定義域限定に基づく論理関数分割を用いた並列論理検証 *

5B－2

永見康一　　木村晋二
奈良先端科学技術大学院大学 情報科学研究科†

1 はじめに

ディジタルシステムの CAD 技術は、BDD[1] の登場により大規模な問題に適用できるようになってきている。特に、設計が仕様を満たすかどうかを厳密に判定する形式的論理検証は、BDD による効率的な論理関数処理によって実用化が可能になったといえる。また、並列計算機の実用化にともなって、CAD への並列処理の応用が現実的になっており、BDD 処理の並列化技術の確立が望まれる。本稿では、分散計算環境向けの論理等価性判定手法を提案する。

2 準備

2.1 論理検証

論理検証とは、設計した回路の論理的な機能が与えられた仕様を満たすかどうかを判定する作業である。組み合わせ論理回路の場合には、仕様の論理関数と設計回路が表す論理関数との一致検査や、最適化などの変更を施した設計と元の設計との論理的な一致検査などの論理照合がこれにあたる。これらは論理関数の等価性判定によって行われる。

2.2 共有二分決定グラフ (SBDD)

二分決定グラフ (BDD) は、論理関数を根つきの非巡回有向グラフで表現したものであり、関数間の論理演算もグラフ操作で定義することができる。入力変数の順序を固定すれば論理関数の一意な表現となるので、複数の論理関数の間で同型な部分グラフを共有した共有二分決定グラフ (SBDD)[2] を用いれば、論理関数の等価性判定がポインタ (根節点を指し示す識別子) の比較のみで行なえる。したがって、組合せ論理回路の等価性は回路が表す論理関数を SBDD として構成することにより判定できる。

2.3 SBDD の並列構成手法

大規模な設計問題に対応するために、分散計算環境での SBDD の並列構成手法は重要である。そのような手法として、これまでに (1) 論理関数の再帰的な Shannon 展開によって定義されるコファクタを、各プロセッサで独立に処理する **Shannon 展開手法** (2) 多出力回路について各出力関数毎にプロセッサを割り当てる **出力分割手法**、(3) (1) と (2) の組み合わせ手法 が提案されている [3]。しかし、これらの手法を用いても例題によっては負荷の偏りが避けられない。例えば Shannon 展開手法について考えると、それぞれのコファクタが BDD として似た構造になる場合 (典型的には一致する場合) には台数効果が得られない。

*Parallel Logic Verification Using Domain Restriction Based Division

†Kouichi NAGAMI, Shinji KIMURA, Nara Institute of Science and Technology

3 定義域限定による並列等価性判定

回路の論理関数としての分割に着目し、より一般的な関数展開を用いた等価性判定手法を提案する。

3.1 定義

n 変数論理関数ベクトル $\boldsymbol{R} = (r_1,\cdots,r_P)$ が $\sum_{i=1}^{P} r_i = 1$ をみたすとき、$\boldsymbol{R}$ を**定義域被覆ベクトル**と呼ぶことにする。定義域被覆ベクトル $\boldsymbol{R}$ が与えられたとき、論理関数 $f = f(x_1,\cdots,x_n)$ について

$$f_i \stackrel{def}{=} f(y_1,\cdots,y_n) \quad (i=1,\cdots,P)$$

$$y_j = r_i \cdot x_j \quad (j=1,\cdots,n)$$

で定義される関数ベクトル $\boldsymbol{F} = (f_1,\cdots,f_P)$ を、f の$\boldsymbol{R}$ - コファクタ・ベクトルと呼ぶことにする。各 f_i は、r_i のオンセットについては f と等しい値をとり、オフセットに対しては $f(0,\cdots,0)$ をとる関数である。すなわち、r_i によって限定された定義域に関して f を表現する。回路で示すと図 1 のようになる。

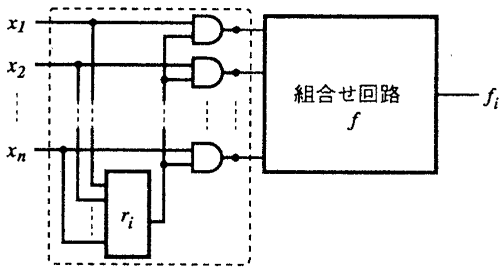

図 1: f_i を表す回路

また、展開式 $f = \sum_{i=1}^{P} r_i \cdot f_i$ が成立するが、$P = 2, \boldsymbol{R} = (x_1, \overline{x_1})$ とすると $f = x_1 \cdot f_1 + \overline{x_1} \cdot f_2$ であり、$x_1 \cdot f_1 = x_1 \cdot f|_{x_1=1}, \overline{x_1} \cdot f_2 = \overline{x_1} \cdot f|_{x_1=0}$ となる。すなわち、Shannon 展開を含むより一般的な展開である。

3.2 並列等価性判定

$\boldsymbol{R}$ - コファクタ・ベクトル は、以下の性質を持つ。

性質 1 (等価性の保存)

f, g を n 変数論理関数とし、$\boldsymbol{R}$ をある与えられた定義域被覆ベクトルとする。$\boldsymbol{F}, \boldsymbol{G}$ をそれぞれ f, g の $\boldsymbol{R}$ - コファクタ・ベクトルとするとき

$$f \equiv g \Leftrightarrow \boldsymbol{F} \equiv \boldsymbol{G}$$

が成り立つ。ただし、右辺は "$f_1 \equiv g_1$かつ $f_2 \equiv g_2$かつ$\cdots f_P \equiv g_P$" を表す。 □

すなわち、与えられた等価性判定問題は、P 個の独立な部分問題 $(f_i \equiv g_i : i = 1, \cdots, P)$ に分割することができる。アルゴリズムの概略は次のようになる。

[並列論理等価性判定]
入力: n 変数論理関数 f, g の記述
　　定義域被覆ベクトル$\boldsymbol{R}$の記述
出力: $f \equiv g$ または $f \not\equiv g$
begin
　for $1 \leq i \leq P$ **pardo begin**
　　r_i の BDD を構成
　　for $j := 1$ **to** n **do**
　　　$y_j = r_i \cdot x_j$ の BDD を構成
　　f, g の記述から $f_i = f(y_1, \cdots, y_n)$ と
　　　$g_i = g(y_1, \cdots, y_n)$ の BDD を構成
　　if (f_i の BDD と g_i の BDD が一致)
　　　then yes **else** no
　end
　全ての i について yes ならば $f \equiv g$
　　さもなくば $f \not\equiv g$
end.

3.3 一般化コファクタによる節点数の削減

論理関数 f の BDD B_f について、ある限定された定義域 $C \subseteq \{0,1\}^n$ のみに着目してより小さな BDD を得る手法として、一般化コファクタ [4] が知られている。これは、B_f と、C の特徴関数 c の BDD B_c が与えられたとき、$\overline{C}$ における f の関数値を don't care として BDD が小さくなるように B_f を変形するグラフ操作である。結果の BDD が表す論理関数を $(f)_c$ と書く。$(f)_c$ は以下の性質を持つ。

(i). (変数順序づけ固定の下で) f, c に対して一意

(ii). $(f \circ g)_c = (f)_c \circ (g)_c$, $\left(\overline{f}\right)_c = \overline{(f)_c}$, $(f)_f = 1$
　(ただし、$\circ$ は任意の二項論理演算を表す)

先の性質 1 は、(i) より $\boldsymbol{F}, \boldsymbol{G}$ をそれぞれ $(\boldsymbol{F})_{\boldsymbol{R}} \stackrel{def}{=} ((f_1)_{r_1}, \cdots, (f_n)_{r_n})$, $(\boldsymbol{G})_{\boldsymbol{R}} \stackrel{def}{=} ((g_1)_{r_1}, \cdots, (g_n)_{r_n})$ に置き換えても成立する。さらに (ii) より、

$$\begin{aligned}(f_i)_{r_i} &= \left(f(y_1, \cdots, y_n)\right)_{r_i} \\ &= f\left((r_i \cdot x_1)_{r_i}, \cdots, (r_i \cdot x_n)_{r_i}\right) \\ &= f\left((x_1)_{r_i}, \cdots, (x_n)_{r_i}\right)\end{aligned}$$

となるので、先のアルゴリズムにおいて y_j の定義を $r_i \cdot x_j$ から $(x_j)_{r_i}$ に変更すれば $(\boldsymbol{F})_{\boldsymbol{R}}, (\boldsymbol{G})_{\boldsymbol{R}}$ を用いた等価性判定になる。($(\boldsymbol{F})_{\boldsymbol{R}}, (\boldsymbol{G})_{\boldsymbol{R}}$の節点数 $\leq$ $\boldsymbol{F}, \boldsymbol{G}$の節点数) であるので、節点数を削減できる。

4 実験と考察

現在、提案手法の実現を進めているが、ここでは予備実験の結果によって提案手法の可能性を示す。実験では、ISCAS'89 のベンチマーク回路 C3540 の回路記述から各手法を用いて SBDD を構成し、実行時間と総節点数を比較している。実験環境として富士通の分散主記憶型高並列計算機 AP1000 を 64 プロセッサ構成で用いた。定義域被覆ベクトルとしては、Shannon 展開の経験的手法である折り返し割当手法を模倣するものを試験的に採用している。

表 1 : C3540 による各手法の比較

	1 台	(1)	F	$(F)_R$
# nodes (min)	-	20,791	40,032	32,374
(max)	608,890	117,034	67,818	59,681
time (sec.)	28.0	5.3	2.9	2.6

1 台 : プロセッサ 1 台 ; (1): 従来手法 (1) ; F : $y_j = r_i \cdot x_j$; $(F)_R$: $y_j = (x_j)_{r_i}$; # nodes : 各プロセッサ上の節点数の 最小値と最大値 ; time: 実行時間

提案手法では、従来手法に比べて負荷分散が均等化されていることがわかる。また、一般化コファクタによりさらに効率が改善されている。

5 おわりに

以上、分散計算環境に適する論理等価性判定手法について述べた。提案手法は、定義域被覆ベクトル $\boldsymbol{R}$ の選定が効率を左右する。今後は良い負荷分散を与える $\boldsymbol{R}$ を見つける手法を考案することが課題である。

謝辞　日頃から御討論いただく本学情報科学研究科渡邊勝正教授をはじめ渡邊研究室の皆様に感謝致します。また、AP1000 の利用環境を与えて下さる富士通研究所に深謝致します。

参考文献

[1] R.E.Bryant: Graph-based Algorithms for Boolean Function Manipulation, *IEEE Trans. Comput.*, C-35, No.8, pp.677-691 (Aug. 1986).

[2] S.Minato, N.Ishiura and S.Yajima: Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean Function Manipulation, *Proc. 27th ACM/IEEE Design Automat. Conf.*, pp.52-57 (June 1990).

[3] 木村晋二, 永見康一: 分散記憶型並列計算機に対する並列二分決定グラフ構成アルゴリズム, 情報処理学会第 47 回全国大会,6H-08 (1993).

[4] H.J.Touati, et al.: Implicit State Enumeration of Finite State Machines using BDD's, *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp.130-133 (Nov. 1990)