

回路パターンによる回路変換を併用した トランスダクション法

熊沢雅之† 澤田直‡ 上林彌彦†

† 京都大学大学院工学研究科

京都市左京区吉田本町 京都大学

TEL:075-753-5395 FAX:075-753-4971

‡ 九州大学大学院システム情報科学研究科情報工学専攻

福岡県春日市春日公園 6-1 九州大学

TEL:092-583-7622 FAX:092-583-1338

E-Mail : † {kumazawa,yahiko}@kuis.kyoto-u.ac.jp, ‡ sawada@csce.kyushu-u.ac.jp

トランスダクション法は、許容関数を用いて回路変換を行なう手法である。トランスダクション法は、初期回路の形状や回路変形の適用順序によっては、最適解を発見するより前にそれ以上処理を行えなくなってしまうことがある。この状態を局所解と呼ぶ。本稿では局所解脱出の1手法として、パターンに基づく回路変換を併用することにより局所解から抜け出す方法について検討する。この手法においては、回路形状によって適したパターンが異なることが知られている。そこで我々は、回路変換時にその時点での回路形状に適したパターンを選択して適用する手法を提案する。パターン発見は一般にグラフの同型判定と同様に困難な問題になるが、個々では比較的発見の容易なパターンに絞っている。また、ファンインの直列分割を用いたトランスダクション法の手続きを変更して上記手法との併用を行う。本手法をベンチマーク回路に適用した結果、このような単純なパターンによる回路変換が十分有効であることが示された。

あらまし

キーワード

論理設計 論理最適化. トランスダクション法

Transduction Method with Pattern-based Transformations

M. Kumazawa†, S. Sawada(Kyushu Univ.), Y. Kambayashi†

† Department of Information Science, Graduate School of Engineering, Kyoto University

TEL:075-753-5395 FAX:075-753-4971

‡ Department of Computer Science and Communication Engineering, Kyushu University

TEL:092-583-7622 FAX:092-583-1338

E-Mail : † {kumazawa,yahiko}@kuis.kyoto-u.ac.jp, ‡ sawada@csce.kyushu-u.ac.jp

Abstract

The Transduction Method transforms a network based on the concept of the Permissible Function. This method transforms an initial network repeatedly and sometimes falls into local minimum depending on the initial network and order of transformation application. In this paper we mention a method to escape from local minimum by using some pattern-based transformations. In this method, the most effective pattern depends on the shape of a network. We select and use a pattern suitable for the shape of a network among three ones when we execute pattern-oriented transformations. Next, we improve the procedure of the Transduction Method with the serial duplication of fan-in and use our method together. The results of experiments for bench mark circuits shows that the effectiveness of pattern-based transformations although we selected simple patterns which can be easily detected.

key words

logic synthesis, logic optimization, Transduction Method

1 はじめに

VLSI 技術および論理回路の自動設計化技術の進歩に伴い、計算機による論理回路設計が行なわれるようになった。

1970年代前半にイリノイ大学において開発されたトランスダクション法 [1][2] は、許容関数という概念に基づき回路内の潜在的なドントケアを有効に利用して形状変換と冗長部分の削除を行う手法である。近年、SBDD[4]との併用により論理関数を効率よく表現することが可能となり、実用的な規模の回路を計算機上で扱うことが可能となった。

トランスダクション法は、初期回路の形状や回路変形を行う順序によっては最適解を発見するより前にそれ以上処理を行えなくなってしまうことがある。この状態を局所解と呼ぶ。

この局所解脱出の1手法として、あらかじめ回路の変換パターンを用意しておき、それを用いて回路全体を変化させる手法 [7][8] が研究されている。この手法では、まず初期回路に対しトランスダクション法により最適化を行い、これ以上トランスダクション法による最適化が不可能な状態になった時点でパターンに基づく回路変換を回路全体に実行する。このことにより、回路形状が変化し、回路中のいくつかのゲートの論理関数が変化するため、再度トランスダクション法による最適化を実行できる可能性が生じる。

一方、局所解脱出のための回路変換においては、使用されるパターンによりその効果が異なってくる。文献 [8] においては、回路変換のためのパターンが3種類提案されており、各々のパターンは一度の最適化プロセスにおいて一つずつ用いられている。その結果として、回路によって適したパターンが異なることが報告されている。また、パターンに基づく回路変換は、一度の最適化プロセスにおいて複数回実行されるため、その時点での回路形状によっても適したパターンが異なるものと考えられる。

そこで本稿では、最適化プロセスにおけるパターンに基づく回路変換を実行する各時点で、複数のパターンの中から一つを選択して適用する手法について提案する。回路変換パターンの選択は、各時点での回路形状に基づいて行う。具体的には、(a) 回路全体に存在するパターンの数が最大となるパターンを選ぶ、(b) パターンに基づく回路変換を実行後に論理関数の変化するゲート数が最大となるパターンを選ぶ、の2種類の方針で選択を行うこととする。これは、回路変換の目的が回路形状を変化させることにより、各ゲートの論理関数を変化させることにあるためである。

また、文献 [8] ではファンインの直列分割 [6][9] を用いた局所解脱出の手法が研究されている。この手法は、ファンインの制限なしのトランスダクション法とファンインの直列分割を併用することにより、局所解の脱出を行うものである。本稿ではこの手法と複数のパターンを併用する手法とを組み合わせ、手続きの変更を行う。具体的にはファンイン

制限なしのトランスダクション法が実行される回数を増やすことにより、より自由度の高い最適化が行われるように改良を施す。

上記の変更点について、従来手法との比較実験をMCNCベンチマーク回路に対して行なった結果、殆どどの回路においてコストが改善され、本手法の有用性を示すことができた。

2 基本的事項

本章では、トランスダクション法の基本的な概念を述べる。論理回路はNORゲートのみのものを扱っているが、一般化は容易である。

2.1 許容関数

あるゲート(結線)の実現する関数 f を、論理関数 f' で置き換えても回路の出力に変化が無い時、そのような f' をゲート(結線)の許容関数(Permissible Function)であるという。また許容関数の集合の中で同時に置き換え可能なものからなる部分集合をCSPFs(Compatible Set of Permissible Functions)と呼び、0, 1, * (*don't care*) の3値をとる関数 G で表す。

2.2 許容関数集合とトランスダクション法

ある回路中の結線の許容関数集合に恒偽関数が含まれる時、この結線は削除可能(Disconnectable)である。また、ある結線をゲート v に接続した時の v の実現する関数が v の許容関数集合に含まれる時、この接続によって回路の出力は変化しない。ゲート v_i から到達可能なゲート集合に含まれないゲート v_j の出力をゲート v_i の入力に接続可能(Connectable)である条件は、以下のように表すことができる。

$$G^{on}(v_i) \cap f^{on}(v_j) = \emptyset \quad (1)$$

ここで G^{on} , f^{on} はCSPFs, 論理関数のON集合を表す。

トランスダクション法のうち一般によく用いられる手法である手続きC/DC (Connectable/ Disconnectable) は、この性質を利用して回路を変形、単純化する手法である。

2.3 パターンに基づく回路変換を併用したトランスダクション法

トランスダクション法は与えられた初期回路を元に回路変形を繰り返すため、初期回路の形状や回路変形の順序によってはこれ以上変形できない状態、すなわち局所解に陥ることがある。

この局所解を脱出する手法として、パターンに基づく回路変換を併用する手法が研究されている。この種の手法としては、文献 [7][8] に述べられた手法があり、図1に示すような流れで処理を行っている。

手続き C/DC において局所解に陥った時点でパターンに基づく回路変換を施すことにより、回路の形状が変化し回路中のいくつかのゲートの論理関数が増える。このことにより、変換前と異なった CSPFs が得られるため、手続き C/DC による最適化が継続できる可能性がある。また、この手法は論理関数を計算する必要がないため、大変高速に行えるという特徴がある。

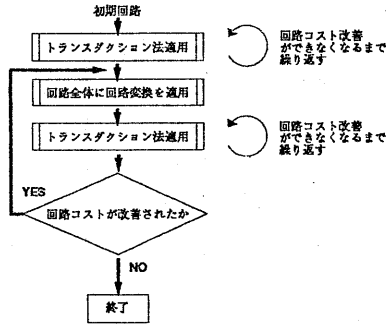


図 1: パターンに基づく回路変換を併用したトランスタクション法

3 複数の回路変換パターンの選択的使用

3.1 回路変換パターン

前章で述べた手法において使用されるパターンは一回の最適化プロセスにおいては一種類のみであった。しかし、回路の形状によって有効なパターンは異なるため、最適化プロセスにおける回路変換実行時点での回路形状により、最適なパターンを選択する必要がある。そこで我々は、文献 [3] で紹介されている以下の三種類のパターンを回路変換時点で選択的に使用する手法を提案する。

1. 並列素子の併合 (図 2)
2. 入力結線の回路の入力端子側へのシフト (図 3)
3. 入力結線の回路の出力端子側へのシフト (図 4)

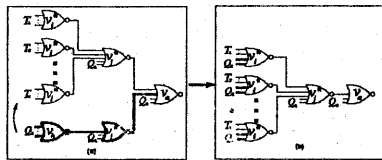


図 2: 並列素子の併合

図 1 におけるパターンに基づく回路変換実行の際に次節に述べる選択方針により 1. ~ 3. のパターンから一つのパターンを選択し、回路全体に適用する。

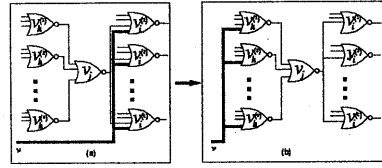


図 3: 入力結線の回路の入力端子側へのシフト

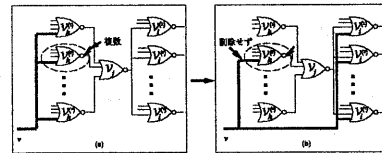


図 4: 入力結線の回路の出力端子側へのシフト

3.2 回路パターンの選択方針

局所解脱出のためにパターンに基づく回路変換を行う目的は、回路変換により回路形状を変化させ、回路中のゲートの論理関数を変化させることである。このことにより、異なる CSPFs が得られるため手続き C/DC を再度実行できる可能性が生じる。

従ってパターンとしては、(a) 回路形状を大きく変化させるようなパターン、(b) 回路変換により多くのゲートの論理関数を変化させるパターンが望ましいと考えられる。このことから、我々は以下の方針でパターンの選択を行う。

選択方針 1: 回路中のマッチするパターンの数が最大のパターンを選択

選択方針 2: 回路中のマッチするパターン全てを変換したときに論理関数の変化するゲートの数が最大のパターンを選択

選択方針 2 において、論理関数の変化するゲートの数は各パターン毎に異なる。例として並列素子の併合について説明する。図 5 のように点線で囲まれたゲートの論理が変化している。

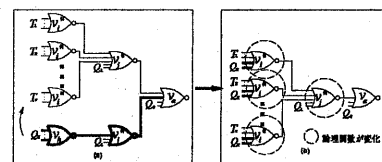


図 5: 並列素子の併合におけるゲートの論理関数の変化

すなわち、点線で囲まれた $r + 1$ 個のゲートの論理関数が増える。このことにより $v_i^{(1)}, v_i^{(2)}, \dots, v_i^{(r)}$ の入力側のゲート全てに CSPFs の変化が現れるため、手続き C/DC が継続できる可能性が生じる。

4 ファンインの直列分割との併用

4.1 ファンインの直列分割

ファンインの直列分割 [6] (図 6) はゲートのファンインが制限値を越えた場合に NOR ゲート対を図のように挿入することにより、ファンインの制限値を満たすものである。

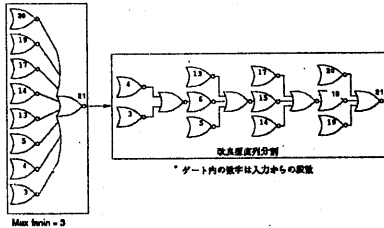


図 6: ファンインの直列分割

改良型直列分割 [8] においては分割の際、ゲートの段数ができるだけ増加しないように分割できるように改良がされている。

4.2 ファンインの直列分割と手続き C/DC の併用

局所解脱のための回路変換を併用した手法に文献 [9][8] などで提案されている手法がある。これはファンイン制限なしの手続き C/DC とファンインの直列分割を併用することにより局所解脱を行うものである。ファンイン制限を設けずに手続き C/DC を行うことにより、自由度の高い最適化が可能である。最適化後の回路はファンイン制限を満たしていないため、直列分割により全ゲートが制限を満たすよう変形を行う。その後ファンイン制限を設けた手続き C/DC を行うことにより、より回路コストが改善できる。

4.3 ファンインの直列分割とパターン変換手法の併用

ファンインの直列分割を併用したトランスダクション法は大変強力であり、この手続きに複数パターンの併用手法を組み込むことにより、より強力な最適化が可能であると考えられる。

そこで我々は、最適化の自由度を上げるためにファンイン制限なしの手続き C/DC が行われる回数が増加するように手続きを変更し、3 章で述べた複数パターンの併用手法を手続きに組み込むことにより、より回路コストの改善が行われるような手続きを提案する。図 7 にその手続きを示す。

直列分割実行後にファンイン制限なしの手続き C/DC を実行することにより、より自由度の高い最適化が可能である。しかし、ファンイン制限なしの手続き C/DC においては接続可能なゲートの探索を多くのゲートに対して行うため、計算時間は非

常に長い。従って本手続においては回路コストが改善される分、計算時間は長くなると考えられる。

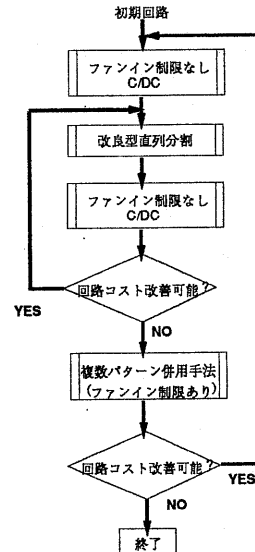


図 7: ファンインの直列分割手法と複数パターン併用手法の併用

5 実験結果

3 章及び 4 章で述べた手続きを C 言語を用いて作成し、提案手法の有用性を検証した。実装の際には、論理関数を計算機で効率よく表現するために、現 NTT の湊真一氏による SBDD パッケージ [4] を用いた。実験は Sun 社の Ultra2 (200MHz) 上で行った。初期回路としては、MCNC ベンチマーク回路 [5] をファンイン 4 までの NOR ゲートにマッピングしたものを用いた。

3 章で述べた複数パターンの併用手法による実験を行い、パターンを単独で使用した場合との比較を行った。表 1 にその結果を示す。表中の各数字はゲート数 / 結線数 / 回路段数 / 計算時間 (秒) (初期回路を除く) を表す。

Trans の列は手続き C/DC を用いた実験結果、単独パターン利用は各変換パターンを単独で使用した結果のなかで最も良い結果を表す。最後の 2 列は提案手法の実験結果を表す。最大数マッチングは回路変換の際に回路中に存在するパターン数が最大のパターンを選択した場合を表し、変化最大パターンは回路変換を実行後に論理関数の変化するゲート数が最大のパターンを選択した場合を表す。結果のうち太字で示されたものは、「単独」よりも良い結果を、下線で示されたものは最も良い結果を表す。

結果より選択方針 1 においては全回路中 81%、選択方針 2 においては 95% の回路において回路コストの改善が見られた。選択方針 1 の t481 においてはゲート数、結線数ともに単独と比較して 50% 以上の改善がされている。また、計算時間では単

独の場合が平均 4173 秒であるのに対し、選択方針 1 は 4760 秒、選択方針 2 では 4648 秒となっている。

これらの結果より、一回の最適化で一つのパターンを用いるよりも複数のパターンを選択的に使用したほうがより回路コストの改善が行えることがわかった。また、回路パターンの選択は大変高速に実行できるため、計算時間には殆んど影響を与えないこともわかった。選択方針としては 2 のように論理関数の変化するゲート数で評価する場合のほうが良い結果が多いが、選択方針 1 のほうがより良い結果を出している回路も存在するため、回路により適当な選択方針が異なることもわかる。

次に 4 章で述べたファンインの直列分割との併用手法について実験を行い従来手法との比較を行った。表 2 にその結果を示す。表中の数字は表 1 と同様である。

Trans は表 1 と同様で、基本手法は文献 [8] における手続きを使用した結果を表す。最後の 2 列は図 7 に示した改良型の手続きを適用した結果を示す。ボタン変化なし繰返しは図 7 における複数パターン併用手法を手続き C/DC に置換えたものを表し、ボタン変化付繰返しは複数パターン併用手法を用いたものを表す。図中の太字は基本手法よりも良い結果を、下線で示されたものは最も良い結果を表す。

結果より、ボタン変化なし繰返しにおいては全回路中 46%、ボタン変化付繰返しでは全ての回路で回路コストの改善が見られた。一方計算時間は従来手法が平均 271 秒であるのに対し、ボタン変化なし繰返しでは 2107 秒、ボタン変化付繰返しでは 988 秒となっている。

これらの結果から提案手法により従来手法と比較して、殆んどの場合に回路コストの改善が行えることがわかる。しかし、ファンイン制限なしの手続き C/DC の計算時間がファンイン制限ありの手続き C/DC よりも非常に長いために、計算時間は非常に長くなっている。また、提案手法に手続き C/DC を用いた場合よりも複数パターンの併用手法を用いた場合のほうが計算時間が短くなっている。これは複数パターンの併用手法が強力であるために、これ以上回路コストが改善できない状態に陥るのが早くなり、その結果として手続きの繰返しの回数が少なくなっているためと考えられる。

6 結論及び今後の課題

本稿ではトランスダクション法における局所解脱出の手法として、複数の回路パターンにより回路変換を行う手法を提案した。また、回路変換を実行する時点で最適なパターンを選択する方針として 2 種類の評価方法を検討した。従来は一度の最適化に種類のパターンのみを用いていたため、回路により結果が良いパターンが異なっていたが、本手法により用意した全てのパターンの特徴を生かすことが可能となった。実験の結果より、殆んどの回路で回路コストが改善された。選択方針としては回路により適当なものとは異なることがわかった。また、パター

ンの選択に必要な計算時間が非常に短いため、全体の計算時間も殆んど変化しなかった。次にファンインの直列分割併用手法に改良を加えることにより、より回路コストを改善することが可能となった。しかし、自由度の高い最適化を数多く行うため、計算時間は非常に長くなった。

今後の課題としては、新たな変換パターンを作成して選択肢に加えることと、手続きを最適化することにより計算時間の短縮を行うことなどを考えている。

謝辞

有益な助言をしていただいたイリノイ大学の室賀教授、及び上林研究室の皆様へ感謝します。

参考文献

- [1] 上林彌彦, 室賀三郎, "Permissible Function による論理関数の単純化", 電子通信学会研究報告 AL73-13 (1973).
- [2] S.Muroga, Y.Kambayashi, H.C.Lai, J.N.Culliney, "The Transduction Method-Design of Logic Networks Based on Permissible Functions", IEEE Trans.Comput., pp.356-359, Nov.1989
- [3] Y.Kambayashi, H.C.Lai, S.Muroga, "Pattern - Oriented Transformations of NOR Networks", Technical Report UIUCDCS-R-90-1573, Dep. Comput. Sci., Univ. of Illinois (February 1990).
- [4] S.Minato, N.Ishiura, S.Yajima, "Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean Function Manipulation", Proceedings of 27th Design Automation Conference(1990), 52-57.
- [5] S.Yang, "Logic Synthesis and Optimization Benchmarks User Guide Version 3.0", in 1991 MCNC International Workshop on Logic Synthesis (1991).
- [6] S.Sawada, Y.Kambayashi, S.Muroga, "Generation of Fan-in Restricted Initial Networks for Transduction Method", Proceedings the Synthesis and Simulation Meeting and International Interchange, SASIMI '92 pp. 36-45 (1992).
- [7] 澤田直, 日野健介, 上林彌彦, "パターンベースによる冗長性の負荷を考慮したトランスダクション法に関する考察", 情報処理学会第 48 回全国大会, 5B-7, pp. 83-84 (1994).
- [8] 熊沢雅之, 澤田直, 上林彌彦: "回路パターンに基づく回路変換システムの開発", 情処第 52 回全大, 1K-7, 1996 年 3 月
- [9] 高田秀志, 石垣博康, 上林彌彦, 室賀三郎, "トランスダクション法の初期回路における一般化直列分解に基づくファンイン制限", 電子情報通信学会論文誌, D-I Vol.J80-D-I No.4, 1997

表 1: 複数パターンの併用手法

回路名	初期回路	Trans	単独パターン利用	最大数マッチング	変化最大パターン
9symml	168/402/14	152/376/14/6	132/379/15/16	132/379/15/16	<u>129/366/13/22</u>
C1908	718/1335/37	438/954/32/2535	414/948/33/3863	<u>411/947/33/4632</u>	414/948/33/4092
C432	209/421/25	124/297/26/65	123/293/23/143	<u>122/290/31/212</u>	<u>122/290/31/209</u>
alu2	360/747/39	215/524/36/84	193/498/38/130	<u>187/488/33/170</u>	<u>188/491/33/153</u>
alu4	720/1441/42	455/1083/39/489	416/1063/49/1005	<u>409/1051/39/1320</u>	<u>411/1045/39/2129</u>
apex7	268/501/17	216/418/21/12	207/421/17/23	<u>204/410/17/35</u>	<u>205/413/17/30</u>
b9	126/247/10	105/213/10/5	101/205/10/9	105/213/10/6	<u>99/200/10/12</u>
c8	188/371/7	138/274/7/9	116/263/7/13	<u>114/254/7/15</u>	<u>114/254/7/15</u>
cm163a	55/92/9	47/87/9/0	43/84/7/0	<u>41/85/7/1</u>	<u>41/85/7/1</u>
cordic	105/197/13	64/122/11/1	59/134/14/3	<u>59/130/11/4</u>	<u>59/130/11/4</u>
example2	366/642/12	322/592/18/31	315/590/18/51	<u>320/588/18/53</u>	<u>313/586/18/65</u>
frg1	108/225/15	99/213/15/66	88/212/13/244	<u>85/202/13/320</u>	<u>85/202/13/316</u>
i2	175/407/12	161/381/12/8	151/436/10/21	151/379/10/25	<u>151/379/10/25</u>
mux	90/176/14	59/125/10/1	56/121/10/2	<u>54/118/10/2</u>	<u>54/118/10/2</u>
pml	53/107/6	49/97/7/0	45/89/7/0	45/89/7/0	<u>42/83/7/1</u>
sct	117/243/8	74/153/7/2	72/148/7/2	<u>70/143/7/3</u>	<u>70/143/7/3</u>
t481	3393/8144/20	849/1987/22/36784	610/1450/18/63985	<u>273/717/26/72477</u>	605/1547/32/69679
tern1	391/869/14	172/375/12/55	164/362/12/99	<u>159/358/12/133</u>	<u>159/359/12/83</u>
too_large	748/1743/24	449/1057/41/6545	390/971/45/15667	<u>374/941/32/13877</u>	<u>374/941/32/14179</u>
ttt2	215/486/9	144/332/9/10	142/332/9/15	144/332/9/13	<u>142/328/9/18</u>
vda	926/2226/14	617/1460/33/1226	574/1399/35/2332	<u>514/1274/55/6660</u>	<u>514/1274/55/6566</u>

(ゲート数 / 結線数 / 段数 / 計算時間 (秒))

* 太字は単独パターン利用より良い結果、下線は最良の結果を表す

表 2: ファンインの直列分割との併用手法

回路名	初期回路	Trans	基本手法	ボタン変化なし繰返し	ボタン変化付繰返し
9symml	168/402/14	152/376/14/6	136/350/12/10	<u>127/332/13/315</u>	<u>123/339/14/126</u>
C432	209/421/25	124/297/26/65	109/265/29/71	<u>108/261/23/575</u>	<u>107/259/30/504</u>
alu2	360/747/39	215/524/36/84	166/433/21/74	<u>158/412/19/462</u>	<u>161/424/21/488</u>
alu4	720/1441/42	455/1083/39/489	396/1042/31/747	<u>350/912/32/19572</u>	<u>361/964/38/6092</u>
apex7	268/501/17	216/418/21/12	208/417/17/16	212/414/20/295	<u>198/397/16/212</u>
b9	126/247/10	105/213/10/5	103/208/8/6	101/203/10/30	<u>99/200/10/29</u>
c8	188/371/7	138/274/7/9	121/241/9/10	121/244/8/44	<u>100/230/7/42</u>
cm162a	54/93/9	45/91/10/0	44/95/12/0	44/95/13/4	<u>39/85/10/2</u>
cm163a	55/92/9	47/87/9/0	45/91/9/0	46/92/9/3	<u>41/85/7/1</u>
cm82a	31/51/8	27/50/11/0	26/48/11/0	26/48/11/0	<u>22/42/10/0</u>
cm85a	42/82/12	32/71/12/0	32/71/12/0	32/71/12/1	<u>30/67/9/0</u>
cordic	105/197/13	64/122/11/1	63/121/11/1	63/121/11/6	<u>59/129/11/7</u>
example2	366/642/12	322/592/18/31	313/608/15/43	<u>312/605/16/237</u>	<u>311/592/15/152</u>
f51m	143/277/13	73/150/24/3	73/150/24/1	73/150/24/4	<u>63/136/16/8</u>
frg1	108/225/15	99/213/15/66	84/180/15/122	88/194/15/786	<u>68/160/13/792</u>
i2	175/407/12	161/381/12/8	161/381/12/52	161/381/12/475	<u>151/375/26/1946</u>
lal	148/301/9	97/187/8/2	97/187/8/2	97/187/8/8	<u>91/178/9/13</u>
mux	90/176/14	59/125/10/1	49/106/9/1	49/106/9/3	<u>41/97/7/5</u>
pcl	86/141/18	74/138/18/1	74/138/18/0	74/138/18/2	<u>66/134/16/2</u>
pcler8	103/174/18	90/169/15/1	90/169/15/1	90/169/15/5	<u>84/166/15/6</u>
sct	117/243/8	74/153/7/2	71/150/8/2	74/153/7/4	<u>65/130/7/6</u>
tern1	391/869/14	172/375/12/55	135/296/9/51	126/269/10/232	<u>118/272/11/186</u>
ttt2	215/486/9	144/332/9/10	119/272/9/13	<u>111/259/12/56</u>	<u>116/270/12/41</u>
vda	926/2226/14	617/1460/33/1226	468/1203/46/5269	<u>415/1072/53/27445</u>	<u>409/1054/53/13046</u>

(ゲート数 / 結線数 / 段数 / 計算時間 (秒))

* 太字は基本手法より良い結果、下線は最良の結果を表す