ISSN 2186-5647

2-24

必須割当て情報を用いたテストパターン数の下界算出法

日大生産工　○曽根　隆暢　　日大生産工　細川　利典
京産大理工　吉村　正義　　日大生産工　新井　雅之

1. はじめに

近年，半導体微細化技術の進歩に伴って，超大規模集積回路（Very Large Scale Integrated circuits：VLSI）の大規模化・複雑化が急速に進展している．これに伴い，テスタでテストする際のテスト実行時間と，テスタに記憶させるテストパターンのデータ量が増加し，テストコストが増大するという問題が発生している[1]．テストコストはテストパターン数と比例関係にあるため，テスト品質を落とさずにテストパターン数を削減するテスト圧縮[2]を用いることにより，テストコストの削減が期待できる．

小規模回路では，ほぼ最小のテスト集合を得ることができる静的圧縮と動的圧縮を適用したアルゴリズム[2]が提案されている．しかしながら，大規模回路では最小のテスト集合を得るための計算量が大きく，現実的な計算時間での適用は困難である．

大規模回路に適用可能な手法として，文献[3]では，自動テストパターン生成（Automatic Test Pattern Generation：ATPG）が生成した初期テスト集合に対しドントケア判定[4]，頂点彩色問題を用いた極小解圧縮[5][6]，2 重検出法[2]を繰り返し行う静的圧縮手法が提案されている．しかしながら，文献[3]の手法では，初期テスト集合に対して静的圧縮を行うため，テスト圧縮効率が初期テスト集合に依存する．したがって，初期テスト集合中にテスト圧縮に非効率的なテストパターンが存在したとき，テストパターン数削減の妨げになる可能性がある．よって，初期テスト集合中のテスト圧縮に非効率的なテストパターンを削除し，テスト圧縮に効率的なテストパターンを再生成する，またはテスト圧縮に効率的な初期テスト集合を生成することにより，最終テスト集合のテストパターン数を削減できると考えられる．しかしながら，テスト圧縮の研究を行う上で，まず評価の対象となるベンチマーク回路のテストパターン数の下界を明らかにし，定量的評価の目標として設定する必要がある．

本論文では，組合せ回路における単一縮退故障[7]を対象として，必須割当て情報[9]を用いて同時検出不可能グラフを生成し，独立故障集合生成[8]を行い，最大独立故障集合サイズを計算し，テストパターン数の下界を求める手法を提案する．

本論文は以下のように構成されている．第 2 章では必須割当てについて説明する．第 3 章では独立故障集合生成について説明する．第 4 章では提案手法を説明する．第 5 章では実験結果について説明する．最後に第 6 章では，まとめと今後の課題について述べる．

2. 必須割当て

本章では必須割当て[9][10]について説明する．必須割当てとは，故障を検出するために必要な信号線の論理値割当てである．故障励起，一意活性化[11]及びある信号線の論理値を決定することにより，一意に決定する他の信号線の論理値を含意操作を用いて求める操作である．本論文では，必須割当て情報を 4.1 節で示す同時検出不可能グラフの生成に用いる．

図 1 に必須割当てアルゴリズムを示す．このアルゴリズムは対象回路 C と対象故障 f を入力とし，信号線と論理値 NA を出力する．はじめに，故障励起を行う(行 1)．次に，一意活性化が可能な限り，一意活性化を行い，NA に信号線と論理値を追加する(行 3，4，5)．そして，含意操作を行い，NA に信号線と論理値を追加する(行 6，7)．一意活性化が不可能になった場合，NA をアルゴリズムの出力として返す(行 9)．

```
1.  Necessary_assignment(C,f){
2.        NA=fault_excitation(C,f);
3.        while(unique_sensitizable){
4.           NA'=unique_sensitization(C,f);
5.           NA=NA∪NA';
6.           NA'=implication(C);
7.           NA=NA∪NA';
8.     }
9.  }return NA;
```

図 1．必須割当てアルゴリズム

A Lower Bound Calculation Method for the Number of Test Patterns Using Mandatory Assignment Information

Takanobu SONE，Toshinori HOSOKAWA，Masayoshi YOSHIMURA and Masayuki ARAI

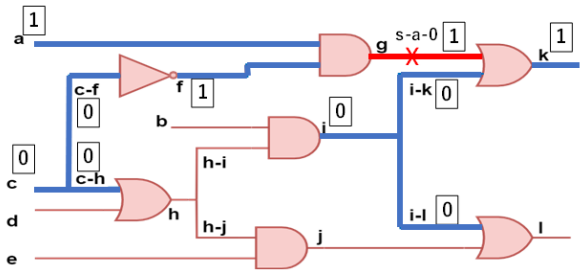

図 2. g の 0 縮退故障における必須割当ての例

図 2 にある回路の信号線 g の 0 縮退故障における必須割当ての例を示す．図 2 において，信号線 a，f,，g，k に 1 が割当てられ，信号線 c，c-h，c-f，i，i-k，i-l に 0 が割当てられる．故障励起のために信号線 g に 1 が割当てられ，一意活性化のために信号線 i-k に 0 が割当てられ，これらの含意操作から割当て値が求められた．

3. 独立故障集合生成

独立故障集合とは，同一テストパターンで検出することができない故障の集合のことである．独立故障集合の最大サイズを求めることによってテスト対象回路のテストパターン数の下界を求めることができる．テストパターン数の下界とは，テスト対象回路の検出対象故障をすべて検出可能なテストパターン数の最小値以下の値であり，最小値に近ければ，下界の質が高いと考えることができる．テストパターン数の質の高い下界を求めることで，テスト圧縮の研究を行う上での定量的評価の目標を定めることができる．

4. テストパターンの下界算出手法

本章は，提案手法であるテストパターン数の下界算出手法について説明する．本手法は，対象回路の必須割当て情報を用いて同時検出不可能故障グラフ生成を行い，最大独立故障集合を抽出することでテストパターン数の下界を算出する手法である．初めに 4.1 で同時検出不可能故障グラフについて説明し，4.2 で最大クリーク抽出について説明する．

4.1 同時検出不可能故障グラフ

本節では同時検出不可能故障グラフの説明をする．同時検出不可能故障グラフとは，各故障を頂点とし，同一テストパターンで検出することが不可能な故障を辺で接続したものである．各故障の必須割当て情報から同時検出不可能故障グラフを生成する．

図 3 に二つの故障の必須割当ての衝突の例を示す．ある信号線 a から g までの信号線からなるテスト対象回路において，故障 f1 と f2 を仮定する．必須割当て情報として，故障 f1 の検出には信号線 b，e に 0 を割当て，信号線 g に 1 を割当てる．故障 f2 の検出には信号線 c，d に 0 を割当て，信号線 a，e，f に

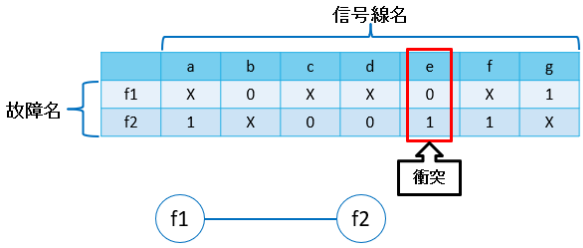

図 3. 必須割当ての衝突の例

1 を割当てる．それぞれの故障の必須割当て値を比較すると，信号線 e で割当て要求が衝突している．そのため，同一のテストパターンで検出することが不可能な故障として同時検出不可能故障グラフの頂点 f1 と頂点 f2 の間に辺を挿入し，接続する．

図 4 に同時検出不可能故障グラフの例を示す．あるテスト対象回路で検出故障が 5 つあり，それぞれの故障を f1，f2，f3，f4，f5 とする．図 4 で示すように，辺で接続されている f1 と f2，f2 と f3，f3 と f4，f2 と f4，f1 と f5 はそれぞれ同時検出不可能故障である．

4.2 最大クリーク抽出

本節では最大クリーク抽出アルゴリズムの説明をする．同時検出不可能グラフから最大独立故障集合を求めるために最大クリーク抽出を行う．最大クリーク抽出とは，対象のグラフでクリーク分割を行った際に，1 つのクリーク内の頂点数であるクリークサイズが最大になるようにクリーク分割を行うことである．クリーク分割とは，同時検出不可能グラフの各頂点を部分的な完全グラフになるように分割することである．図 5 に最大クリーク抽出アルゴリズム[12]を示す．

(Step1)

最大クリークサイズが更新される見込みがあるか否かを判断する．この判断基準としては，頂点集合 C の全頂点と隣接している頂点数と C の要素数を加算した値が現段階で抽出した最大クリークサイズよりも大きいか否かを基準とする．最大クリークサイズを更新する見込みがある場合は Step2 へ進み，見込みがない場合は，Step5 へ進む．

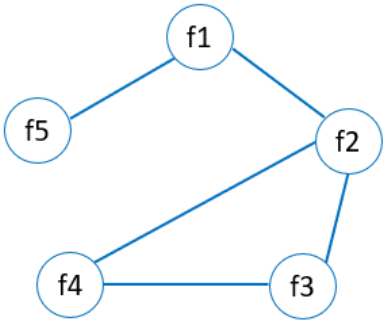

図 4. 同時検出不可能故障グラフの例

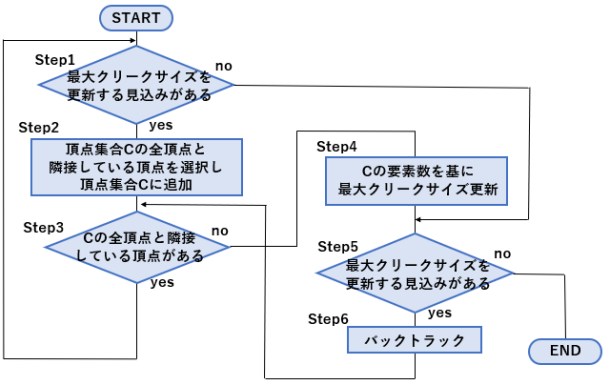

図 5. 最大クリーク抽出アルゴリズムのフローチャート

(Step2)

頂点集合 C の全頂点と隣接している頂点の中から頂点を 1 つ選択し，頂点集合 C に追加する．

(Step3)

頂点集合 C の頂点全てと隣接している頂点が存在するか探索する．存在する場合は Step1 へ進み，存在しない場合は Step4 へ進む．

(Step4)

頂点集合 C の要素数が最大クリークサイズよりも大きい場合，頂点集合 C の要素数を最大クリークサイズとする．

(Step5)

最大クリークサイズが更新される見込みがあるか否かを判断する．この判断基準としては，頂点集合 C の初期候補集合の要素数が最大クリークサイズよりも大きいか否かを基準とする．初期候補集合とは，頂点集合に追加する 1 つ目の頂点となる候補の集合である．最大クリークサイズを更新する見込みがある場合は Step6 へ進み，見込みがない場合は，最大クリークサイズを出力して終了する．

(Step6)

直前に追加した頂点を頂点集合 C から除外し，次に追加する候補集合からも除外する．Step3 へ進む．

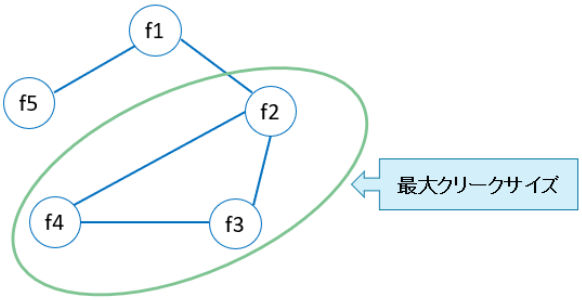

図 6. 最大クリーク抽出アルゴリズム適用例

図 6 に図 4 の同時検出不可能故障グラフに最大クリーク抽出アルゴリズムを適用した例を示す．この例では，頂点 f2，f3，f4 からなるクリークが最大サイズとなり，その要素数が 3 であることから最大クリークサイズが 3 であると求めることができる．よってテストパターン数の下界は 3 になる．

5. 実験結果

本章では，本論文で提案したテストパターン数の下界算出法を実装し，実験を行った結果を示す．実行環境は Intel Core i7-8565u 1.8GHz およびメモリ 8GB のコンピュータを使用した．対象回路は ISCAS’89 ベンチマーク回路である．対象故障モデルは単一縮退故障とする．

表 1 に同時検出不可能故障グラフの実験結果を示す．表 1 において，各故障の隣接頂点数を次数とし，対象回路の各故障の頂点数，辺数，最小次数，最大次数，平均次数，次数の標準偏差，枝密度[12]を記述している．枝密度とは，対象回路で完全グラフに対する同時検出不可能故障グラフ生成を行った際の頂点の間に引かれた枝の割合であり，枝の存在割合である．分子を同時検出不可能故障グラフにおける辺の数として，分母をグラフが完全グラフであるときの辺の数として計算する．つまり，枝密度が小さいほど引かれている辺の割合が少なく，テストパターン数の下界も小さくなる傾向にあると考えられる．例えば，s27 においては，頂点数が 32，辺数が 135，最小次数が 1，最大次数が 17 であり，平均次数が約 8，標準偏差が約 4 となった．このことから，次数が 4～12 である頂点が多いといえる．また，枝密度に関しては，約 27%であることが分かる．他の回路における枝密度としては，s820 の枝密度が約 59%であり，s13207，s15850，s35932 の枝密度が 5%以下であったことが特徴として挙げられる．

表 1. 同時検出不可能故障グラフの特徴に関する実験結果

回路名	頂点数	辺数	最小次数	最大次数	平均次数	標準偏差	枝密度(%)
s27	32	135	1	17	8.44	4.12	27.22
s208	215	6720	3	144	62.51	40.66	29.21
s344	342	7662	1	155	44.81	32.89	13.14
s420	430	21764	3	300	101.23	74.35	23.60
s820	850	212866	1	710	500.86	200.69	58.99
s1238	1353	187875	1	979	277.72	188.54	20.54
s1494	1506	469820	1	1224	623.93	294.31	41.46
s13207	9663	1808561	1	2010	374.33	651.01	3.87
s15850	11697	1247768	1	1473	213.35	300.59	1.82
s35932	38518	20604652	1	5262	1069.87	1474.91	2.78

表 2. 最大クリーク抽出の実験結果

回路名	下界	CPU時間(s)	クリーク探索回数	枝刈り回数
s27	4	0.37	258	74
s208	26	22406.08	84317725	42084867
s344	10	106.42	430765	208907

表 2 に最大クリーク抽出の実験結果を示す．表 2 において，対象回路におけるテストパターン数の下界，実験における CPU 時間，クリークの探索回数，最大クリーク抽出アルゴリズムによる枝刈り回数を記述している．テストパターン数の下界が s27 では 4 であり，s208 では 26 であり，s344 では 10 となった．

6. おわりに

本論文では，必須割当て情報を用いて，同時検出不可能故障グラフを生成し，最大独立故障集合生成を行い，最大独立故障集合サイズを算出し，テストパターン数の下界を求める手法を提案した．実験に関しては，対象回路の対象故障の各故障の頂点数，辺数，次数，枝密度，テストパターン数の下界の評価を行った．

今後の課題については，最大独立故障集合生成のアルゴリズムを改良し，実行時間の短縮及び，より正確なテストパターン数の下界を算出する．そして，求めたテストパターン数の下界をテスト圧縮の研究を行うための定量化評価の目標として定め，テスト圧縮を行うことが挙げられる．

参考文献

1) Y.Sato, T.Ikeda, M.Nakao, and T.Nagumo, “A bist approach for very deep sub-micron (vdsm) defect, ”Proc. International Test Conference, pp.283-291, 2000.
2) Seiji Kajihara, Irith Pomeranz, Kozo Kinoshita, “Cost-Effective Generation of Minimal Test Sets for Stuck-at Faults in Combinational Logic Circuits”, 30th ACM/IEEE Design Automation Conference, pp102-106,1993.
3) K. Miyase, S. Kajihara and Sudhakar M. Reddy, “A Method of Static Test Compaction Based on Don’t Care Identification,”IPSJ Journal，Vol.43, No.5，pp.1290-1293，May 2002.
4) K. Miyase, S. Kajihara “XID: Don’t Care Identifi-cation of Test Patterns for Combinational Circuits,” IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, Vol. 23, No. 2, pp. 321-326,Fed. 2004.
5) 八木澤圭，山崎浩二，細川利典，玉木久夫，“テストパターンの静的圧縮における厳密解と貪欲解の比較” 電子情報通信学会, 107, pp.77-82 , Feb. 2008.
6) D.Brelaz, “New methods to color the vertices of a graph” ,Communications of the ACM, 22, pp.251-256, 1979.
7) 藤原秀雄, "ディジタルシステムの設計とテスト," 工学図書株式会社.
8) Seiji Kajihara, Irith Pomeranz, Kozo Kinoshita, “Cost-Effective Generation of Minimal Test Sets for Stuck-at Faults in Combinational Logic Cir-cuits”, 30th ACM/IEEE Design Automation Conference, pp102-106,1993.
9) S. B. Akers， C. Joseph， and B. Krishnamurthy，” On the Role of Independent Fault Sets in the Generation of Minimal Test Sets”， in Proc. Intl. Test Conf.， 1987， pp. 1100–1107
10) M. H. Schulz, E. Trischler, “SOCRATES : A Highly Efficient Automatic Test Generation System,” IEEE Trans. on Computer-Aided design of Integrated Circuits and Systems, vol.7 No.1, pp.126-137, Jan. 1988.
11) 梶原 誠司， 猿渡 慶太郎， “静的学習における効率的な間接含意の発見と保存について，” 電子情報通信学会技術研究報告. VLD, VLSI 設計技術 102 (476), pp.25-28, Nov 2002.
12) 富田悦次，藤井利昭，"最大クリーク抽出の効率化とその実験的評価"信学論（D），vol.J68-D，no.3，pp.221-228，mar.1985