計算機アーキテクチャ 110－7
設 計 自 動 化 73－7
（1995. 1. 19）

ランダムベンチマーク例題による論理最適化システムの評価

日野 健介　岡野 宏哉　岩間 一雄

九州大学工学部

論理最適化システムを実験的に評価する時にベンチマーク回路の選び方は重要である．本論文では，初期回路 (関数) を基本的な属性を制御しながらランダムに生成し，さらにそれをランダムに等価変換することによってテスト例題を生成する手法について述べる．SIS とトランスダクション法に対する実験では興味深いデータを得た．

キーワード：ランダムベンチマーク，論理最適化アルゴリズム

Random Benchmarks for Evaluating Logic Optimizers

Kensuke Hino, Hiroya Okano and Kazuo Iwama

Department of Computer Science and Communication Engineering
Kyushu University

How to select benchmarks is an important issue in evaluating empirically the performance of logic optimizers. To generate test instances, we propose the random applications of transformation rules to an initial circuit(function), which is also generated at random with controllability of basic attributes. Experiments for SIS and Transduction using such random benchmarks provided interesting results.

Keywords: Random Benchmark, Logic Optimizers

1. はじめに

論理合成システムを含むCADシステムを実験的に評価するときに，ベンチマークの選び方が重要であることはいうまでもない．例えばMCNC[12]は70以上の組合わせ多段回路またはその他のタイプの回路を含む最も標準的なベンチマークである．これらの回路は様々な種類の中から注意深く選ばれており，論理設計の分野でも広く認められている．しかし，有限個の回路しか含まれてないことは事実であり，その一般性には疑問が残る．ベンチマークに対して良いシステムは全ての回路に対して良いということを証明することはできない．(ベンチマークに対して良いCADプログラムがそれ以外の回路に対して性能が悪くなることはまれではない.) またベンチマークに対する不正なチューンアップの疑いを完全に否定することはできない．

この問題を解決する方法として，ベンチマークの集合にランダムな例題を加えることが考えられる．ランダム例題を用いることはグラフアルゴリズム[11]や組合わせ問題[4]などの他の分野ではすでに一般的である．しかし，意味のあるランダム例題の生成は容易ではない．ランダム例題はあまりにも不自然で，実世界とはほど遠いという強い批判がある．そのような批判に対する答の一つは例題の特徴や属性をうまく制御することであるが，それは容易ではない．例えばランダムグラフの生成法と同様な手法，すなわち適当にゲートを配置しそれらの間を適当な確率で配線することによってランダム回路を生成することが考えられるが，回路が表す関数が恒真(偽)になってしまうことがしばしば生じる．

文献[3]において我々は回路をランダムに等価変換することによって，MIS[1]やSIS[9]，トランスダクション法[7]といった論理最適化システムを評価するためのランダム例題の生成法を提案した．生成のアルゴリズムは適当な初期回路に対してランダムに等価変換の列を適用するというものである．各変換は，変換ルールの集合からランダムに選ばれる．各ルールは等価な変換，すなわち回路が表す関数を変えることなく変換することができる．1回の等価変換は多項式時間で実行可能である．変換ルールの集合は完全である．つまり任意の回路から任意の等価な回路への変換が可能である．よって小さな初期回路からサイズの異なる多くの回路を生成でき，最適化システムのテスト例題として与えることができる．初期回路は最適化された“答”のヒントとなるというメリットがある．

この生成においては初期回路をいかに選ぶかが重要になる．例えば実用的な回路を集めたMCNCの中から選ぶことが考えられる．MCNC自体は有限であるが，変換系列を変えることにより多くの異なった回路を得ることができる．しかし限られた関数の回路しか生成されないので，我々のランダムベンチマーク生成の当初の目的である例題の多様性に欠けるのではないかと考えられる．

そこで例題生成システムの拡張として，初期回路のランダム生成を行なう．初期回路の生成は論理関数の生成と考えてよい．我々の手法は主項をランダムに生成することである．関数のオンセットの大きさと関数の複雑さの度合を含む基本的な属性を制御できる．これにより初期回路を選んだりそれを計算機へ入力する面倒な手間がなくなり，10個ほどの簡単なパラメータで様々なランダム回路を生成できるようになった．

まだ初期の段階ではあるが，実験の結果いくつかの興味深いデータを得ることができた．我々はSISとトランスダクション法を大別して2種類の回路(例題)でテストした．一つはMCNCから初期回路を選んできてそれをランダムに等価変換した例題(MCNC-RT)であり，もう一つはランダム初期回路をランダムに等価変換した例題(R-RT)である．結果はおおよそ以下のようになった．(1)R-RT例題はSIS，トランスダクション法のいずれに対してもMCNC-RT例題よりも難しいことが分かった．MCNC-RT例題に対する最適化システムの出力の簡単化の度合はほとんど最良のサイズと変わらないが，R-RTの場合は，最良の回路のサイズより3倍ほど悪くなっている．(2)トランスダクション法は，簡単化の度合に関してSISよりも良い結果を出すことがしばしばあった．(3)SISやトランスダクション法で簡単化できない例題がいくつかあった．(SISに対しては約8%，トランスダクション法では約10%)．

2節では簡単に文献[3]のランダム変換について述べる．3節では新しく開発した初期回路のランダム生成について述べる．実験結果は4節にその考察とともに述べる．

2. ランダム等価変換[3]

現在のところ生成できる回路はファンイン，ファンアウト制限のないNANDゲートのみからなる組合せ回路である．例えばAND, OR, NOT[2]などの他の種類のゲートおよびファンイン，ファンアウト制限のある回路については今後の課題である．回路は次のような式の集合で表される．

$$g[0] = (g[1], (g[10], (x_3)))$$
$$g[1] = (g[10])$$
$$g[10] = (x_1, (x_2), x_4)$$

これは，図1の回路を表す．

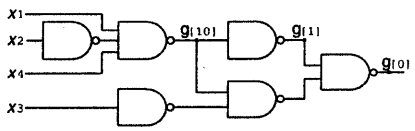

図1

変換ルール(単にルールともいう)は$f \Longrightarrow g$の形で表される．ルールの集合$\Re$は以下に示すものである．

(1) $(0) \Longleftrightarrow 1$　(2) $(x,x) \Longleftrightarrow (x)$
(3) $(x,(x)) \Longleftrightarrow 1$　(4) $x,((y,z)) \Longleftrightarrow x,y,z$
(5) $x,y \Longleftrightarrow y,x$　(6) $(x,1) \Longleftrightarrow (x)$
(7) $((x)) \Longleftrightarrow x$
(8) $(x,(y,z)) \Longleftrightarrow (((x,(y)),(x,(z))))$
(9) $g[\ell] = f$という部分式の定義が存在するとき，$g[\ell] \Longleftrightarrow f$.
(10) ラベル$g[\ell]$が出力でないかまたは部分式の定義の右辺に現れないとき，$g[\ell]$の定義を回路を表現する部分式の集合から除去できる．これを消去とよぶ．
(11) ラベル$g[\ell]$の部分式の定義が存在しないとき，$g[\ell]$= 部分式の形の定義を集合に付加できる．ただしそのサイズ(長さ)は，現在の式のサイズの多項式で制限される．これを生成とよぶ．

$f \Longleftrightarrow g$は，$f \Longrightarrow g$，$f \Longleftarrow g$の両方向の変換が可能であることを意味する．また各x,y,zは定義を満たす任意の部分式でなければならない．$\Re$のルールを適用することにより，回路C_1は等価な回路C_2に変換される．以下に変換の例を示す．

例：変換ルール

$$x,((y,z)) \Longrightarrow x,y,z$$

を回路 C_1

$$g[0]=((x_1),((x_2,(g[1],((x_4,x_1))))),x_3)$$

に適用する．x,y,z は任意の部分式であれば良いので，

$$x=(x_1),\ y=x_2,\ z=(g[1],((x_4,x_1)))$$

とすることによりもとの回路 C_1 から回路 C_2

$$g[0]=((x_1),x_2,(g[1],((x_4,x_1))),x_3)$$

に変換される．また別の可能性として，

$$x=g[1],\ y=x_4,\ z=x_1,$$

ととることもできるので，そのとき回路 C_2 は

$$g[0]=((x_1),((x_2,(g[1],x_4,x_1))),x_3)$$

となる．

次に変換ルールの集合 $\Re$ に関する定理を示す．

定理1： サイズ n の回路 C に対して，$\Re$ のルールの1回の適用は n の多項式時間で実行可能である．C に適用できる部分がないとき，その判定も多項式時間で可能である．

定理2： 回路 C_1 と C_2 が等価であるとする．このとき C_1 から C_2 へ変換するための $\Re$ に含まれる変換ルールの列が存在する．

定理2は与えられた回路からより簡単なあるいは最適な回路への変換の道が存在することを保証している．しかしそのような最適な回路への変換の列を得ることあるいはその列の長さについてまでは言及していない．よってこのルールの集合を回路の最適化のために使うことは現実的ではない．一方，ランダムに適用することによってより複雑な回路へ自然に変換されていくと考えられる．我々の提案する例題生成のアルゴリズムの基本的な流れを以下に示す．

ランダム変換アルゴリズム RT.

入力： 初期回路 C_1.

出力： C_1 と等価な回路 C_2(おそらく C_1 より 複雑である).

Step1: $C \leftarrow C_1$

Step2: $\Re$ からランダムにルール r を選ぶ．

Step3: C に r を適用して C' を得る．2ヶ所以上適用可能な場合はいずれか1ヶ所をランダムに選ぶ．

Step4: $C \leftarrow C'$ として，Step2 から Step4 を規定回数になるかまたは C が規定のサイズになるまで繰り返す．

Step5: $C_2 \leftarrow C$

Step2 においてルールによっては選び方の頻度を多少変えてある．(例えば $x,((y,z)) \Longrightarrow x,y,z$ は他のルールの10倍の頻度で適用した.) この頻度の設定は重要であり最終的に生成される回路の特徴を決定する．適用の頻度をさらに綿密に設定したりあるいは特定のルールの後にいくつかのルールを多く適用するなどの条件を導入することによって生成例題の制御が期待できる．例えばこの改良によって相対的に段数の深い回路などが生成できることが望ましい．

3. 初期回路のランダム生成

3.1 基本方針

ランダム変換アルゴリズム RT が同じ論理関数を表す多くの異なった (理論的にはすべての) 回路を生成できることを思い出されたい．よって初期回路の役割は論理関数を決定することであると考えられる．(以下では初期回路を初期関数とよぶこともある.) またランダム生成はしばしば人工的なあるいは実際に意味のない例題を生成してしまい，例題の特徴や属性を制御する能力に欠けていることもすでに指摘した．

論理関数に関してどのような属性が考えられるだろうか．変数の数 n のような明らかなもののほかに，以下のような2つの属性が重要となるであろう．

(1) オンセットのサイズ，すなわち 2^n 通りの変数への割り当てのうち関数値が 1(真) になる割り当ての数．

(2) 関数を実現する最適な回路の段数や大きさ，つまり関数の複雑さ．

論理関数を指定する基本的な方法として真理値表を用いることが考えられる．ランダムな真理値表は 2^n 通りの割り当てに対してランダムに0か1を割り当てることで容易に生成できる．明らかにオンセットのサイズの制御は容易である．しかし前述の属性 (2) を制御することは困難である．よく知られているように [10] 統計的にはほとんどすべての論理関数が指数サイズの回路を必要とする．それゆえにそのような真理値表を実現する回路のサイズはそれが最適であっても高い確率で非常に大きくなる．これは "人工的な性質" の典型例であり重大な欠点である．

1節で述べたように，回路をランダムグラフのように生成するのも良い方法ではない．この方法では属性 (1) と (2) いずれにも不適である．たとえゲートの数や結線の数を増やしても，すなわち回路の複雑さに寄与すると考えられるものを増しても，実質的に関数の複雑さを増すことにはつながらないであろうし，またオンセットのサイズには関係ない．

我々のアルゴリズムは例えば $x_1\overline{x_2}x_3+x_2x_4\overline{x_7}x_8+x_1x_5+\cdots$ のような DNF 式のランダム生成に基づいている．この方法は明らかに前述のランダムグラフの方法よりも関数の複雑さを制御し易い．その理由として，より多くのランダム項を生成した場合，仮にそれらの項のほとんどが主項であるならば，関数の複雑さを増す傾向にあると考えられるためである．(もし2段回路で関数を実現しようとするならばより多くのゲートを必要とする．我々が現在扱っている多段回路の場合いくつかの例外がある．例えばパリティ関数は，2^{n-1} の非常にたくさんの主項を必要とするが，小さな多段回路が存在する．しかしこの種の例外はあまり深刻に考えなくてもよいであろう.) よって原理的には多くの項を生成することによってより複雑な関数を得ることができる．ではどのようにしてオンセットのサイズを制御すればよいのだろうか．これはそれほど易しくない．

3.2 アルゴリズム

まず，いくつかの定義を行なう．**リテラル**とは変数 x に対して x またはその否定 $\overline{x}$ である．**項**とは $x_1\overline{x_2}x_3$ のようなリテラルの積である．項は冗長でない，すなわち2つ以上の同じリテラルを含まない，または変数 x に対して x と $\overline{x}$ の両方を含まないと仮定する．(DNF) 式は項の和である．**セル**とは例えば4変数関数の場合 x_1=x_3=x_4=1, x_2=0 のような特定の割り当てである．項 C がセルの表す割り当てに対して1(真) となるとき，セルは C で**覆われている**という．例えば項 $x_1\overline{x_2}$ は，前述の例のセルを覆っている．多くのセルを覆っている項は**大きい**という．明らかに大きな項は少ないリテラルからなる．k 個のリテラルを含む項を k-**項**という．

我々の目標は項$C_1, C_2, \cdots$をこの順序で生成し，最終的に以下のようなn変数の式を得ることである．

$$f = C_1 + C_2 + \cdots + C_m + C_{m+1} + \cdots + C_{m+t}$$

各C_iは少なくともC_1からC_{i-1}までのどの項にも覆われていない新しいセルを覆わなければならない．C_1からC_mはk_1-項である．すなわちm個のすべての項は同じサイズである．これらの項を1次項(*primary term*)とよぶ．C_{m+1}からC_{m+t}までの残りのt項は2次項(*secondary term*)とよばれ，k_2からk_3の範囲でランダムに決める．パラメータk_1, k_2, k_3, tは入力として与えられる．別のパラメータX(1から99までの整数)はオンセットのサイズを決める．式fは次の条件を満たさねばならない．部分式$f_{m-1} = C_1 + C_2 + \cdots + C_{m-1}$は全体で$2^n$個あるセルのうちの$X$%以下を覆い，$f_m = C_1 + \cdots + C_m$は少なくとも$X$%のセルを覆っている．すなわちオンセットのサイズが初めて少なくともX%になったときにk_1-項の生成を停止する．よって例えばX=5, k_1=3は，最初の1個の3-項で全体の12.5%のセルを覆ってしまうので意味のない値である．

関数の複雑さを増すためにC_{m+1}からC_{m+t}までの2次項を加える．オンセットのサイズXとの誤差を小さくするために2次項は小さくなければならない，すなわちk_2とk_3は大きくなければならない．20変数の式をX=50で生成するとき，各パラメータの値は例えばk_1 =3, k_2=15, k_3=20ぐらいにとるとよい．k_2とk_3は大きい(各C_{m+i}は無視していいほど小さい)のでtの値はオンセットのサイズの増加とは関係なくとることができる．以下に生成アルゴリズムを示す．

ランダム初期回路ジェネレータ RIC-GEN

入力： 変数の数n, 上述のk_1, k_2, k_3, t, X

出力： DNF式$C_1 + \cdots + C_{m+t}$

Step 1: $P \leftarrow 0, i \leftarrow 1, j \leftarrow 1$.

Step 2: ランダムなk_1-項Cを生成し，Cで覆われる新しい(C_1からC_{i-1}までに覆われていなかった)セルの数hを計算する．

Step 3: $h > 0$ならば$C_i \leftarrow C$, $P \leftarrow P + h$, $i \leftarrow i + 1$とする．$h \leq 0$すなわちCが新しいセルを覆わないならばStep2へ戻る．

Step 4: $P < (X/100) \cdot 2^n$ならばStep2へ戻る．

Step 5: k ($k_2 \leq k \leq k_3$)をランダムに選び，k-項のCをランダムに生成する．

Step 6: Cが新しいセルを覆っているならば$C_i \leftarrow C$, $i \leftarrow i + 1$とする．そうでないならばStep5へ戻る．

Step 7: $j = t$ならば終了．そうでないならば$j \leftarrow j + 1$としてStep5へ戻る．

Step2(Step6も同様)のhの計算はみかけよりも難しい．例えばこれまでに覆われた全てのセルをメモリーに蓄える方法がある．しかし2^nのメモリー空間が必要となり計算も遅い．我々の手法は，文献[5]で用いている包除原理とバックトラックを発展させた文献[6]の手法を組み合せたものである．

例えば3つの項C_1, C_2, C_3があるとする．それらの項で覆われているセルの数は次のように計算できる．

$$U(\{C_1\}) + U(\{C_2\}) + U(\{C_3\}) - U(\{C_1, C_2\})$$
$$-U(\{C_2, C_3\}) - U(\{C_1, C_3\}) + U(\{C_1, C_2, C_3\})$$

ここで項の集合Sに対して，$U(S)$はSのすべての項によって覆われているセルの数である．TをSの中の項に存在する異なったリテラルの集合とすると，$U(S)$は次のようになる．

$$U(S) = \begin{cases} 0 & T \text{が} x \text{と} \overline{x} \text{の両方を含む} \\ 2^{n-|T|} & \text{それ以外} \end{cases}$$

例えばn=4, C_1=x_1x_2, C_2=$x_1\overline{x_3}x_4$, C_3=$\overline{x_2}x_4$とする．このとき，$U(\{C_1\})$=2^{4-2}=2^2, $U(\{C_1, C_2\}) = 2^{4-4} = 1$, $U(\{C_1, C_3\})$=0となる．

各$U(S)$はビット命令を使って高速に計算できる．問題は$U(S)$を計算する順序である．Sが多くの項を含むとき$U(S)$はほとんど0となる．よって$S' \supseteq S$であるようなS'に対して$U(S)$=0ならば$U(S')$=0となる．これは深さ優先探索やバックトラック法の研究でよく知られている．詳しくは述べないが，少ないメモリーでかなり高速に実行できる．

4. 計算機実験

4.1 例題生成

全ての実験はSUN Sparc Station10上で行なった．約200個のR-RT(ランダム初期回路＋ランダム変換)例題を生成した．各パラメータは以下のとおりである．

(1) 変数の数： 10, 20.

(2) オンセットのサイズ： 1, 5, 50, 95, 99(%). 他に，1つの最小項のみ(2^nのセルの中から1つを選んだもの)からなる初期回路も生成した．

(3) 追加する2次項の数： 0, 10 (オンセットが50%の場合のみ).

例えば10-.50-0は10変数，オンセットのサイズ50%，追加する2次項は0の初期回路を示す．1次項の数を例えば10程度にするためにはk_1=4すなわち各1次項に4リテラル含ませるようにすればよい．他のサイズについては99%, 95%, 5%, 1%のときそれぞれk_1=3, 3, 9, 10とした．1次項の数はおおよそ20～30となる．2次項はオンセットのサイズが50%の場合にのみ10個加えることにした．

計算時間は生成された項の数に依存する．すなわちオンセットのサイズが大きくなるほど遅くなる．99%の場合10変数で約10sec(CPU time)，20変数の場合は10^2secほどかかった．ほとんどの時間はStep2に費やされる．計算時間は95%の場合には約1/10になる．50%またはそれ以下の場合は計算時間はほとんど無視できる．2次項を加えるための時間もかかる．例えば20変数95%の場合，加える項が0のときは約10secであるが，10個の項を追加すると120secかかる(k_2=15, k_3=20)．オンセットのサイズが大きいので，多くの2次項がStep6において新しいセルを覆うことができずに生成に失敗するためである．各パラメータに対して3つの初期関数を生成した．例えば10-.50-0に対しては10-.50-0(1), 10-.50-0(2), 10-.50-0(3)と表す．

それらの初期関数(回路)に対してランダム変換を行ない，3つの異なる回路を生成した．(初期関数10-.50-0(1)に対してそれぞれ10-.50-0(1,1), 10-.50-0(1,2), 10-.50-0(1,3)と表す.) これが最終的なテスト回路であり，別のグループで開発されたプログラムで初期回路と等価であることを確認している．ランダム変換は1000ゲートになるかまたは計算時間が10^3secになるまで実行した．

また，MCNC-RT例題の生成も行なった．初期回路は9symml, cm82a, z4ml, frg1, cordicを選び，各初期回路に対して10個の例題を生成した．ルールの適用は2000回行なった．

4.2 SIS とトランスダクション法に対する実験

SIS は script.algebraic を用いて最適化を行ない，面積優先でマッピングした．トランスダクション法は京都大学上林研究室で開発されたプログラムを用いた．SIS，トランスダクション法ともにライブラリは 4 入力までの NAND と NOR を使用した．表 1 は 10-.50-0(2,1), 10-.50-0(2,2), 10-.50-0(2,3) に対する SIS とトランスダクション法の結果である．1 行目は初期回路 10-.50-0(2) を簡単化した結果である．下の 3 つの行は上記の 3 つのテスト例題についての結果である．例えばテスト回路 10-.50-0(2,1) はゲート数 1000, 結線数 2161, 段数 23 であり，これが SIS によってそれぞれ 45, 91, 11 に簡単化され，またトランスダクション法によって 38, 87, 9 に簡単化されたことを示す．最適な回路は初期回路を簡単化したときのデータであるゲート数 26, 結線数 61, 段数 5 に近いものであると仮定する．(我々は特にゲート数に注目した.) この仮定のもとで 10-.50-0(2,1) に対して SIS によるゲート数の簡単化の度合は 45/26=1.73，トランスダクション法では 38/26=1.46 と計算することにする．結線数，段数についても同様である．第 4 列目は秒単位での CPU time を表す．

表 2 から 10 についても同様である．簡単化の度合は時々大きくなっていることがある．例えば表 5 では 4.5 というのがある．表 2 において，トランスダクション法は SIS よりもかなり悪くなっている．トランスダクション法はどちらかといえば段数を優先するシステムである．表 9 は初期回路 10-s-0(1)(オンセットが 1 個のセルのみからなる) についての結果である．表 4 の n.a. はシステムが簡単化できなかった (Segmentation Fault または 5000sec 以上かかる) ことを意味する．

表 11 は今回行なった実験をまとめたものである．最初の 5 行は MCNC-RT 例題に対しての結果であり，残りは R-RT 例題に対してのものである．各行は各初期関数に対して 3 つ (SIS またはトランスダクション法が失敗した場合 2 つ) の例題の平均値をとったものである．deg は簡単化の度合，t/s は (トランスダクション法の deg÷SIS の deg)，div は (3 つのうちの最大の deg÷ 最小の deg) である．この結果から，次のようなことが言える．

(1) MCNC-RT 例題に対する簡単化の度合はほとんどが 2 以下である．R-RT 例題では 2 から 4 の間となっている．ゆえに R-RT 例題は MCNC-RT 例題よりも難しいのではないかと考えられる．

(2) t/s はほとんどが 1 以下となっている．つまり，トランスダクション法は SIS よりも良い結果を出している．しかし SIS の方が実行時間は短い．

(3) 簡単化の度合は初期関数の違いに関してはほとんど差がない．例えば 10-.50-10 は 10-.50-0 よりも複雑であると考えられるが，deg の値はほとんど同じである．しかし計算時間は前者の方が明らかに遅くなっている．

(4) 最後に，計算時間に大きな差があることに注意されたい．同じ初期関数から生成した回路でも div 値が 100 倍も違うことが時々ある．

5. おわりに

固定されたベンチマーク集合 (多くは MCNC) を用いた議論では，ふつう「あるシステムが別のシステムより平均 8.3% 良い」などのように行なわれる．我々の実験によればランダム回路を用いるとこの差が 400% になることがある．さらにその差は MCNC-RT よりも R-RT の方が大きくなる．ゆえに最適化システムに対してランダムベンチマークは固定されたベンチマークよりも難しい，あるいは手に負えないといえるのではなかろうか．

今後の方向としては (1) ファンイン，ファンアウトの制限，(2) ルールの優先度の検討といったさらに多様な例題を生成するためのシステムの開発とさらに多くの実験を行なうことを考えている．

参考文献

[1] R. K. Brayton, R. Rudell, A. L. Sangiovanni-Vincentelli, and A. R. Wang, "Mis: A multiple-level logic optimization system," *IEEE Trans.* CAD, 6, pp. 1062-1081, 1987.

[2] K. Hino and K. Iwama,"On a complete set of basic operations to transform between equivalent switching circuit," Technical Report of the Institute of Electronics, Information and Communication Engineers, COMP92-67, 1992.

[3] K. Iwama and K. Hino,"Random Generation of Test Instances for Logic Optimizers," 31st *ACM/IEEE* Design Automation Conference, pp. 430-434, 1994.

[4] K. Iwama, H. Abeta, and E. Miyano, "Random generation of satisfiable and unsatisfiable CNF predicates," *in* Proc. 12th IFIP World Computer Congress, pp. 322-328, 1992.

[5] K. Iwama, "CNF Satisfiability Test by Counting and Polynomial Average Time," SIAM J. Computing, 18,2, 385-391, 1989.

[6] O. Kanmoto and K. Iwama, "On Improvement of the Satisfiability Test by Counting using Backtracking," Record of Joint Conference of Electrical and Electronics Engineers in Kyushu, 1505, Kumamoto, 1994.

[7] S. Muroga, Y. Kambayashi, H. C. Lai, and J. N. Culliney, "The Transduction method – Design of logic networks based on permissible functions," *IEEE Trans. Comput.* 38, 10, 1989.

[8] D. Mitchell, B. Selman, and H. Levesque, "Hard and easy distributions of SAT problems," *in* Proc. 10th National Conference on Artificial Intelligence, pp. 459-465, 1992.

[9] E. M. Sentovich, K. J. Singh, *et al.*, "SIS: A system for sequential circuit synthesis," Memorandum No. UCB/ERL M92/41, 1992.

[10] J. Savage, *The complexity of Computing*, Wiley, New York, 1976.

[11] G. Tinhofer, "Generating graphs uniformly at random," *in* Computational graph theory, pp. 235-255, Springer, 1990.

[12] S. Yang, "Logic synthesis and optimization Benchmarks user guide version 3.0," *in* 1991 MCNC International Workshop on Logic Synthesis.

Circuit		gate	conn.	level	time
Initial	input	20	59	3	–
	sis	28	60	7	2.1
	trans	26	61	5	1.5
No.1	input	1000	2161	23	–
	sis	45	91	11	16.7
	trans	38	87	9	134.4
No.2	input	1188	2236	22	–
	sis	124	258	19	28.1
	trans	42	98	11	139.9
No.3	input	1114	2405	15	–
	sis	71	144	21	24.7
	trans	44	106	11	100.1

表 1: 10-.50-0(2)

Circuit		gate	conn.	level	time
Initial	input	38	271	4	–
	sis	83	189	12	14.4
	trans	148	320	10	5.1
No.1	input	1154	3235	19	–
	sis	309	691	22	116.3
	trans	202	451	22	169.4
No.2	input	1041	2883	16	–
	sis	188	434	18	50.0
	trans	179	421	14	133.2
No.3	input	1053	3586	14	–
	sis	192	443	16	86.9
	trans	207	475	16	190.7

表 2: 10-.05-0(2)

Circuit		gate	conn.	level	time
Initial	input	31	90	3	–
	sis	29	57	6	2.3
	trans	43	88	7	1.2
No.1	input	1416	2955	19	–
	sis	99	200	19	26.0
	trans	50	112	13	251.1
No.2	input	1077	2160	16	–
	sis	131	269	17	25.5
	trans	57	121	10	60.6
No.3	input	857	1630	17	–
	sis	53	112	10	11.2
	trans	61	132	11	50.9

表 3: 10-.95-0(1)

Circuit		gate	conn.	level	time
Initial	input	43	273	4	–
	sis	436	996	15	135.8
	trans	164	372	10	23.2
No.1	input	1408	4422	18	–
	sis	n.a.	n.a.	n.a.	
	trans	213	477	22	408.8
No.2	input	1119	4618	16	–
	sis	424	940	21	2053.6
	trans	230	511	18	285.4
No.3	input	1030	3766	14	–
	sis	n.a.	n.a.	n.a.	
	trans	225	498	16	223.1

表 4: 20-.01-0(2)

Circuit		gate	conn.	level	time
Initial	input	27	70	3	–
	sis	27	61	6	2.0
	trans	32	69	5	2.5
No.1	input	1007	2378	24	–
	sis	79	162	25	17.8
	trans	55	127	13	169.1
No.2	input	1278	2439	21	–
	sis	119	236	17	26.1
	trans	56	124	11	106.6
No.3	input	1058	2181	19	–
	sis	100	207	19	19.9
	trans	40	90	9	170.7

表 5: 20-.50-0(1)

Circuit		gate	conn.	level	time
Initial	input	39	104	3	–
	sis	55	107	8	4.8
	trans	53	117	7	14.3
No.1	input	1113	2190	23	–
	sis	134	270	17	28.8
	trans	88	198	15	221.2
No.2	input	1776	3364	21	–
	sis	78	154	13	39.5
	trans	82	180	17	564.2
No.3	input	1053	1940	19	–
	sis	113	219	19	25.8
	trans	76	165	17	198.8

表 6: 20-.95-0(2)

Circuit		gate	conn.	level	time
Initial	input	32	175	3	–
	sis	59	135	11	4.5
	trans	84	201	9	3.0
No.1	input	1013	2813	19	–
	sis	98	227	19	22.2
	trans	123	285	13	99.8
No.2	input	1020	2657	17	–
	sis	133	300	15	22.2
	trans	125	289	15	87.1
No.3	input	1554	5702	14	–
	sis	115	255	19	39.9
	trans	115	267	13	1130.0

表 7: 10-.50-10(2)

Circuit		gate	conn.	level	time
Initial	input	43	259	3	–
	sis	86	212	12	9.4
	trans	138	332	8	31.4
No.1	input	1006	5991	7	–
	sis	150	352	16	84.9
	trans	171	399	17	738.8
No.2	input	1038	5310	15	–
	sis	172	407	19	1630.4
	trans	197	459	16	745.4
No.3	input	1036	2906	15	–
	sis	97	240	15	30.7
	trans	207	468	16	241.6

表 8: 20-.50-10(3)

Circuit		gate	conn.	level	time
Initial	input	10	29	2	–
	sis	13	32	4	1.2
	trans	22	41	5	0.9
No.1	input	468	906	18	–
	sis	13	32	4	5.1
	trans	26	46	8	16.3
No.2	input	1232	2541	24	–
	sis	13	32	4	11.7
	trans	26	46	10	325.5
No.3	input	1296	2491	24	–
	sis	13	32	4	13.8
	trans	24	45	6	495.1

表 9: 20-s-0(1)

Circuit		gate	conn.	level	time
Initial	input	6	15	2	–
	sis	6	15	3	0.9
	trans	12	21	4	0.8
No.1	input	2056	3908	25	–
	sis	6	15	3	18.5
	trans	16	25	8	1253.1
No.2	input	1409	2519	26	–
	sis	6	15	3	13.0
	trans	16	28	8	331.3
No.3	input	1110	2290	17	–
	sis	6	15	3	11.4
	trans	14	23	6	443.8

表 10: 10-s-0(1)

Initial Circuit		gate			conn.			level			time		
		deg.	t/s	diver.	deg.	t/s	diver.	deg.	t/s	diver.	av.	t/s	diver.
9symml	sis	2.03		1.84	1.73		1.76	1.71		1.32	63.9		2.57
	trans	1.26	0.62	1.11	1.26	0.73	1.10	1.35	0.79	1.53	89.9	1.41	2.92
cm82a	sis	1.08		1.31	1.03		1.35	1.73		1.67	10.0		2.46
	trans	0.95	0.88	1.33	0.98	0.95	1.23	1.27	0.73	1.67	71.4	7.13	23.92
z4ml	sis	2.90		2.31	2.86		3.04	2.30		2.18	23.5		2.65
	trans	1.02	0.35	1.32	1.02	0.36	1.21	1.18	0.51	1.25	96.5	4.10	18.01
frg1	sis	1.71		1.70	1.72		1.65	1.24		1.35	37.0		2.88
	trans	1.23	0.72	1.27	1.41	0.82	1.35	1.14	0.92	1.24	3201.0	86.63	2.55
cordic	sis	1.88		1.47	1.88		1.47	1.72		1.93	26.1		2.51
	trans	1.29	0.69	1.65	1.33	0.71	1.78	1.35	0.78	1.50	139.2	5.32	27.29
10-.01-0(1)	sis	2.87		2.86	2.76		2.73	1.76		1.71	37.1		1.44
	trans	2.10	0.73	1.31	1.94	0.70	1.26	1.33	0.76	1.50	360.5	9.71	4.13
10-.05-0(1)	sis	3.83		2.84	3.64		2.70	1.97		1.58	660.3		26.96
	trans	2.28	0.60	1.06	2.29	0.63	1.04	1.50	0.76	1.38	145.6	0.22	1.13
10-.05-0(2)	sis	2.77		1.64	2.76		1.59	1.56		1.38	84.4		2.33
	trans	2.36	0.85	1.13	2.38	0.86	1.13	1.44	0.92	1.57	164.4	1.95	1.43

表 11: 実験結果のまとめ

Initial Circuit		gate			conn.			level			time		
		deg.	t/s	diver.	deg.	t/s	diver.	deg.	t/s	diver.	av.	t/s	diver.
10-.50-0(1)	sis	2.26		2.05	2.01		2.11	3.00		2.38	17.3		1.61
	trans	1.41	0.62	1.03	1.42	0.76	1.04	2.07	0.69	1.22	144.4	8.33	2.99
10-.50-0(2)	sis	3.08		2.76	2.69		2.84	3.40		1.91	23.2		1.68
	trans	1.59	0.52	1.16	1.59	0.59	1.22	2.07	0.61	1.22	124.8	5.39	1.40
10-.95-0(1)	sis	3.25		2.47	3.40		2.40	2.56		1.90	20.9		2.32
	trans	1.93	0.59	1.22	2.13	0.63	1.18	1.89	0.74	1.30	120.9	5.78	4.93
10-.95-0(2)	sis	3.47		3.37	3.11		3.41	3.20		1.91	26.4		1.70
	trans	1.74	0.50	1.15	1.67	0.54	1.14	2.80	0.88	1.15	102.0	3.87	1.31
10-.99-0(1)	sis	7.33		1.89	8.00		1.92	2.60		1.89	20.7		1.39
	trans	1.71	0.23	1.05	1.67	0.21	1.03	1.70	0.65	1.43	106.75	5.16	1.84
10-.99-0(2)	sis	4.80		1.61	5.59		1.57	3.50		1.47	44.2		2.39
	trans	1.88	0.39	1.14	2.10	0.38	1.12	1.58	0.45	1.11	301.1	6.81	6.72
20-.01-0(1)	sis	1.09		1.09	1.13		1.07	1.80		1.25	74.8		1.71
	trans	1.47	1.35	1.04	1.40	1.24	1.06	2.00	1.11	1.00	331.7	4.43	1.11
20-.50-0(1)	sis	3.68		1.51	3.31		1.46	3.39		1.47	21.3		1.47
	trans	1.86	0.51	1.40	1.86	0.56	1.41	1.83	0.54	1.44	148.8	7.07	1.60
20-.50-0(2)	sis	2.58		1.44	2.34		1.44	3.53		1.77	23.5		2.33
	trans	1.48	0.57	1.25	1.58	0.68	1.24	2.20	0.62	1.44	503.1	21.41	10.43
20-.50-0(3)	sis	3.80		1.89	3.21		1.94	3.93		1.24	35.2		2.76
	trans	1.61	0.42	1.09	1.49	0.46	1.13	2.33	0.59	1.44	573.5	16.31	14.01
20-.95-0(1)	sis	1.73		1.25	1.61		1.25	2.24		1.27	100.2		5.12
	trans	1.59	0.92	1.11	1.58	0.98	1.16	2.14	0.96	1.00	397.8	3.97	1.59
20-.95-0(2)	sis	2.04		1.72	1.83		1.75	2.33		1.46	31.4		1.53
	trans	1.55	0.76	1.16	1.55	0.85	1.20	2.33	1.00	1.13	328.1	10.46	2.84
20-.95-0(3)	sis	1.93		1.53	1.73		1.42	2.29		1.67	24.07		1.45
	trans	1.68	0.87	1.14	1.62	0.94	1.16	1.81	0.79	1.08	180.8	7.51	1.83
20-.99-0(1)	sis	1.51		1.55	1.51		1.52	1.71		1.56	20.2		1.27
	trans	1.68	1.12	1.11	1.89	1.25	1.16	1.95	1.14	1.15	264.7	13.1	1.31
20-.99-0(2)	sis	1.36		1.66	1.39		1.63	1.86		2.33	18.0		1.52
	trans	1.49	1.10	1.14	1.66	1.20	1.19	1.67	0.90	1.18	191.2	10.65	1.76
20-.99-0(3)	sis	1.68		1.22	1.70		1.21	1.63		1.36	20.4		1.18
	trans	1.73	1.03	1.08	1.85	1.09	1.09	1.75	1.07	1.15	133.0	6.53	1.03
10-.50-10(1)	sis	3.04		3.08	3.00		2.84	2.10		2.82	91.0		7.34
	trans	2.14	0.71	1.04	2.29	0.76	1.04	1.80	0.86	1.40	5191.9	57.09	93.06
10-.50-10(2)	sis	1.95		1.36	1.93		1.32	1.61		1.27	28.1		1.80
	trans	2.05	1.05	1.09	2.08	1.08	1.08	1.24	0.77	1.15	439.0	15.63	12.97
10-.50-10(3)	sis	2.46		1.42	2.40		1.37	1.79		1.53	28.5		1.37
	trans	2.03	0.83	1.09	2.15	0.90	1.09	1.30	0.73	1.73	137.4	4.87	2.13
20-.50-10(1)	sis	1.67		1.09	1.62		1.17	1.07		1.13	103.7		3.93
	trans	2.08	1.25	1.14	1.98	1.22	1.16	1.07	1.00	1.20	614.0	5.92	2.57
20-.50-10(2)	sis	2.29		1.12	2.15		1.05	1.36		1.12	63.8		1.38
	trans	2.65	1.16	1.10	2.40	1.12	1.12	1.38	1.02	1.31	381.6	5.98	2.62
20-.50-10(3)	sis	1.62		1.77	1.57		1.70	1.39		1.27	582.0		53.11
	trans	2.23	1.38	1.21	2.08	1.33	1.17	1.36	0.98	1.06	575.3	0.99	3.09
10-s-0(1)	sis	1.00		1.00	1.00		1.00	1.00		1.00	14.3		1.62
	trans	2.56	2.56	1.14	1.69	1.69	1.22	2.44	2.44	1.33	676.1	47.28	3.78
20-s-0(1)	sis	1.00		1.00	1.00		1.00	1.00		1.00	10.2		2.71
	trans	1.95	1.95	1.08	1.43	1.43	1.02	2.00	2.00	1.67	279.0	27.35	30.37

表 11: 実験結果のまとめ (つづき)