計算機アーキテクチャ　103－18
設　計　自　動　化　69－18
(1993. 12. 17)

代数的仕様からのマイクロプログラム自動合成

木下貴史† 直井徹†† 今井正治†

†豊橋技術科学大学情報工学系 †† 岐阜大学工学部

〒441 愛知県豊橋市天伯町字雲雀ヶ丘1-1
Tel: (0532)47-0111, ext. 599
E-mail:kinosita@imaisun.tutics.tut.ac.jp

あらまし　Zhu,Johnsonにより提案されたマイクロプログラム合成問題に対し，著者らが以前に提案したサブゴール生成に基づく手続きの適用範囲を拡張する．これにより，仕様の等式理論がユニタリ単一化理論である場合にのみ適用可能であったサブゴール生成式の探索手続きが，最汎意味単一化子が存在しなくても意味単一化子の最小集合が存在すれば利用可能になる．また，メモリとCPUからなるハードウェアの仕様を与え，拡張されたサブゴール生成式手続きを，この仕様のもとで要求された計算を実現するプログラムを合成する問題に適用する．

和文キーワード　抽象データ型，意味単一化，マイクロプログラム，逐次化問題，ナローイング

Automatic Microprogram Synthesis from Algebraic Specification of Hardware

Takafumi KINOSHITA† Tohru NAOI†† Masaharu IMAI†

†Toyohashi University of Technology †† Gifu University

1-1 Hibarigaoka Tenpaku-cho Toyohashi-shi Aichi,441, Japan
Tel: (0532)47-0111, ext. 599
E-mail:kinosita@imaisun.tutics.tut.ac.jp

Abstract　This paper describes the extended subgoaling procedure for automatic microprogram synthesis , based on the the Serialization Problem, which is formalized by Zhu and Johnson. While our previous subgoaling procedure was applicable only when the given equational theory is unitary unification, e.g. the minimal set of semantic unifier is always singleton, the extended procedure proposed in this paper is applicable even when the minimal set of semantic unifier is not singleton. This procedure is applicable to wider class of problems. As an example of such problems, a program synthesis under specification of a hardware with a memory is given.

英文 key words　abstract data types, semantic unification, microprogram, serialization problem, narrowing

1 はじめに

Zhu,Johnson は抽象データ型にもとづいたアーキテクチャ仕様の記述法を提案し，その上でマイクロプログラムの自動合成に関する問題を逐次化問題 (serialization problem) として定式化した [1]．これに対し著者らは，意味単一化可能性検査を用いて解の探索空間の枝刈りを行う手続きを提案し，その有効性を実験により確かめた [3]．

さらに，探索中に得られた意味単一化子を用いて，もとの逐次化問題に対するサブゴールを生成しながら探索を行う方式により，意味単一化処理の効率化が期待できることを示した [4]．しかし，この方式が適用できるのは，仕様の等式理論 Eがユニタリ単一化理論，すなわち最汎意味単一化子が常に存在する理論である場合に限られていた．ブール環，有限体，あるいは多値論理といった，ハードウェアにおける演算の表現に適した代数系がユニタリ単一化理論で表されることが知られている [2]．しかし，仕様記述の利便性のためには，より多くのデータ型が扱えることが望ましい．

本稿ではまず，最汎意味単一化子が存在しなくても，意味単一化子の最小集合が存在すれば利用可能な，拡張されたサブゴール生成式逐次化手続きを示す．また，拡張されたサブゴール生成式手続きが適用できる仕様記述の例として，メモリと CPU からなる簡単なハードウェアの仕様を与える．この仕様は，ハードウェアがメモリに格納された命令列に従って動作する様子を記述したもので，メモリ演算といった，ブール環などが持たない演算を含んでいる．次に，上述の仕様記述のもとで所望の計算を実現するための命令列を求める問題を定め，提案した逐次化手続きを適用してそのふるまいを検討する．

2 準備

2.1 アーキテクチャ仕様記述法と逐次化問題

Zhu,Johnson の仕様記述法を以下に示す (文献 [3] も参照)．$\mathcal{S}$をソートの集合，$\mathcal{F}$を関数記号集合，$\mathcal{X}$を変数記号集合とする．$\mathcal{F}$のすべての要素には，ランクと呼ばれるソートの列が対応する．$\mathcal{F},\mathcal{X}$による項の集合を $\mathcal{T}(\mathcal{F},\mathcal{X})$ と表わす．$t \in \mathcal{T}(\mathcal{F},\mathcal{X})$ に含まれる変数記号の集合を $\mathcal{V}(t)$ と表す．等式は項の対 $t = s$ である．Eを等式の集合とする．二つの項 $t, s \in \mathcal{T}(\mathcal{F},\mathcal{X})$ が等式集合 Eに対する等式理論を法として同値であることを $t =_E s$ と表す．4 項組 $\langle \mathcal{S},\mathcal{F},\mathcal{X},E\rangle$ を抽象データ型仕様という．

$\mathcal{X}$と素な変数記号の有限集合 $\mathcal{R} = \{r_1, r_2, \ldots, r_N\}$ をレジスタ変数記号集合と呼ぶ．代入$\delta : \mathcal{R} \to \mathcal{T}(\mathcal{F},\mathcal{R})$ をデータパス代入と呼ぶ．データパス代入の任意の有限集合を制約集合と呼ぶ．$\mathcal{R},\Delta$をそれぞれレジスタ変数記号集合および制約集合とする．6 項組 $\langle \mathcal{S},\mathcal{F},\mathcal{X},E,\mathcal{R},\Delta\rangle$ を，アーキテクチャ仕様という．

代入の合成$\sigma \circ \rho$を$\sigma \circ \rho(t) = \sigma(\rho(t))$ と定義する．また，$\mathcal{D}(\sigma) = \{x | \sigma(x) \neq x\}$, $\mathcal{I}(\sigma) = \bigcup_{x \in \mathcal{D}(\sigma)} \mathcal{V}(\sigma(x))$ とする．$\mathcal{D}(\sigma) = \{x_1, \ldots, x_n\}$ で$\sigma(x_i) = t_i$であるσを $\langle x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\rangle$ と表す．σの変数記号集合 Vへの制限を$\sigma|_V$と表す．項の E同値性を次のように，代入の同値性へ拡張する．すなわち，二つの代入σ，ρが変数記号集合 Vの任意の要素 vに対し，$\sigma(v) =_E \rho(v)$ となるとき，$\sigma =_E \rho[V]$ と表す．Zhu, Johnson によるマイクロプログラム合成問題は次のように定義される．

定義 1 アーキテクチャ仕様 $\langle \mathcal{S},\mathcal{F},\mathcal{X},E,\mathcal{R},\Delta\rangle$ のもとで，$\mathcal{R}$ の部分集合 $\mathcal{Q}$ とデータパス代入$\gamma = \langle r_1 \leftarrow t_1, \ldots, r_m \leftarrow t_m\rangle$ $(r_i \in \mathcal{Q}; r_i \neq r_j (i \neq j))$ に対し，

$$\delta_1 \circ \delta_2 \circ \cdots \circ \delta_n =_E \gamma[\mathcal{Q}] \qquad (1)$$

をみたす制約集合の要素$\delta_1, \delta_2, \ldots, \delta_n$が存在するかどうかを決定する問題を逐次化問題という． ■

この問題は一般に決定不能 (準決定可能) で，Eが空のとき，また仕様の基底代数がブール環のときには決定可能である [1] [5]．

2.2 意味単一化

項対 (t, s) の集合 Pに対し，代入σが存在して $\sigma(t) =_E \sigma(s)$ $((t, s) \in P)$ となるとき，σを Pの E(意味) 単一化子と呼び，Pは E(意味) 単一化可能であるという．

二つの代入 σ，ρと変数記号集合 Vについて代入 λが存在し，$\rho =_E \lambda \circ \sigma[V]$ となることを $\sigma \leq_E \rho[V]$ と表し，$\sigma \leq_E \rho[V]$ かつ$\rho \leq_E \sigma[V]$ であることを $\sigma \equiv_E \rho[V]$ と表す．$\mathcal{V}(P) = \bigcup_{(t,s) \in P} (\mathcal{V}(t) \cup \mathcal{V}(s))$ とする．Pの E単一化子の集合Σは，次の二つの条件

1. Pの任意の E単一化子ρに対し，$\sigma \in \Sigma$が存在して$\sigma \leq_E \rho[\mathcal{V}(P)]$.

2. 任意のσ, $\rho \in \Sigma$に対し, $\sigma \leq_E \rho[V]$ ならば$\sigma = \rho$ $(\sigma, \rho \in \Sigma)$.

を満たすとき, E単一化子の最小集合と呼ばれる. E単一化可能な項対集合に対し, ただ1つの要素からなる最小集合が存在する等式理論をユニタリ単一化理論という. また, 同様に, 有限個の要素からなる最小集合が存在する理論を有限単一化理論という. E単一化可能な項対集合に対し, 最小集合が常に存在するとは限らず, また, 存在するならば$=_E$ $[V]$の意味で一意的であることが知られている (文献 [7] などを参照).

3 サブゴール生成式手続きの拡張

まず, サブゴール生成式手続きの基本操作である, 代入の導出を定める. 任意の $r \in \mathcal{R}$ に対して $\mathcal{V}(\sigma(r)) \cap \mathcal{R} = \emptyset$ である代入σを, $\mathcal{R}$ と交わらない代入と呼ぶ. Pの E単一化子をσとし, すべてのレジスタ変数記号 r_iを $\mathcal{I}(\sigma) \cup \mathcal{R} \cup \mathcal{V}(P)$ に含まれない変数記号 v_iに一意的に付け替える代入を $rename_\sigma$ とすると, $rename_\sigma \circ \sigma$は Pの $\mathcal{R}$ と交わらない E単一化子であることが示せる. また, $rename_\sigma \circ \sigma \leq_E \sigma[\mathcal{V}(P)]$ かつ$\sigma \leq_E rename_\sigma \circ \sigma[\mathcal{V}(P)]$ であることも示されるので, Pの E単一化子の最小集合が存在するとき, $\mathcal{R}$ と交わらない E単一化子の最小集合も存在する.

定義 2 ρを $\mathcal{R}$ と交わらない代入, $\mathcal{P}$を $\mathcal{R}$ の部分集合とし, δを制約集合の要素とする. 項対集合 $P = \{(\rho(p), \delta(p)) | p \in \mathcal{P}\}$ が E単一化可能で, その $\mathcal{R}$ と交わらない E 単一化子の最小集合を $MSU(\rho, \delta, \mathcal{P})$ と表すと, 任意の$\sigma \in MSU(\rho, \delta, \mathcal{P})$ に対して

$$\rho \Rightarrow_{\mathcal{P}, \delta} \sigma \qquad (2)$$

と表し, **δによる$\mathcal{P}$上の導出**と呼ぶ. ■

次に, 導出操作と逐次化問題の関係を示す. 各レジスタ変数記号 $r_i \in \mathcal{R}$ に一意に対応する自由定数記号 c_iを用意し, 代入 $\langle r_1 \leftarrow c_1, r_2 \leftarrow c_2, \ldots, r_N \leftarrow c_N \rangle$ を regと表す. このとき, 次の性質が導かれる.

定理 1 等式理論 Eに対し, E単一化子の最小集合が存在するとき, 仕様 $\langle \mathcal{S}, \mathcal{F}, \mathcal{X}, E, \mathcal{R}, \Delta \rangle$ のもとで, $\mathcal{Q}$, γが式 (1) を満たすならば, 導出列

$$reg \circ \gamma \Rightarrow_{Q_0, \delta_n} \zeta_1 \Rightarrow_{Q_1, \delta_{n-1}} \cdots \Rightarrow_{Q_{n-1}, \delta_1} \zeta_n \qquad (3)$$

$$Q_0 = \mathcal{Q}$$

$$Q_i = \bigcup_{q \in \mathcal{Q}} \mathcal{V}(\delta_{n-(i-1)}(\delta_{n-(i-2)}(\cdots \delta_n(q) \cdots)))(i > 0)$$

が存在する.

(略証) 付録を参照. ■

定理1より, 次のような手続きが得られる. すなわち, 導出列を非決定的に生成し, 各導出ステップ iで $\delta_i(\delta_{i-1}(\cdots \delta_1(\overline{r}) \cdots))$ と $\overline{w}$が E同値かどうかを検査する手続きである. このとき, 導出列が生成できなければ探索は打ち切られるので, 探索空間の枝刈りが可能である. 著者らが文献 [4] で示した手続きは, 最汎 E単一化子を導出する操作を用いて探索を行うものであった. 従って, 文献 [4] のサブゴール生成式手続きは定理1を用いて枝刈りを行う手続きの特殊な例である. この意味で, 定義2の導出操作を用いた手続きは, 導出される E単一化子が最汎性を持つという条件を取り除き, 著者らの以前の手続きを一般化したものとなっている.

各ζ_iは, もとの導出ゴールγに対する, 各導出ステップでの部分的な導出ゴールとみることができる. このように, もとのゴールγ を部分ゴールへ分解する探索方式では, ゴールをγで一定とする探索方式にくらべて探索中の E単一化処理が, 分解法に基づく E単一化の手法 ([8] などを参照) を用いることにより効率化される可能性がある.

手続きの詳細を表1に示す. 恒等代入をϵ と表す. 仕様の等式理論 Eに対し, E単一化子の最小集合を求める手続きが存在することを仮定する[1]

この手続きでは, すべてのデータパス代入, および E単一化子の最小集合のすべての要素について非決定的な探索が行われることに注意されたい. 従って, 文献 [3] の逐次化手続きにくらべて探索空間は大きくなるが, 解の候補に対する枝刈りの能力は同じであることが示せる. すなわち,

定理 2 E単一化可能な項対集合に対し, 最小 E単一化子が存在すると仮定する. 導出列 (3) が存在するとき, かつそのときに限り項対集合 $P_i = \{(\delta_{n-(i-1)}(\delta_{n-(i-2)}(\cdots \delta_n(q) \cdots)), \gamma(q)) | q \in \mathcal{Q}, i = 1, \ldots, n\}$ は E単一化可能である.

(略証) 証明の方針のみ示す. 導出列 (3) の

[1] E単一化は一般に決定不能 (準決定可能) であるから, E単一化子の最小集合を求める手続きが停止するとは限らない. ただし, 演算公理を表す等式集合が完備な項書換え系とみなせて, かつ基底ナローイングが停止するような仕様では, E単一化子の最小集合を有限時間内に求めることができる [9].

$serialize(\mathcal{Q}, \gamma)$
　$subgoaling(\mathcal{Q}, \gamma, \epsilon)$ を返す.

$subgoaling(Q, \zeta, \eta)$

1. $\eta =_E \gamma$ [Q] ならば「成功」を返す.

2. $\delta \in \Delta$を非決定的に 1 つ選ぶ.

◇ $MSU(\zeta, \delta, Q)$ が空でなければ,
　$\sigma \in MSU(\zeta, \delta, Q)$ を非決定的に 1 つ選び,
　$subgoaling(\bigcup_{q \in Q} \mathcal{V}(\delta(q)), \sigma, \delta \circ \eta)$ を返す.

◇ $MSU(\zeta, \delta, Q)$ が空ならば,「失敗」を返す.

表 1: サブゴール生成式逐次化手続き

各ζ_iが $\{(\delta_i(\delta_{i-1}(\cdots\delta_1(q)\cdots)), \gamma(q)) | q \in \mathcal{Q}, i = 1, \ldots, n\}$ の E単一化子であることは容易に示すことができる. また, 逆の場合, 定理 1 の証明と同様の議論により, 導出列 (3) の存在が示される. ■

4　仕様記述の例と命令列の合成

本節では, 拡張されたサブゴール生成式手続きの適用例について述べる. まず, メモリを持つデータパスの仕様記述を与え, この仕様記述のもとで命令列を求める問題を定める. 次に, この問題に対する上述の逐次化手続きの振る舞いを検討する.

4.1　メモリの仕様記述

ALU, レジスタ r, プログラムカウンタ pc および 4 ワードのメモリからなる図 1 のデータパスを考える. 命令は 1 オペランド形式とし, オペランドにはデータが直接指定されるものとする. 簡単のため ALU は加算だけができるものとする. データ, アドレスは自然数として表されるものとし[2], このハードウェアに対するアーキテクチャ仕様 $\mathcal{M}$ を以下のように与える.

ソートの集合 $\mathcal{S}_\mathcal{M}$を

$$\{memory, nat, constructor, bool\}$$

関数記号集合 $\mathcal{F}_\mathcal{M}$を

$$\{array, mnil, ap, 0, s, nil, read, if, zero, true, false, add, inc, diff\}$$

とする. 各関数記号のランクをそれぞれ,

$array : \langle nat, nat, memory\rangle$	$mnil : \langle memory\rangle$
$ap : \langle constructor, nat\rangle$	$nil : \langle nat\rangle$
$0 : \langle constructor\rangle$	$s : \langle constructor\rangle$
$read : \langle nat, memory, nat\rangle$	$zero : \langle nat, bool\rangle$
$if : \langle bool, nat, nat, nat\rangle$	$true : \langle bool\rangle$
$false : \langle bool\rangle$	$add : \langle nat, nat, nat\rangle$
$diff : \langle nat, nat, nat\rangle$	$inc : \langle nat, nat\rangle$

とする. $zero$ は自然数がゼロかどうかを判定する関数, add は自然数の加算, inc は自然数のインクリメント, $diff$は二つの自然数の減算の絶対値を計算する関数である.

シグニチャ$\langle \mathcal{S}_\mathcal{M}, \mathcal{F}_\mathcal{M}\rangle$ を用いると, 例えば自然数 "2" は $ap(0, ap(s, ap(s, nil)))$ と表される. メモリは,

$$array([アドレス], [データ], array(\cdots))$$

という形式のリストとして表現される. 例えば 4 ワードのメモリは

$$array(ap(0, nil), D_0),$$
$$array(ap(0, ap(s, nil)), D_1),$$
$$array(ap(0, ap(s, ap(s, nil))), D_2),$$
$$array(ap(0, ap(s, ap(s, ap(s, nil)))), D_3, mnil))))$$

と表される. ここで D_0, D_1, D_2, D_3はソート nat の変数である.

等式集合 $E_\mathcal{M}$を表 2 のように与える.

式 (4), (5) は, アドレス A とアドレス Bが同じなら, Bで示されるワードに値を書き込み (あるいは値を読み出し), そうでなければさらに先のメモリに対して書き込み (あるいは読み出し) 操作を続けることを表している. これらの等式を左辺から右辺へ方向付けることにより, 完備な項書換え系が得られる. 従って, この公理に対して基底ナローイングを用いた E単一化手続き [9] が利用できる.

データの表現に整数を用いるので, 命令も 1 つの整数として表される. このとき, 扱える整数の範囲を限定すれば, 2 進数で表された命令と同様に, 適当な処理によってオペコードとオペランドを取り出すことができるが, ここでは簡単のため, メモリのす

[2] ここでは, 仕様記述を簡単にするためにアドレスやデータを自然数として扱うが, これらを 2 進数とみなしてビット列により表現することも可能である (文献 [3] を参照).

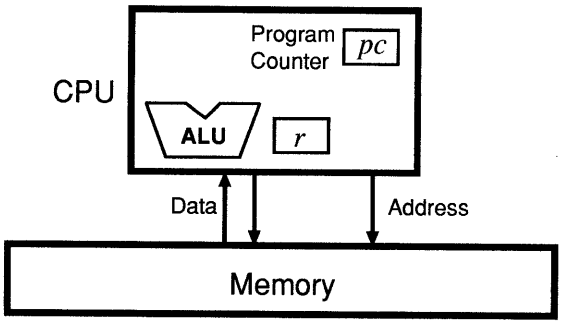

図 1: メモリを持つ簡単な CPU

べての領域は図 2 のようにオペコードが格納される領域 M_c と，オペランドが格納される領域 M からなるものとして表す．データのみが格納される領域ではオペコードの部分に，データであることを示す適当なコードを格納しておき，プログラムとして読み込まれないようにする．また，データはオペランド部分に格納する．メモリの各ワードに対応する領域をそれぞれ，レジスタ変数記号 $Mc0$，$Mc1$，$Mc2$，$Mc3$，および $M0$，$M1$，$M2$，$M3$ で表す．

Address	OPcode	Operand
0	Mc0	M0
1	Mc1	M1
2	Mc2	M2
3	Mc3	M3

Memory

図 2: 仕様 $\mathcal{M}$ におけるメモリ表現

制約集合 $\Delta_{\mathcal{M}}$ を表 3 のように与える．$\Delta_{\mathcal{M}}$ は 1 つの要素 $\delta 1$ だけを持つ．このデータパス代入は，命令のオペコードを格納するメモリ Mc からオペコードを読み出して，これが $ap(0, ap(s, nil))$ ならば加算を実行し，それ以外のコードであれば何もしないといった，ハードウェアの動作を表現している．ここでは CPU の機能を単純にしているので仕様記述も単純であるが，命令に応じた条件分岐の記述を用いればパイプライン化されたプロセッサの動作記述なども可能である [6]．

図 1 のデータパスのアーキテクチャ仕様 $\mathcal{M}$ を $\langle \mathcal{S}_{\mathcal{M}}, \mathcal{F}_{\mathcal{M}}, \mathcal{X}, E_{\mathcal{M}}, \{r, ra, M_0, M_1, M_2, M_3, Mc_0, Mc_1, Mc_2, Mc_3\}, \Delta_{\mathcal{M}} \rangle$ と定める．

データパス代入 $\delta 1$ に現れるメモリは，常に有限リストとして表されているから，$E_{\mathcal{M}}$ に対する項書換え系を用いた基底ナローイングは停止することが示せる．従って，逐次化手続き中で現れる項対集合の $E_{\mathcal{M}}$ 単一化子の最小集合は，基底ナローイングを用いた E 単一化手続きにより求めることができる [9]．

4.2 命令列の合成

表 3 のように，命令に応じたハードウェアの動作仕様を与えると，逐次化問題の枠組みを用いて命令列を合成する問題を扱うことができる．ここでは，命令列合成問題として次のような問題を扱うことにする．

> 与えられた動作仕様，およびオペランドの初期値のもとで，所望の計算を行うためのオペコードの列が存在するかどうかを決定し，存在するならばそのオペコードをメモリ上に書き込む．

この問題を逐次化問題の枠組みで扱うためには，メモリの任意のアドレスに，任意のオペコードを書き込むようなデータパス代入を制約集合に追加すればよい．逐次化手続きがこれらのデータパス代入のいずれかを選択することにより，オペコード列の探索が行われる．

ステップ	解候補の数
0	1
1	10
2	90
3	800
4	7040

表 4: 探索中の解候補の数と実行時間

図 1 のデータパスについて，それぞれのメモリ記述法を用いたオペコード列の探索例を考えよう．簡単のため，命令は加算と nop の 2 つだけとし，それぞれを $ap(0, ap(s, nil))$，$ap(0, nil)$ のコードで表す．表 3 で示した仕様 $\mathcal{M}$ の制約集合に，8 つのデータパス代入

```
[ Mc0 <- ap(0,nil) ]                (δ 2)
[ Mc0 <- ap(0,ap(s,nil)) ]          (δ 3)
[ Mc1 <- ap(0,nil) ]                (δ 4)
[ Mc1 <- ap(0,ap(s,nil)) ]          (δ 5)
[ Mc2 <- ap(0,nil) ]                (δ 6)
[ Mc2 <- ap(0,ap(s,nil)) ]          (δ 7)
[ Mc3 <- ap(0,nil) ]                (δ 8)
[ Mc3 <- ap(0,ap(s,nil)) ]          (δ 9)
```

を追加する．それぞれ，メモリ Mc の各ワードへコードを書き込むデータパスである．さらに，プログラムカウンタの初期値を $ap(0, nil)$ とし，メモリ Mのアドレス $ap(0, nil)$ および $ap(0, ap(1, nil))$ の初期値はレジスタ rの初期値と同じ値とする．

この仕様のもとで，rに $add(add(r, r), r)$ を格納するオペコード列を合成する問題を考える．逐次化手続き中の E単一化は基底ナローイングにより行われるものとする．逐次化手続きを実行した結果，長さ 4 の解が求められた．各探索ステップでの解候補の数を表 4 に示す[3]．解は $(\delta 5), (\delta 3), (\delta 1), (\delta 1)$ であった．この解は $Mc0$ と $Mc1$ に加算のオペコードを格納することにより，所望の演算ができることを示している．

[3]サブゴール生成式の手続きはまだ実現されていないので，この解候補の数は，文献 [3] の手続きにより求めたものを掲載した (先に述べた定理 2 より，本稿で述べたサブゴール生成式逐次化手続きと，文献 [3] の手続きで生成する解候補の数は等しい)．

5 むすび

Zhu, Johnson により示されたマイクロプログラム合成問題に対し，著者らが以前に提案したサブゴール生成式準手続きの適用範囲を拡張し，最汎意味単一化子が存在しない場合でも，意味単一化子の最小集合が存在すれば利用可能な，拡張されたサブゴール生成式逐次化手続きを示した．また，拡張されたサブゴール生成式手続きが適用できる仕様記述の例として，メモリと CPU からなる簡単なハードウェアの仕様を与た．さらに，この仕様記述のもとで，命令列を求める逐次化問題を与え，本手続きを適用した．

問題をサブゴール化して解く方式は，探索中の意味単一化処理を効率化する上で有効であり，その適用範囲を広げることは逐次化手続きをより実用的なものとするために必要と考えられる．本稿では最小集合が存在する場合に対して適用できるように拡張したが，さらに最小集合が存在するとは限らない場合についても適用できるようにする予定である．本稿で提案した手続きは意味単一化子の最小集合すべてについて非決定的に探索するが，導出される意味単一化子をより制限されたものとすることができれば，手続きの効率はさらに向上すると思われる．

今後は，サブゴール生成式逐次化手続きの実現を行い，実行効率についての議論を可能にする．

また，命令列合成問題の定式化を厳密に行い，逐次化問題の枠組みをプログラム (命令列) の合成へ応用する予定である．本稿で与えたメモリ記述の方法はメモリの実体を有限リストとして表すものであったが，この方法では大きなメモリの記述に不便である．従って，プログラム合成問題の定式化とともに，メモリを表現する，より効率的な方法についても検討したい．

謝辞

日頃ご指導頂く岐阜大学工学部 後藤宗弘教授，名古屋大学工学部 稲垣康善教授に感謝いたします．また，本研究を進めるに当たりご討論頂いた豊橋技術科学大学 塩見彰睦助手，鶴岡高専 佐藤淳助手，豊橋技術科学大学 VLSI 設計研究室の諸兄，九州大学総合理工学研究科 安浦寛人教授，ならびに関連文献の調査に際してお世話になった北陸先端科学技術大学院大学 外山芳人教授，筑波大学電子・情報工学系 Aart Middeldorp 講師，NTT 基礎研究所 小川

瑞人氏に感謝いたします.

参考文献

[1] Z. Zhu and S. D. Johnson : "An Algebraic Characterization of Structural Synthesis for Hardware", Proc. of the IFIP Int. Workshop on Applied Formal Method for Correct VLSI Design, pp.261-269, North-Holland(1989).

[2] T. Nipkow : "Unification in Primal Algebras", LNCS, Vol.**299**, pp.117-131, Springer-Verlag(1988).

[3] 木下貴史, 直井徹, 今井正治 : "E同一化を用いたマイクロプログラム自動合成手続き", 信学論(D-I), 掲載予定 (1993-11).

[4] 木下貴史, 直井徹, 今井正治 : "意味単一化子を利用したマイクロプログラム自動合成手続き", 情報処理学会 DA シンポジウム'93(1993).

[5] 木下貴史, 直井徹, 今井正治 : "ブール代数上の逐次化問題の決定可能性", 1993 信学春期全大, A-105.

[6] 庄内亨, 清水嗣雄 : "実行命令数に関する数学的帰納法と再帰帰納法による形式検証", 信学技報, **VLD92**-77(1993).

[7] J. H. Siekmann : "Unification Theory", J. Symbolic Computation, Vol.**7**, pp.207-274(1989).

[8] J. H. Gallier and W. Snyder : "Complete Sets of Transformatons for General E-Unification", Theoretical Computer Science, Vol.**67**, pp.203-260(1988).

[9] J. -M. Hullot : "Canonical Forms and Unification", Proc. of the 5th Conf. on Automated deduction, pp.318-334(1980).

[10] A. Middeldorp and E. Hamoen : "Counter Examples to Completeness Results for Basic Narrowing", CWI report CS-R9154(1991).

付　　録

定理 1 の証明

はじめに, 次の補題を示す.

補題 1 等式理論 Eに対し, E単一化子の最小集合が存在するとき, 仕様 $\langle \mathcal{S}, \mathcal{F}, \mathcal{X}, E, \mathcal{R}, \Delta \rangle$ のもとで $\mathcal{Q}$, γが式 *(1)* を満たすならば, $k = 1, \ldots, n$ について, 項対集合 $\{(\delta_{n-(k-1)}(q), reg \circ \delta_1 \circ \delta_2 \circ \cdots \circ \delta_{n-(k-1)}(q)) | q \in Q_k\}$ は E単一化可能である.

(証明)　regは定数の代入であるから, 任意の代入ρ に対し, $\rho \circ reg \circ \delta_1 \circ \delta_2 \circ \cdots \circ \delta_{n-(k-1)} = reg \circ \delta_1 \circ \delta_2 \circ \cdots \circ \delta_{n-(k-1)}$である. このことと逐次化問題の定義より示される. ■

従って, $k = 1, \ldots, n$ について, 補題 1 が成り立つ場合に$\zeta_k \leq_E reg \circ \delta_1 \circ \delta_2 \circ \cdots \circ \delta_{n-k}[Q_k]$ であるような導出列 (3) が常に存在することを示せば定理 1 が示される. これを導出ステップ数 $k = 1, \ldots, n$ に関する帰納法により示す. $\sigma_k = reg \circ \delta_1 \circ \delta_2 \circ \cdots \circ \delta_{n-k}$とする.

〈$k = 1$ **の場合**〉 導出の定義および最小集合の定義より, $MSU(\gamma, \delta_n, \mathcal{Q})$ には$\zeta_1 \leq_E \sigma_1[Q_1]$ を満たす E単一化子 ζ_1が含まれる.

〈$1 < k \leq n$ **の場合**〉 $i = 1, \ldots, k-1$ までの導出ステップに対し, 定理 1 が成り立つとする. このとき, ある代入ηが存在して$\eta \circ \zeta_{k-1} =_E \sigma_{k-1}[Q_{k-1}]$ となり, ζ_{k-1}は $\mathcal{R}$ と交わらない代入だから $(reg \circ \delta_1 \circ \cdots \circ \delta_{n-k}) \circ \eta|_{\mathcal{D}(\eta)}$により, 項対集合 $\{(\zeta_{k-1}(r), \delta_{n-(k-1)}(r)) | r \in Q_{k-1}\}$ は E単一化可能である. この項対集合の E単一化子の最小集合には必ず

$$\zeta_k \leq_E (reg \circ \delta_1 \circ \cdots \circ \delta_{n-k})[Q_k \cup \bigcup_{r \in Q_{k-1}} \mathcal{V}(\zeta_{k-1}(r))]$$

なる E単一化子ζ_kが含まれるが, $\leq_E [V]$ の定義より, 任意のσ, ρ, および Vに対し, $\sigma \leq_E \rho[V]$ ならば $W \subseteq V$について $\sigma \leq_E \rho[W]$ だから,

$$\zeta_k \leq_E (reg \circ \delta_1 \circ \cdots \circ \delta_{n-k})[Q_k]$$

なる E単一化子ξ_kが成り立つ. このことは, 導出列

$$\xi_{k-1} \Rightarrow_{Q_{k-1}, \delta_{n-(k-1)}} \xi_k$$

の存在が示している. ■

$$write(A,D,arry(X,Y,Z)) = if(eq(A,X),arry(X,D,Z),arry(X,Y,write(A,D,Z))) \quad (4)$$
$$read(A,arry(X,Y,Z)) = if(eq(A,X),Y,read(A,Z)) \quad (5)$$
$$eq(A,B) = zero(diff(A,B)) \quad (6)$$
$$if(true,X,Y) = X \quad (7)$$
$$if(false,X,Y) = Y \quad (8)$$
$$inc(ap(0,X)) = ap(0,ap(s,X)) \quad (9)$$
$$zero(ap(0,nil)) = true \quad (10)$$
$$zero(ap(X,ap(s,Z))) = false \quad (11)$$
$$diff(ap(X,Z1),ap(X,Z2)) = diff(Z1,Z2) \quad (12)$$
$$diff(nil,Z) = ap(0,Z) \quad (13)$$
$$diff(Z,nil) = ap(0,Z) \quad (14)$$

表 2: 等式集合 $E_{\mathcal{M}}$

```
[ r <- if( zero(
             diff(
                read( ra,
                  array(ap( 0, nil ),Mc0, array(ap(0,ap(s,nil)),Mc1,
                  array(ap( 0, ap(s,ap(s,nil))), Mc2,
                  array(ap(0,ap(s,ap(s,ap(s,nil)))),Mc3,mnil))))
                ),
                ap(0,ap(s,nil)) ) ),
           add( r,
               read( ra,
                 array(ap( 0, nil ),M0, array(ap(0,ap(s,nil)),M1,
                 array(ap( 0, ap(s,ap(s,nil))), M2,
                 array(ap(0,ap(s,ap(s,ap(s,nil)))),M3,mnil)))) )  ),
           r ),
  ra <- inc(ra) ]                                                    (δ 1)
```

表 3: 制約集合$\Delta_{\mathcal{M}}$