

An Effective Implementation of a Special Quantifier Elimination for a Sign Definite Condition by Logical Formula Simplification

Iwane, Hidenao
Fujitsu Laboratories Ltd

Higuchi, Hiroyuki
Fujitsu Laboratories Ltd

Anai, Hirokazu
Fujitsu Laboratories Ltd | Kyushu University

<https://hdl.handle.net/2324/1430844>

出版情報 : COE Lecture Note. 49, pp.33-42, 2013-08-09. 九州大学マス・フォア・インダストリ研究所
バージョン :
権利関係 :



セッション 3

Session 3

制御系設計

Design of control systems

An Effective Implementation of a Special Quantifier Elimination for a Sign Definite Condition by Logical Formula Simplification

岩根秀直

(株) 富士通研究所*

HIDENAO IWANE

FUJITSU LABORATORIES LTD

樋口博之

(株) 富士通研究所†

HIROYUKI HIGUCHI

FUJITSU LABORATORIES LTD

穴井宏和

(株) 富士通研究所/九州大学‡

HIROKAZU ANAI

FUJITSU LABORATORIES LTD/KYUSHU UNIVERSITY

Abstract

This paper presents an efficient quantifier elimination algorithm tailored for a sign definite condition (SDC). The SDC for a polynomial $f \in \mathbb{R}[x]$ with parametric coefficients is written as $\forall x (x \geq 0 \rightarrow f(x) > 0)$. To improve the algorithm, simplification of an output formula is needed. We show a necessary condition for the SDC and an approach to simplify formulae by using a logic minimization method. Experimental results show that our approach significantly simplify formulae.

1 はじめに

限量記号消去アルゴリズム (quantifier elimination (QE) algorithm) とは、与えられた形式的理論 (formal theory) について「限量記号付きの一階述語論理式」を入力とし「等価で限量記号無しの論理式」を出力するアルゴリズムのことである。QE は工学や産業上の問題などの多くの応用があり重要なアルゴリズムである。しかし、QE は計算量の下限が限量記号がついた変数の数に対して二重指数であることが示されており、本質的に規模の大きな問題を解くことができない。そのため、限量記号がついた変数が線形の場合 [5] など制限された入力の一階述語論理式に対する専用アルゴリズムの研究がすすめられている。

多項式 $f(x) \in \mathbb{R}[x]$ に対して、 $\forall x (x \geq 0 \rightarrow f(x) > 0)$ を sign definite condition (SDC) と呼ぶ。制御系設計の様々な条件が SDC で記述することができる [2] ため、SDC に対する QE は実用上重要な問題のクラスである。SDC が「定数項が正の多項式が $x \geq 0$ において実根を持たないこと」と等価なことを利用して、Sturm-Habicht 列 [4] を用いた実根の数え上げにより高速に計算する手法が提案されている [7]。

本稿では、SDC 専用の QE の出力となる論理式の簡単化の手法について述べる。出力の論理式の簡単化は、QE の計算と実行可能領域の描画などの後処理の高速化につながる。論理式の簡単化のために、最初

*iwane@jp.fujitsu.com

†h-higuchi@jp.fujitsu.com

‡anai@jp.fujitsu.com

に、SDC を満たす多項式の Sturm-Habicht 列が満たす条件を示し、Sturm-Habicht 列による実根の数え上げによって得られる符号列の数を削減することで、論理式を簡単化する。次に、既存の論理関数処理を適用して $g > 0 \wedge g = 0 \leftrightarrow g \geq 0$ の規則による論理式の簡単化を行う。計算機実験結果により、本稿で提案する手法の効果を示す。

本稿の構成は以下の通りである。まず、2 章において Sturm-Habicht 列と SDC 専用 QE アルゴリズムについて紹介する。3 章では、SDC を満たす多項式の Sturm-Habicht 列においてあらわれない符号列を示し、4 章では、論理関数処理の紹介とそれを用いた論理式の簡単化手法を述べる。5 章で、実験結果により本稿で述べる手法による論理式簡単化の効果を示す。最後に 6 章で本稿のまとめと今後の課題を述べる。

2 Sign Definite Condition 専用 Quantifier Elimination

本章では、sign definite condition の定義と、[7] で提案されている専用の QE について述べる。

2.1 Sign Definite Condition と Sturm-Habicht 列による実根の数え上げ

最初に sign definite condition (SDC) を定義する。制御系設計の様々な条件が SDC で記述できるため、SDC は重要な問題のクラスであり、専用の QE アルゴリズムが提案されている [7, pp. 208–211]。

定義 1

多項式 $f(x) \in \mathbb{R}[x]$ に対する以下の条件を **sign definite condition (SDC)** という。

$$\forall x (x \geq 0 \rightarrow f(x) > 0)$$

定義 2

$f(x)$ を n 次の実係数多項式とする。このとき、 $\text{SH}_n(f) = f$, $\text{SH}_{n-1}(f) = \frac{df}{dx}$ とし、整数 j ($0 \leq j \leq n-2$) について $\text{SH}_j(f) = \delta_{n-j} \text{Sres}_j(f, \frac{df}{dx})$ として構成する多項式の列 $\text{SH}(f) := \{\text{SH}_n(f), \dots, \text{SH}_0(f)\}$ を **Sturm-Habicht 列** という。ここで、 $\text{Sres}_j(f, g)$ は f と g の j 次部分終結式 [7, p. 129] で、 $\delta_j = (-1)^{j(j-1)/2}$ である。また、 $\deg(\text{SH}_k(f)) = k$ のとき、 $\text{SH}_k(f)$ は正則であるという。

定義 3

符号とは、正、負または 0 のことで、それぞれ、 $+1$, -1 または 0 で表す。実数の有限列 $A = \{a_m, \dots, a_0\}$ における符号変化の数 $V(A)$ は、以下の規則に従い数える。

- 次の符号列を 1 と数える： $\{-1, +1\}, \{+1, -1\}, \{-1, 0, +1\}, \{+1, 0, -1\}, \{-1, 0, 0, +1\}, \{+1, 0, 0, -1\}$
- 次の符号列を 2 と数える： $\{+1, 0, 0, +1\}, \{-1, 0, 0, -1\}$
- 上記以外の符号列は 0 と数える

さらに、実数係数の有限個の多項式列 $S(x) = \{S_n(x), S_{n-1}(x), \dots, S_0(x)\}$ とするとき、 $\{h_m(x), \dots, h_0(x)\}$ を $S(x)$ から恒等的に 0 になる多項式を取り除いたものとし、実数 α に対して、 $V_\alpha(S)$ を $V(\{h_m(\alpha), \dots, h_0(\alpha)\})$ と定義する。

次の定理により Sturm-Habicht 列を用いて任意の区間における多項式の実根を数え上げることができる。

定理 4

$f(x) \in \mathbb{R}[x]$, $a, b \in \mathbb{R} \cup \{-\infty, +\infty\}$ ($a < b$) とし、 $f(a)f(b) \neq 0$ を満たすとする。このとき、 $V_a(\text{SH}(f)) - V_b(\text{SH}(f))$ は区間 $[a, b]$ における $f(x)$ の実根の数に一致する。

本稿の残りでは次の記法を用いる.

記法 1

$\text{SH}_k(f)$ の $x = \infty$ における符号を s_k , $\text{SH}_k(f)$ の $x = 0$ における符号を c_k と表記する.

注意 1

$\text{SH}_k(f) = a_{k,k}x^k + a_{k,k-1}x^{k-1} + \cdots + a_{k,0}$ とするとき, 以下が成立する.

$$\begin{aligned} s_k = 0 &\leftrightarrow a_{k,i} = a_{k,k-1} = \cdots = a_{k,0} = 0 \\ s_k > 0 &\leftrightarrow (a_{k,k} > 0) \vee (a_{k,k} = 0 \wedge a_{k,k-1} > 0) \vee \cdots \vee (a_{k,k} = a_{k,k-1} = \cdots = a_{k,1} = 0 \wedge a_{k,0} > 0) \end{aligned}$$

したがって, $s_k = 0$ のとき, $\text{SH}_k(f)$ は恒等的に 0 であり, c_k は $a_{k,0}$ の符号に一致する. また, n 次の多項式に対して, $s_n = s_{n-1}$, $s_0 = c_0$ となることに注意する.

Sturm-Habicht 列に対して以下の Sturm-Habicht Structure Theorem [4] が成立する.

定理 5

f を次数 n (≥ 2) の多項式とする. $\text{SH}_{k+1}(f)$ が正則となるすべての k に対して, $\deg(\text{SH}_k(f)) = r \leq k$ とするとき, 以下が成立する.

- (A) $r < k - 1$ のとき, $\text{SH}_{k-1}(f) = \cdots = \text{SH}_{r+1}(f) = 0$,
- (B) $r < k$ のとき, $\text{lc}(\text{SH}_{k+1}(f))^{k-r} \text{SH}_r(f) = \delta_{k-r} \text{lc}(\text{SH}_k(f))^{k-r} \text{SH}_k(f)$,
- (C) $r < k$ のとき, $\text{lc}(\text{SH}_{k+1}(f))^{k-r+2} \text{SH}_{r-1}(f) = \delta_{k-r+2} \text{Prem}(\text{SH}_{k+1}(f), \text{SH}_k(f))$.

ここで, $\text{lc}(g)$ は g の主係数で, $\text{Prem}(g, h)$ は, g と h の擬剰余であることを表す.

2.2 SDC 専用 QE の実装

本節では, 穴井らにより提案された SDC 専用の QE アルゴリズムの実装方法 [7] について述べる. 提案手法は, 次数一定の多項式 $f(x)$ に対する SDC が, $f(x)$ が $x \geq 0$ で実根を持たないことと等価であることを利用する.

最初に, オフラインで次数毎に係数をパラメータとする多項式に対して, $x \geq 0$ で実根を持たない符号条件 φ_n をあらかじめ求め, その情報をデータベースなどに蓄積しておく. f が入力された後 (オンライン) の計算は Sturm-Habicht 列を求めて代入する部分だけとなるので, 高速な QE 計算が実現される. 実際, 5 次の問題は汎用の QE アルゴリズムである Cylindrical Algebraic Decomposition では 1 時間たっても計算が停止しないが, 提案手法では 1 秒に満たない時間で計算できる.

オンラインの計算を高速化するためには, オフラインでの φ_n の表現を簡単化することが必要である. また, φ_n の表現の簡単化は出力される論理式の簡単化につながり, 実行可能領域の描画や真偽値を判定する場合など後処理の高速化にもつながる. 次章以降では, オフラインで計算する φ_n の簡単化について考える.

例 1

2 次の多項式 $f(x) = x^2 + bx + c$ の場合を考える. $V_0(\text{SH}(f)) - V_\infty(\text{SH}(f)) = 0$ となる符号条件は表 1 のようになる. ここで, 注意 1 より, $s_2 = s_1 > 0$, $s_0 = c_0$ であり, $f(0) > 0$ より, $c > 0$ なので $c_2 > 0$ となることに注意する.

表 1: φ_2

s_2	s_1	s_0	c_2	c_1	c_0	s_2	s_1	s_0	c_2	c_1	c_0
+1	+1	-1	+1	-1	-1	+1	+1	-1	+1	0	-1
+1	+1	-1	+1	+1	-1	+1	+1	0	+1	0	0
+1	+1	0	+1	+1	0	+1	+1	+1	+1	+1	+1

$f(x)$ の Sturm-Habicht 列は $\text{SH}(f) = \{x^2 + bx + c, 2x + b, b^2 - 4c\}$ なので、以下が φ_2 から得られる.

$$\begin{aligned}
 (b^2 - 4c < 0 \wedge c > 0 \wedge b < 0) \quad \vee \quad (b^2 - 4c < 0 \wedge c > 0 \wedge b = 0) \quad \vee \\
 (b^2 - 4c < 0 \wedge c > 0 \wedge b > 0) \quad \vee \quad (b^2 - 4c = 0 \wedge c > 0 \wedge b = 0) \quad \vee \\
 (b^2 - 4c = 0 \wedge c > 0 \wedge b > 0) \quad \vee \quad (b^2 - 4c > 0 \wedge c > 0 \wedge b > 0)
 \end{aligned} \tag{1}$$

式 (1) は $g > 0 \wedge g = 0 \leftrightarrow g \geq 0$ の規則を利用して、以下のように簡単化できる.

$$\begin{aligned}
 (b^2 - 4c < 0 \wedge c > 0) \quad \vee \quad (b^2 - 4c = 0 \wedge c > 0 \wedge b \geq 0) \quad \vee \\
 (b^2 - 4c > 0 \wedge c > 0 \wedge b > 0)
 \end{aligned}$$

$b^2 - 4c = 0 \wedge c > 0 \wedge b = 0$ を満たす実数 b, c が存在しないことを利用するとさらに簡単化できる.

$$(b^2 - 4c < 0 \wedge c > 0) \quad \vee \quad (b^2 - 4c \geq 0 \wedge c > 0 \wedge b > 0)$$

上記の条件をオフラインで構築しておき、具体的な問題に対して b, c に値を代入するだけで QE が実現される. このように事前の簡単化によりオンラインでの代入回数を削減し、専用 QE の高速化が実現できる.

3 SDC の必要条件

例 1 で見たように、Sturm-Habicht 列から得られる条件には、それを満たす実数が存在しないことがあり、それらを削減により論理式を簡単化できる. 本章では SDC を満たす多項式 の Sturm-Habicht 列が満たす必要条件を示す. ここでは、定理 4 および記法 1 の記法を使用する.

定理 6

f を n 次の実係数多項式, u を $s_k \neq 0$ となる最小の非負整数 k とする. f が SDC を満足するとき、以下に示す条件を満たす.

$$\begin{aligned}
 V_0(\text{SH}(f)) - V_\infty(\text{SH}(f)) &= 0, \quad s_n > 0, \quad s_{n-1} > 0, \quad c_n > 0, \quad s_0 = c_0, \\
 c_u &\neq 0, \quad c_{n-1} = 0 \rightarrow c_{n-2} < 0, \quad s_{n-2} = 0 \rightarrow s_{n-3} = \dots = s_0 = 0, \\
 s_k = 0 &\rightarrow c_k = 0, \quad (\forall k \in \{0, \dots, n-2\}), \\
 c_{k+2} &\neq 0 \wedge c_{k+1} = 0 \rightarrow c_k \neq c_{k+2}, \quad (\forall k \in \mathcal{N} = \{u, \dots, n-2\}), \\
 c_k = c_{k+1} &= 0 \wedge c_{k-1}c_{k+2}s_k s_{k+1} \neq 0 \rightarrow s_k s_{k+2} < 0, \quad (\forall k \in \mathcal{N}), \\
 c_k = \dots &= c_{k+m} = 0 \rightarrow s_{k+1} = \dots = s_{k+m-1} = 0 \quad (\forall k \in \mathcal{N}, m > 1), \\
 s_{k+2} &= 0 \wedge s_{k+1} \neq 0 \rightarrow s_k \neq 0, \quad (\forall k \in \mathcal{N}), \\
 s_{k-1} &\neq 0 \wedge s_k = \dots = s_{k+m} = 0 \wedge s_{k+m+1} \neq 0 \rightarrow s_{k+m+2}^m s_{k-1} = \delta_{m+2} s_{k+m+1}^{m+1} \\
 \wedge s_{k+m+2}^m c_{k-1} &= \delta_{m+2} s_{k+m+1}^m c_{k+m+1}, \quad (\forall k \in \mathcal{N}, m \geq 0).
 \end{aligned}$$

定理の証明は以下の補題で与える。補題 8 と補題 12 の一部は [4] で示されている。

補題 7

次数 n の多項式 $f(x) \in \mathbb{R}[x]$ に対して, $s_n = s_{n-1} = c_n > 0$ は SDC の必要条件である。

証明 $f(x) = \sum_{i=0}^n p_i x^i$ ($p_n \neq 0$) とする。今, $f(0) = p_0 > 0$ なので, $c_n > 0$ 。また, 十分大きな x に対して $f(x) > 0$ を満たすには, $p_n > 0$ が必要なので, $s_n > 0$ となる。注意 1 より $s_{n-1} = s_n$ が成立する。■
以下では, 補題 7 より, 入力多項式 f の主係数および定数項は正, つまり, $s_n = c_n > 0$ と仮定する。また, 以降では簡単のため, $SH_k(f)$ を SH_k と表記する。

補題 8

u を $s_k \neq 0$ を満たす最小の非負整数とすると, $c_u \neq 0$ が成立する。

証明 定義から SH_u は $\gcd(f, df/dx)$ の定数倍になる。 $p_0 > 0$ から, $f(0) \neq 0$ なので, $SH_u(0) \neq 0$ 。■

補題 9

$c_{n-1} = 0$ ならば $c_{n-2} < 0$ が成立する。

証明 $SH_{n-1}(0) = p_1$, $SH_{n-2}(0) = p_1 p_{n-1} - n^2 p_0 p_n^2$ で, 今 $p_n > 0$, $p_0 > 0$ であり, $c_{n-1} = 0$ のとき, $p_1 = 0$ なので, $SH_{n-2}(0) = -n^2 p_0 p_n^2 < 0$ ■

補題 10

$s_{n-2} = 0$ ならば $s_{n-3} = \dots = s_1 = s_0 = 0$ が成立する。

証明 $s_k \neq 0$ となる非負の整数 $k \leq n-3$ が存在すると仮定し, その最大値を k_m とする。このとき, 定理 5 から SH_{n-1} の次数は k_m でなければならないが, $SH_{n-1} = df/dx$ なので矛盾する。■

補題 11

任意の $k = 0, \dots, n$ に対して, $s_k = 0$ ならば $c_k = 0$ が成立する。

証明 注意 1 より, $s_k = 0$ のとき, $SH_k = 0$ なので, $c_k = 0$ 。■

補題 12

任意の $k = 1, \dots, n-1$ に対して, $c_{k+1} \neq 0 \wedge c_k = 0$ ならば $c_{k-1} \neq c_{k+1}$ が成立する。

証明 以下の 5 つの場合を考えればいい。

(1) $\deg(SH_k) = r, \deg(SH_{k+1}) = k+1, SH_k(0) = c_k = 0$ のとき,

(a) $r = k$ の場合, 定理 5 (C) から, $\text{lc}(SH_{k+1})^2 SH_{k-1} = -\text{Prem}(SH_{k+1}, SH_k)$ 。よって, $c_{k-1} = -\text{lc}(SH_{k+1})^{-2} c_{k+1} \cdot f(0) \neq 0$ なので, SH_k と SH_{k+1} が同時に x を共通因子に持つことはない。したがって, $c_{k+1} \neq 0$ となる。したがって, $c_{k+1} \neq c_{k-1}$ 。

(b) $r < k$ の場合, 定理 5 (A), (C) より, $c_{k-1} = 0$ 。

(2) $\deg(SH_{k+1}) = r < k+1, \deg(SH_{k+2}) = k+2, SH_k(0) = c_k = 0$ のとき,

(a) $r < k-1$ 場合, 定理 5 (A) より $c_{k-1} = 0$ 。

(b) $r = k-1$ 場合, 定理 5 (B) より $\text{lc}(SH_{k+2})^2 SH_{k-1} = -\text{lc}(SH_{k+1})^2 SH_{k+1}$ なので, $SH_{k-1} SH_{k+1} \leq 00$ となる。

(c) $r = k$ 場合, 定理 5 (B) より SH_k は SH_{k+1} の定数倍となり, $c_{k+1} \neq 0 \wedge c_k = 0$ を満たさない。

補題 13

任意の $k = u + 1, \dots, n - 2$ に対して, $c_{k-1} \neq 0 \wedge c_k = c_{k+1} = 0 \wedge c_{k+2} \neq 0 \wedge s_k \neq 0 \wedge s_{k+1} \neq 0$ ならば $s_{k+2}s_k < 0$ が成立する.

証明 補題 11 より $c_{k+2} \neq 0$ なので $\text{SH}_{k+2} \neq 0$. SH_{k+2} が正則でないと仮定すると, 定理 5 (A) より SH_{k+1} は正則となる. さらに, 定理 5 (B) より SH_{k+1} は SH_{k+2} の定数倍となり, $c_{k+2} \neq 0$ かつ $c_{k+1} = 0$ に矛盾する. したがって, SH_{k+2} は正則.

SH_{k+1} が正則とすると, 定理 5 (C) より, $\text{lc}(\text{SH}_{k+2})^2 \text{SH}_k = \delta_2 \text{Prem}(\text{SH}_{k+2}, \text{SH}_{k+1})$ が成立. SH_k と SH_{k+1} が x を共通因子を持ち, $f(0) \neq 0$ に矛盾する. したがって, SH_{k+1} は正則ではない. また定理 5 (A) よりその次数は k でなければならない. 定理 5 (B) より, $s_k = -s_{k+2}$ が得られる. ■

補題 14

任意の $k = u + 1, \dots, n - 2, m = 2, \dots, n - k - 1$ に対して, $c_{k+m+1} \neq 0 \wedge c_{k+m} = \dots = c_k = 0$ ならば $s_{k+m-1} = \dots = s_{k+1} = 0$ が成立する.

証明 (1) $s_{k+m} \neq 0$ のとき, 補題 13 の証明より, SH_{k+m} は正則ではなく, SH_{k+m+1} は正則となる. SH_{k+m} の次数 d_1 が k より大きいとすると, 定理 5 (C) より, SH_{d_1-1} は $\text{Prem}(\text{SH}_{k+m+1}, \text{SH}_{k+m})$ の定数倍になる. $c_{k+m} = c_{d_1-1} = 0$ より, x が共通因子となり, $f(0) \neq 0$ に矛盾する. したがって, $d_1 \leq k$. このとき, 定理 5 (A) より, $s_{d_1+1} = \dots = s_{k+m-1} = 0$.

(2) $s_{k+m} = 0$ のとき, SH_{k+m} の次数 d_2 が k 以上とすると, 定理 5 (B) より, SH_{d_2} は SH_{k+m+1} の定数倍なので $c_{k+m+1} \neq 0, c_{d_2} = 0$ に矛盾する. したがって, $d_2 < k$. このとき, 定理 5 (A) より, $s_{d_2+1} = \dots = s_{k+m-1} = 0$. ■

補題 15

任意の $k = u + 1, \dots, n - 3$ に対して, $s_{k+2} = 0 \wedge s_{k+1} \neq 0$ ならば $s_k \neq 0$ が成立する.

証明 定理 5 (B) より, SH_{k+1} は正則. 定理 5 (C) から, SH_k はある SH_r ($r > k + 1$) と SH_{k+1} の擬剰余により得られる. 今, $k > u$ なので, $\text{SH}_k \neq 0$ となる. ■

補題 16

$k = u + 1, \dots, n - 3$ に対して, ある整数 $m \geq 0$ が存在し, $s_{k-1} \neq 0 \wedge s_k = \dots = s_{k+m} = 0 \wedge s_{k+m+1} \neq 0$ ならば $s_{k+m+2}^m s_{k-1} = \delta_{m+2} s_{k+m+1}^{m+1} \wedge s_{k+m+2}^m c_{k-1} = \delta_{m+2} s_{k+m+1}^m c_{k+m+1}$ が成立する.

証明 定理 5 (B) より得られる. ■

4 論理式の簡単化

本章では, 論理関数処理を用いた論理式の簡単化について述べる.

4.1 論理代数と論理関数の簡単化

本節では論理代数と論理関数を定義する.

定義 17

論理代数は、論理値の集合 $B = \{0, 1\}$ に関する論理積 (\cdot)、論理和 ($+$)、論理否定 ($'$) の 3 つの演算からなる代数系として定義される。ここで論理積、論理和、論理否定は図 1 のように定義される。

x	y	$x \cdot y$	x	y	$x + y$	x	x'
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1	1	0
1	1	1	1	1	1	1	0

図 1: 論理演算子（論理積・論理和・論理否定）

定義 18

論理変数およびその否定のことをリテラル (*literal*)、1 個のリテラル、または複数の互いに異なる変数のリテラルの論理積を積項 (*product term*)、1 個の積項、または複数の異なる積項の論理和を積和形 (*sum-of-products form* または *disjunctive form*) と呼ぶ。

定義 19

定義 17 における論理演算子と括弧、及び任意の個数の論理変数と論理定数 (0 または 1) を組み合わせて計算手順を表した式を論理式と呼ぶ。また、関数 $f: B^n \rightarrow B$ を論理関数 (*logic function*) と呼ぶ。

定義 20

n 変数論理関数の入力値の 0, 1 の組み合わせは 2^n 通りある。通常の論理関数は、すべての入力の組み合わせに対する出力が定義されており、そのような論理関数を完全指定論理関数 (*completely specified logic function*) と呼ぶ。それに対して、一部の入力値に対しては、出力が未定義な論理関数を不完全指定論理関数 (*incompletely specified logic function*) と呼び、出力が未定義な入力値をドントケア (*don't care*) と呼ぶ。

例えば、 $(x + y)'$ と $x' + y'$ は等価な論理式であるように 1 つの論理関数は複数の等価な論理式により表現できる。本稿では、与えられた論理式に対して、等価でより積項数の少ない論理式を得ることを論理式の簡単化と呼ぶ。不完全指定論理関数では、ドントケアを都合の良い値に解釈して、論理式をより簡単化できる場合がある。

論理関数を表す論理式を簡単化することは回路素子の数や配線の本数が少なくなる設計につながるため実用上重要である。そのため、二分決定グラフ (Binary Decision Diagram: BDD) を用いた厳密解法やヒューリスティクスを用いた近似解法 ESPRESSO [3] など多くの研究がなされている。

4.2 論理関数処理を用いた φ_n の簡単化

論理式簡単化手法を利用して、SDC を満足する論理式 φ_n を簡単化することを考える。

まず、論理変数は 2 つの値をとり、本稿で扱う多項式の符号は 3 つの値をとるので 2 つの論理変数を用いて s_k, c_k の符号を表現する。ここでは、 x, y を用いて、 $x'y'$ を 0, xy' を正、 $x'y$ を負と表現し、 φ_n を論理変数を用いた論理式として記述する。その結果に対して論理関数処理による簡単化手法の適用で、より簡単な φ_n の表現を得る。

より簡単な論理式を得るために、SDC 専用の QE においては次のようにドントケアを設定した。(1) 多項式の符号を x, y の 2 変数で表現することを上で述べた。符号は 3 値使用し、論理変数 2 変数で表すことができるのは 4 値であり、 xy は使用していないため、ドントケアとして設定する。(2) 補題 8-16 を満たさない符号列は、補題 7 の仮定の元でそれを満たす実数が存在しない。したがって、真と偽、どちらの値

表 2: 計算実験結果

次数	変数	SyN	DC	ESP	ESP	積項数	時間 近似	時間 厳密	SDC 近似		
				近似	厳密				変数	厳密	時間
2	4	5	2	2	2	5	0.01	0.01	2	1	0.00
3	8	17	7	4	4	21	0.01	0.01	4	3	0.00
4	12	64	24	10	10	99	0.01	0.04	6	4	0.00
5	16	302	85	18	18	480	0.05	1.12	8	15	0.00
6	20	1229	299	57	57	2352	0.72	61.92	10	21	0.01
7	24	5238	1096	121	-	11656	21.40	>350h	12	84	0.15
8	28	20468	4037	353	-	58284	757.59	>350h	14	120	1.49

として扱っても良いためドントケアとして設定する. (3) $V_0(\text{SH}(f)) < V_\infty(\text{SH}(f))$ とはならないので, この場合もドントケアとして設定する.

5 実験結果

論理関数処理による論理式簡単化に対する近似手法 ESPRESSO [1] を用いて φ_n の簡単化を行った結果を表 2 に示す. 計算は Intel(R) Core(TM) 2 Duo CPU 1.6 GHz, 2.0 Gbyte メモリ上で行った. 「次数」は入力となる多項式の次数, 「変数」は ESPRESSO に与える論理変数の数, 「SyN」は SyNRAC の以前の実装での積項の数, 「DC」は 4.2 節の (1) と補題 11 のみをドントケアで設定して ESPRESSO の近似解法を用いて簡単化した積項の数, 「ESP 近似」は ESPRESSO の近似解法を用いて簡単化した積項の数, 「ESP 厳密」は ESPRESSO の厳密解法を用いて簡単化した積項の数, 「積項数」は定理 5 を満たす符号列の数, 「時間近似」は「ESP 近似」での実行時間 (秒) を表す. 「時間厳密」は「ESP 厳密」での実行時間 (秒) を表す.

次数 n の場合には Sturm-Habicht 列は $n+1$ 個からなるが, 補題 7 より, $s_n = s_{n-1} = c_n > 0$ であり, $s_0 = c_0$ となるので, 符号条件を求めるのに考慮すべき対象は $2(n+1) - 4 = 2n - 2$ 個である. したがって論理変数の数としては $4n - 4$ となる. 実験結果から ESPRESSO を用いた簡単化により以前の実装「SyN」に比べて論理和の数を大幅に削減できていることが確認できる. 「DC」列と「ESP 近似」列の比較によりドントケアの設定で, より簡単化ができていることが確認できる. ESPRESSO の厳密解法では 7 次以上の結果が得られていないが, 6 次以下の結果により近似解法で厳密解に近い結果が得られることが期待できる.

図 2 は, 3 次の場合の ESPRESSO コマンドに対する入出力ファイルである. 入力ファイルの 5 行目以降で入力とする論理式表現を定義し, 出力ファイルの 6 行目以降で簡単化された論理式を表す. ハイフンは対応する論理変数が 0 でも 1 でも良いことを表し, 行末の 1, 2 はそれぞれ, 入力値が真とドントケアであることを表す. 各変数は $x_0y_0, x_1y_1, x_2y_2, x_3y_3$ がそれぞれ, s_1, s_0, c_2, c_1 を表す. 例えば 19 行目は, $s_1 < 0 \wedge s_0 = 0 \wedge c_2 > 0 \wedge c_1 > 0$ を表し, このとき, $V_0(\text{SH}(f)) = 0 < 1 = V_\infty(\text{SH}(f))$ なのでドントケアとして設定している.

19 行目, および 27 行目は $V_0(\text{SH}(f)) < V_\infty(\text{SH}(f))$ であること, 30 行目から 33 行目は xy が未使用であることからドントケアを設定し, 28 行目, および 36 行目から 37 行目は補題 12, 34 行目から 35 行目は補題 11, 38 行目から 39 行目は補題 9, 40 行目から 41 行目は補題 8 を満たさないためドントケアとして設定している.

22 個の積項で表される論理式が, 以下の等価で 4 個の積項からなる論理式に簡単化できた.

$$(s_1 < 0 \wedge s_0 > 0) \vee (s_1 < 0 \wedge c_1 < 0) \vee (s_0 < 0 \wedge c_1 < 0) \vee (c_2 \geq 0 \wedge c_1 \geq 0)$$

$f(x) = x^3 + ax^2 + bx + c$ とすると, Sturm-Habicht 列は $\text{SH}_3 = f(x)$, $\text{SH}_2 = f'(x) = 3x^2 + 2ax + b$, $\text{SH}_1 = (2a^2 - 6b)x + ab - 9c$, $\text{SH}_0 = -4b^3 + a^2b^2 - 4a^3c + 18bac - 27c^2$ なので,

$$\begin{aligned} \forall x (x \geq 0 \rightarrow f(x) > 0) \leftrightarrow^{QE} & c > 0 \wedge ((2a^2 - 6b < 0 \vee 2a^2 - 6b = 0 \wedge ab - 9c < 0) \wedge \text{SH}_0 > 0 \vee \\ & (2a^2 - 6b < 0 \vee 2a^2 - 6b = 0 \wedge ab - 9c < 0) \wedge ab - 9c < 0 \vee \\ & \text{SH}_0 < 0 \wedge ab - 9c < 0 \vee \\ & b \geq 0 \wedge ab - 9c \geq 0) \end{aligned}$$

となる.

SyNRAC の以前の実装では積項数が 17 個だったのに対し, 提案手法では積項数が 4 個と簡単化が実現できているが, さらに簡単化する余地があることがわかっている. 例えば, 3 次の結果における最初の積項から得られる条件 $c > 0 \wedge ((2a^2 - 6b < 0 \vee 2a^2 - 6b = 0 \wedge ab - 9c < 0) \wedge \text{SH}_0 > 0)$ を満たす実数は存在しないので, さらに簡単化できる. 3 節で示した SDC を満たす多項式に対する Sturm-Habicht 列に対する条件は, s_k または c_k が 0 になる部分に着目しており, 0 を含まない場合に不要な符号列を削減することは今後の課題である.

表 2 の最後の 3 列 (SDC 近似) は Sturm-Habicht 列に現れる多項式 SH_k の k 次の係数と定数項が 0 の場合をドントケアとして扱った場合の結果である. つまり, すべての SH_k が正則で $c_k \neq 0$ と仮定した結果になる. 「変数」は ESPRESSO に与える論理変数の数, 「厳密」は ESPRESSO の厳密解法を用いて簡単化した積項の数, 「時間」は ESPRESSO の厳密解法での実行時間 (秒) を表す. 描画結果のみを利用するなど正確な論理式を得る必要がない場合, かつ, すべての k に対して SH_k の k 次の係数と定数項が恒等的に 0 でないことがわかっている場合には, 上記の制限を加えても問題がないことがある. この場合には, s_k および c_k が 0 となる場合を考慮する必要が無いので, 論理変数としては次数 n に対して, $2n - 2$ となり, 計算に必要な厳密解法でも時間は短く, より簡単な結果が得られた. 別稿 [6] での実験結果では, この方法で生成される論理式を下に実行可能領域を描画した.

6 まとめ

実用上重要なクラスである SDC に対する専用の QE の高速化のために出力となる論理式の簡単化を行った. そのために, Sturm-Habicht 列が満たす必要条件を示し, 真となる符号条件の数を削減し, 簡単化には論理簡単化手法 ESPRESSO を利用した. このときドントケアの設定でより簡単な結果が得られることが確認できた.

今後の課題としては, SDC を満足する多項式が満たす必要十分条件を求めることと, 7 次以上で得られているのは近似解であるため, BDD を用いた厳密解法による簡単化との比較することが考えられる.

謝 辞

SDC の出力を正則な場合に限定することで, 出力の論理式をより簡単化する手法は, 立教大学の横山和弘教授のコメントを元に行いました. 貴重なコメントを頂いた横山教授に深く感謝いたします.

<pre> 1 .lib x0 y0 x1 y1 x2 x3 y3 2 .i 8 3 .o 1 4 .ob f0 5 01010101 1 6 01010001 1 7 01011001 1 8 01011000 1 9 01011010 1 10 00011000 1 11 10010101 1 12 10010001 1 13 10011001 1 14 10011000 1 15 10011010 1 16 01000101 1 17 01000001 1 18 01001001 1 19 01001010 2 20 00001000 1 21 10001010 1 22 01100101 1 23 01100001 1 24 01101001 1 25 01100100 1 26 01100110 1 27 01101010 2 28 00100100 2 29 10101010 1 30 11----- 2 31 --11---- 2 32 ----11-- 2 33 -----11 2 34 00----1- 2 35 00----- 2 36 --101000 2 37 --010100 2 38 ----0010 2 39 ----0000 2 40 1000--00 2 41 0100--00 2 42 .e </pre>	<pre> .lib x0 y0 x1 y1 x2 x3 y3 .i 8 .o 1 .ob f0 .p 4 -11----- 1 -1-----1 1 ---1---1 1 -----0-0 1 .e </pre>
---	---

図 2: 3 次の問題に対する ESPRESSO コマンドの入力ファイル (左) と出力ファイル (右)

参 考 文 献

- [1] Espresso. <http://embedded.eecs.berkeley.edu/pubs/downloads/espresso/>.
- [2] H. Anai and S. Hara. A parameter space approach to fixed-order robust controller synthesis by quantifier elimination. *International Journal of Control*, 79(11):1321–1330, 2006.
- [3] R. K. Brayton, A. L. Sangiovanni-Vincentelli, C. T. McMullen, and G. D. Hachtel. *Logic Minimization Algorithms for VLSI Synthesis*. Kluwer Academic Publishers, Norwell, MA, USA, 1984.
- [4] L. González-Vega, T. Recio, H. Lombardi, and M.-F. Roy. *Sturm-Habicht sequences determinants and real roots of univariate polynomials*, pp. 300–316. Texts and Monographs in Symbolic Computation. Springer, 1998.
- [5] R. Loos and V. Weispfenning. Applying linear quantifier elimination. *The Computer Journal*, 36(5):450–462, 1993.
- [6] Y. Matsui, H. Iwane, and H. Anai. Two controller design procedures using SDP and QE for a power supply unit. 数式処理研究と産学連携の新たな発展, 2013.
- [7] 穴井, 横山. QE の計算アルゴリズムとその応用 – 数式処理による最適化. 東京大学出版会, 8 2011.