupů stává problematickým. Problémy s návrhem (a nejen řadicích sítí) by mohla pomoci překonat nebo alespoň zmenšit aplikace postupu koevoluce, který může urychlit konvergenci GA k žádanému výsledku a u větších řadicích sítí snížit výpočetní čas.

LITERATURA

[1] Knuth, D.: The Art of Computer Programming (2nd ed.) Vol.3, Addison Wesley, 1998

[2] Bentley, P. (ed.): Evolutionary Design by Computers. San Francisco, Morgan Kaufmann Publishers, 1999, Chapter 1

[3] Hillis, W. D.: Coevolving parasites improve simulated evolution as an optimization procedure, Physica D, 42: 228-234, 1990