

Vlastnosti Kartézského genetického programování

Petr Burian

Katedra aplikované elektroniky a telekomunikací, Západočeská univerzita v Plzni

Features of Cartesian Genetic Programming

Abstract

This paper deals with Cartesian Genetic Programming. It explores the dependence of evolution process on algorithm control parameters. Two logic circuits are used as benchmark – a 5bit majority circuit and a 3b x 2b multiplier.

Keywords

Cartesian Genetic Programming, Evolutionary design.

Úvod

Relativně novým trendem na poli návrhu číslicových systémů je využívání Evolučních výpočetních technik (EVT). Jedná se o skupinu optimalizačních algoritmů, jichž se využívá k hledání vhodného nastavení rekonfigurovatelných struktur a tím i k hledání inovativních zapojení či vlastností obvodů.

Kartézské genetické programování

Velký počet aplikací využívajících evoluční algoritmy pro rekonfiguraci dané struktury je založen na Kartézském genetickém programování (*Cartesian Genetic Programming* - CGP). Tento algoritmus, jehož vznik se datuje do roku 1999, nepředstavuje pouze definici vyhledávacího algoritmu, ale nabízí komplexní přístup k návrhu zejména číslicových kombinačních obvodů. [1]

Reprezentace obvodu

Obvodové řešení je v CGP popsáno pomocí maticového uspořádání funkčních buněk (bloků). Velikost pole těchto bloků definujeme jako $n_c \times n_r$ (sloupce x řádky). Funkční buňka má obecně n_n vstupů a jeden výstup, přičemž nejčastěji je počet vstupů roven dvěma. Každá buňka může realizovat jednu ze skupiny definovaných funkcí. Tyto funkce mohou být naprosto triviální (například logický součet) nebo se může jednat o „vyšší“ funkce typu aritmetická násobička, výběr maxima atd. Definice skupiny funkcí vždy záleží na konkrétní aplikaci. Platí obecné pravidlo, že pokud navrhujeme obvod, jehož cílem je přinést nové úspornější řešení, volíme co nejelementárnější funkce (většinou logické funkce a jejich kombinace). Tento přístup má však i svá negativa. Při použití elementárních funkcí není možné navrhovat složitější obvody, u nichž se předpokládá, že budou tvořeny značným počtem logických prvků (a tedy definovaných elementárních funkcí). Velký počet prvků vede k dlouhému chromozomu evolučního algoritmu a tedy i k rozšíření prohledávacího prostoru. Algoritmus pak v rozlehlém prostoru jen velmi těžce hledá vyhovující řešení – tento problém se obecně nazývá škálovatelnost a je jedním z klíčových omezení návrhu obvodů založených na evolučních algoritmech.

Dalším parametrem, který musí být při inicializaci definován, je počet primárních vstupů (n_i) a výstupů (n_o). Jedná se o vstupy a výstupy hledaného (navrhovaného) obvodu. Algoritmus v průběhu návrhu hledá vhodné propojení jednotlivých funkčních buněk, buněk s primárními vstupy a výstupy. Pro konektivitu CGP platí několik základních

pravidel. Nejsou povoleny žádné zpětné vazby, neboť jejich přítomnost by značně komplikovala proces ohodnocení kandidátních obvodů. Možnost dopředných vazeb je limitována tzv. L-back parametrem (*Level Back Parameter*), který nabývá hodnot z intervalu $\langle 1; n_c \rangle$ (kde n_c je počet sloupců). Pokud L-back = 1, znamená to, že do buňky mohou vstupovat jen a pouze výstupy bezprostředně předcházejícího sloupce. Na druhou stranu, pokud L-back = n_c , vstup buňky může být propojen s jakýmkoli výstupem buňky v předcházejících sloupcích a tak vytvářet i složité kombinační obvody. Naopak L-back = 1 umožní vložit mezi sloupce registry a použít tak zřetězené zpracování. Primární vstupy mohou být připojeny k jakékoli funkční buňce. V praxi se ale často omezuje připojení primárních vstupů například na prvních několik sloupců. Výstupy obvodu mohou být připojeny ke kterékoli buňce v poli.

Kódování chromozomu a evoluční algoritmus

Konfigurace funkčních buněk a jejich propojení s okolím jsou definovány pomocí konfiguračního řetězce – chromozomu. V případě CGP se jedná o lineární řetězec celočíselných hodnot. Jednotlivé primární vstupy a výstupy buněk jsou označeny vzestupnou řadou celočíselných hodnot. Celková délka chromozomu je pak definována jako $d = n_c \times n_r \times (n_n + 1) + n_o$. Pokud uvažujeme, že každá funkční buňka disponuje dvěma vstupy, pak vyžaduje pro svou konfiguraci tři celočíselné hodnoty – dvě pro určení připojení vstupů a jednu pro konfiguraci funkce, kterou bude vykonávat. Konec řetězce je tvořen hodnotami ukazujícími na body (výstupy buněk), které budou považovány za výstup navrhovaného obvodu.

V CGP je používán velmi jednoduchý evoluční algoritmus, který je založen na Evoluční strategii (ES [2]), konkrétně na variantě $(1 + \lambda)$ -ES, kde λ je obvykle rovna 4. Algoritmus ovšem pracuje nad chromozomem celočíselných hodnot, přičemž jen některé kombinace jsou vzhledem k počáteční konfiguraci CGP přípustné, a tak musí být vždy zajištěno, že jednotlivé jeho části budou nabývat korektních hodnot. Mutační operátor je v CGP definován jako náhodná změna genu (jedna celočíselná hodnota). S mutací souvisí další řídicí parametr – četnost mutace (*mutation rate*). Ta určuje počet genů mutovaných v jednom procesu mutace.

Závislost průběhu evolučního návrhu na řídicích parametrech

Jako bylo zmíněno výše, CGP obsahuje několik řídicích parametrů, jež výrazně ovlivňují průběh návrhu obvodu. Jejich vhodné nastavení lze jen obtížně předpokládat, a tak je jejich správná volba vždy otázkou empirie. Následující text popisuje experiment, při němž je měněn počet a geometrické rozložení funkčních buněk v CGP. Jako testovací úlohy byly použity následující dva logické kombinační obvody: návrh násobičky 3 bity x 2 bity a majoritního 5bitového obvodu.

Před započítím experimentů byly definovány následující výchozí podmínky. Použití klasického algoritmu typu $(1 + 4)$ s četností mutace 2. Omezení algoritmu na 1 000 000 generací. Funkční bloky disponují dvěma vstupy. L-back parametr bude vždy nastaven na maximální hodnotu. Skupina funkcí pro funkční buňky: AND, OR, XOR, „propojení“. Funkce „propojení“ je definována jako $y = x$. Jedná se o funkci s jedním vstupem, proto bylo rozhodnuto, že x bude reprezentováno prvním vstupem buňky a druhá vstupní hodnota bude ignorována. Klíčovým prvkem každého evolučního návrhu je hodnotící (účelová) funkce (fitness function) vyjadřující kvalitu kandidátních řešení. Použitá hodnotící funkce má dvě role. První část hodnotí kandidátní řešení na základě jeho schopnosti vykonávat funkci logického obvodu podle požadované pravdivostní tabulky, řekněme tedy z hlediska funkce obvodu. Druhá část hodnotící funkce přispívá k optimalizaci obvodu z hlediska nároků na počet použitých logických elementů. Což je *de facto* účel evolučního návrhu – nacházet dosud neznámá řešení nebo řešení, která mají

jistou přidanou hodnotu oproti řešením získaným konvenčními návrhovými metodami. Celkově definuje použitou účelovou funkci následující vztah:

$$f(x) = \begin{cases} b & \text{když } b < n_o 2^{n_i}, \\ b + (n_c n_r - z) & \text{když } b \geq n_o 2^{n_i}, \end{cases}$$

kde b představuje míru kvality funkce a z počet nepoužitých buněk. [1] Pokud uvažujeme násobičku $3b \times 2b$, jde o obvod s 5bitovým vstupem i výstupem. Pokud je kandidátní obvod z hlediska funkce plně vyhovující, znamená to, že dochází ke shodě mezi pravdivostní tabulkou kandidátního řešení a tabulkou hledaného obvodu ve všech řádcích. Maximální hodnota b bude tedy 160 ($2^5 \times 5$), v případě majoritního obvodu 32. Algoritmus ukončí návrh obvodu, pokud bude nalezeno plně funkční řešení a počet použitých hradel bude roven 13 v případě násobičky a 10 v případě majoritního obvodu. Tyto hodnoty byly zjištěny experimentálně – představují zřejmě maximální optimalizaci, které lze dosáhnout se zvolenými funkcemi buněk.

Evoluční návrh je testován na následujících konfiguracích pole funkčních buněk: 10×10 , 5×5 a 1×20 . Pro získání relevantních statistických údajů o průběhu návrhu obvodu je každý běh evoluce spuštěn 10krát.

Tabulka 1: Výsledky evolučního návrhu pro různé konfigurace CGP

Obvod	Konfigurace buněk	Průměrný počet generací	Plná funkčnost	Průměrně použitých hradel	Nejúspornější řešení [počet hradel]
Násobička $3b \times 2b$	10×10	678657	100%	14	13
Násobička $3b \times 2b$	5×5	203482	100%	13,1	13
Násobička $3b \times 2b$	1×20	188351	100%	13	13
Majoritní 5-bit obvod	10×10	1000000	100%	11,9	11
Majoritní 5-bit obvod	5×5	619462	100%	10,9	10
Majoritní 5-bit obvod	1×20	545586	100%	10,7	10

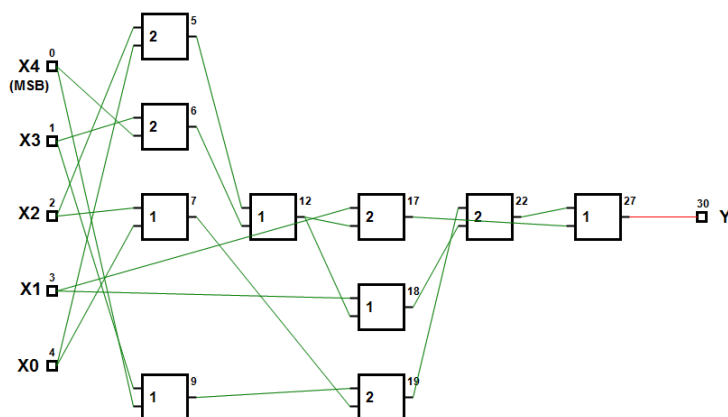
V Tabulce 1 jsou znázorněny výsledky těchto běhů návrhového algoritmu. Třetí sloupec tabulky udává průměrný počet generací evoluce při dané konfiguraci. Připomeňme, že algoritmus je ukončen v okamžiku, kdy dosáhne plné optimalizace (námi zvolené, nikoli nutně absolutně nejlepší), nebo když překročí 1 milion generací. Položka „Plná funkčnost“ informuje o podílu běhů, ve kterých bylo nalezeno plně funkční řešení obvodu. Nároky obvodu na počet hradel jsou vyjádřeny v posledních dvou sloupcích. Můžeme vidět jejich průměrný počet a nejlepší (nejúspornější) nalezené řešení. Při pohledu na výsledky je potěšující, že při všech evolučních bězích, bez ohledu na zvolenou konfiguraci, bylo nalezeno řešení, které představuje plně funkční obvod. Ovšem návrh majoritního obvodu při konfiguraci 10×10 nebyl schopen naleznout námi zadanou hranici pro ukončení algoritmu, tzn. 10 použitých hradel. I v případě násobičky poskytuje tato konfigurace nejhorší výsledky, avšak limitu 13 hradel dosáhla. Zhoršení výsledků evoluce při velkém počtu funkčních buněk je způsobeno tím, že tato konfigurace vede k výrazně delšímu chromozomu a tím pádem se stává situace prohledávání prostoru řešení náročnější. Ideální se zdá být situace, kdy počet buněk v CGP je mírně větší než počet buněk, které mají tvořit výsledný obvod. Samozřejmě že tento počet se nedá vždy dobře odhadnout, a proto se musí vycházet ze zkušenosti. Pokud bychom však zvolili příliš malý počet buněk, nastala by situace, kdy by algoritmus nemusel najít plně funkční řešení.

V následujícím experimentu byla prováděna evoluce obou obvodů při konfiguraci 1×20 , ovšem nyní byla měněna hodnota četnosti mutace v rozsahu 1–3. Měli bychom tak nalézt optimální hodnotu, při které bude algoritmus vyhledávat nejefektivněji (to platí jen pro danou konfiguraci pole buněk).

Tabulka 2: Výsledky evolučního návrhu pro různé četnosti mutace

Obvod	Četnost mutací	Průměrný počet generací	Plná funkčnost	Průměrně použitých hradel	Nejúspornější řešení [počet hradel]
Násobička 3 b x 2 b	1	234314	100%	13,2	13
Násobička 3 b x 2 b	2	188351	100%	13	13
Násobička 3 b x 2 b	3	288122	100%	13,1	13
Majoritní 5-bit obvod	1	458635	100%	10,2	10
Majoritní 5-bit obvod	2	545586	100%	10,7	10
Majoritní 5-bit obvod	3	756880	100%	10,8	10

Při pohledu na výsledky v Tabulce 2 můžeme vidět, že v některých případech může mít četnost mutace velmi výrazný vliv na efektivitu algoritmu. Zaměříme-li se na majoritní obvod, je vidět, že počet průměrných generací pro četnost mutace 1 je výrazně nižší než při četnosti rovné 3. Tomu odpovídá i počet použitých hradel. Při všech bězích bylo nalezeno požadované řešení, tedy minimální počet hradel. Obecně se doporučuje volit tuto řídicí proměnnou v rozsahu 1–3. Příliš vysoká hodnota četnosti mutace způsobuje razantní zásahy do chromozomu a může jen obtížně provádět drobné změny v zapojení obvodu. K výsledkům v Tabulce 2 je nutné poznamenat, že naprosto relevantní data bychom dostali, kdybychom prováděli například 1 000 běhů každé evoluce. Nicméně ani nadějně výsledky v podobě statistických hodnot neznamenají, že algoritmus nemůže uvíznout v lokálním extrému apod. Pro ukázkou je na obrázku 1 zobrazeno schéma majoritního 5bitového obvodu, který byl nalezen CGP při konfiguraci 5 x 5. Číslice ve funkčních blocích symbolizují funkci, kterou buňka vykonává: 1 – OR, 2 – AND.

**Obrázek 1: Majoritní 5 bitový obvod navržený pomocí CGP**

Závěr

Výše provedené experimenty naznačují, jak postupovat při nastavování řídicích parametrů CGP. Podrobná analýza je nutná především tehdy, když budeme chtít princip CGP využít ve vyvíjejících se obvodech. Tam je na rozdíl od obvyčejného evolučního návrhu kladen velký důraz na co nejrychlejší nalezení „přijatelného řešení“. Při evolučním návrhu naopak není čas evoluce zásadní, primární je získat obvodové řešení, které je nějakým způsobem vhodnější než řešení konvenční.

Literatura

- [1] Sekanina L. a kolektiv. *Evoluční hardware: Od automatického generování patentovatelných invencí k sebemodifikujícím se strojům*. Academia : Praha 2009.
- [2] Oplatková, Z., a další. *Evoluční výpočetní techniky - principy a aplikace*. Praha : BEN - technická literatura, 2008. ISBN 978-80-7300-218-3.