数理解析研究所講究録
1041 巻 1998 年 227-234

決定可能な高階単一化問題に関する研究

山田 敬三[1] (Keizo Yamada) 平田 耕一[2] (Koichi Hirata) 原尾 政輝[2] (Masateru Harao)

1998 年 2 月 4 日

あらまし

本論文では, 二階単一化問題が決定可能となるための条件について考察する. そのために, 関数変数の個数に着目し, 関数変数の個数を 1 に制限する. そして, この制限の下で, 与えられた二項のうちどちらか一方にだけ関数変数がちょうど一回現れる場合について, すべての単一化代入を求める手続きを示す. 次に, この条件を拡張し, 各項に高々一回, 関数変数が現れる場合について, 二階単一化問題の決定可能性を示す.

1 はじめに

定理証明系の証明機構や Prolog などのプログラミング言語における計算原理の一つとして, 単一化という操作が知られている. **単一化** (*unification*) とは, 与えられた二項 t, s に対して, 二項が構文的に等しくなる代入を見つける操作であり, このような代入が存在するとき t と s は**単一化可能** (*unifiable*) という. また, このときの代入を t と s の**単一化代入** (*unifier*) という. さらに, 与えられた二項に対して, 単一化代入が存在するか否かを決定する決定問題を**単一化問題** (*unification problem*) という.

一階項言語における単一化問題は決定可能であり, 最も一般的な代入である最汎単一化代入を求める手続きが知られている. ところが, 一般の高階単一化においては, もはや最汎単一化代入が存在するとは限らず, 単一化問題も決定不能である [5]. しかも, この決定不能性は, 高階言語の中でも最も単純な二階項言語に対しても成り立つ [2, 3]. 一方, 二階単一化問題については, その問題が決定可能となるような項言語の条件についても研究されており, 例えば, 与えられた二項に同じ変数が現れない二階項言語 [6], 関数定数を単項関数に制限した二階項言語 (*second-order monadic language*)[1] や, 一方の項には変数の現れない二階項言語 [5] では単一化問題が決定可能である.

本論文では, 関数変数の個数という点に着目し, 二階項言語の関数変数をただ一つに制限した単一化問題の決定可能性について議論する. まず, 与えられた二項のうち一方には関数変数の出現がなく, 他方には関数変数がただ一回出現する場合について, 単一化代入をすべて求め

[1]九州工業大学大学院 情報工学研究科 情報科学専攻

[2]九州工業大学 情報工学部 知能情報工学科

る手続きを示す. 次に, 与えられた二項にそれぞれちょうど一回ずつ変数が出現する場合について, 単一化問題の決定可能性を示す.

2 項言語

本論文では, 二階単一化に対して Goldfarb[3] と Farmar[2] の項言語を用いる.

項言語 (*term language*) とは 4 つ組 $L = (\mathrm{IC}_L, \mathrm{IV}_L, \mathrm{FC}_L, \mathrm{FV}_L)$ である. ここで,

1. IC_L は**固体定数** (*individual constants*) の集合であり, その要素を $a, b, c, \cdots$ と表す.
2. IV_L は**固体変数** (*individual variables*) の集合であり, その要素を $x, y, z, \cdots$ と表す.
3. FC_L は**関数定数** (*function constants*) の集合であり, その要素を $f, g, h, \cdots$ と表す.
4. FV_L は**関数変数** (*function variables*) の集合であり, その要素を $F, G, H, \cdots$ と表す.

FC_L, FV_L の各要素には 1 以上の引数が定められているとし, IC_L, IV_L, FC_L, FV_L は互いに素とする. また, $\mathrm{IC}_L \neq \phi$ とする. $\mathrm{FV}_L = \phi$ のとき L を**一階** (*first-order*) 項言語といい, $\mathrm{FV}_L \neq \phi$ のとき L を**二階** (*second-order*) 項言語という.

L 項を以下のように帰納的に定義する:

1. 各 $d \in \mathrm{IC}_L \cup \mathrm{IV}_L$ は L 項である.
2. $d \in \mathrm{FC}_L \cup \mathrm{FV}_L$ が引数 $n \geq 1$ をもち, $t_1, \cdots, t_n$ がそれぞれ L 項のとき, $d(t_1, \cdots, t_n)$ は L 項である.

$\mathrm{IC}_L \neq \phi$ より L 項の集合は空ではない.

$\mathrm{PM}_L = \{w_1, w_2, w_3, \cdots\}$ を L に含まれない記号の集合とし, その要素を**目印** (*place marker*) という. 項言語 L に PM_L を加えた言語を L^* といい, 以下のように帰納的に定義する:

1. 各 $d \in \mathrm{IC}_L \cup \mathrm{IV}_L \cup \mathrm{PM}_L$ は L^* 項である.
2. $d \in \mathrm{FC}_L \cup \mathrm{FV}_L$ が引数 $n \geq 1$ をもち, $t_1, \cdots, t_n$ がそれぞれ L^* 項のとき, $d(t_1, \cdots, t_n)$ は L^* 項である.

L^* 項を $r, s, t, \cdots$ と表す. L^* 項 t の**度数** (*rank*) を t に出現する目印 w_n のうち, 最大の n とし, これを $rank(t)$ と表す. このとき, L 項の度数は 0 となる. 直観的に $\mathrm{rank}(t)$ が n (≥ 1) となる L^* 項 t は n 引数関数を表す.

L^* 項 t に現れる変数の集合を $V(t)$ と表し, 以下のように帰納的に定義する:

1. $t \in \mathrm{IC}_L \cup \mathrm{PM}_L$ ならば $V(t) = \phi$ である.

2. $t \in \mathrm{IV}_L$ ならば $V(t) = \{t\}$ である.

3. $d \in \mathrm{FC}_L$ かつ $t = d(t_1, \cdots, t_n)$ ならば $V(t) = V(t_1) \cup \cdots \cup V(t_n)$ である.

4. $d \in \mathrm{FV}_L$ かつ $t = d(t_1, \cdots, t_n)$ ならば $V(t) = \{d\} \cup V(t_1) \cup \cdots \cup V(t_n)$ である.

また, t の部分項の集合 $\mathrm{sub}(t)$ を以下のように帰納的に定義する:

1. $t \in \mathrm{IC}_L \cup \mathrm{IV}_L \cup \mathrm{PM}_L$ ならば $\mathrm{sub}(t) = \{t\}$ である.

2. $d \in \mathrm{FC}_L \cup \mathrm{FV}_L$ かつ $t = d(t_1, \cdots, t_n)$ ならば
   $\mathrm{sub}(t) = \{t\} \cup \mathrm{sub}(t_1) \cup \cdots \cup \mathrm{sub}(t_n)$ である.

$\mathrm{rank}(t) \leq n$ の L^* 項 t と, L^* 項 $t_1, \cdots, t_n$ に対して, $t[t_1, \cdots, t_n]$ は次のように定義される:

1. $t \in \mathrm{IC}_L \cup \mathrm{IV}_L$ ならば $t[t_1, \cdots, t_n] = t$.

2. $t \in \mathrm{PM}_L$ かつ $t = w_i$ $(1 \leq i \leq n)$ ならば $t[t_1, \cdots, t_n] = t_i$ である.

3. $d \in \mathrm{FC}_L \cup \mathrm{FV}_L$ かつ $t = d(s_1, \cdots, s_m)$ ならば
   $t[t_1, \cdots, t_n] = d(s_1[t_1, \cdots, t_n], \cdots, s_m[t_1, \cdots, t_n])$ である.

直観的には, $t[t_1, \cdots, t_n]$ は t に現れる w_i を t_i に置き換えた結果となる.

L^* 項 t の最も外側の出現を t の**頭部** (*head*) といい, そこに出現する記号を $hd(t)$ と表す. また, t に変数が出現しないとき, t は**閉じている** (*closed*) という.

例 1 L^* 項 t, t_1, t_2, t_3, t_4 をそれぞれ $t = g(w_2, F(b, w_3, a)), t_1 = f(a, b), t_2 = g(c, a), t_3 = x$, $t_4 = g(f(a, b), c)$ とする. このとき, t_1, t_2, t_4 は閉じているが, t, t_3 は閉じていない. また,

$$\begin{aligned} t[t_1, t_2, t_3, t_4] &= g(w_2[t_1, t_2, t_3, t_4], F(b, w_3, a)[t_1, t_2, t_3, t_4]) \\ &= g(t_2, F(b[t_1, t_2, t_3, t_4], w_3[t_1, t_2, t_3, t_4], a[t_1, t_2, t_3, t_4])) \\ &= g(t_2, F(b, t_3, a)) \\ &= g(g(c, a), F(b, x, a)) \end{aligned}$$

となる.

L における**代入** (*substisution*) σ とは, 有限の定義域 $\mathrm{dom}(\sigma) \subseteq \mathrm{IV}_L \cup \mathrm{FV}_L$ を持ち, 個体変数を L 項に, n (≥ 1) 引数関数変数を度数 n 以下の L^* 項に変換する関数である. 代入 σ に対して, 変数 v を適用することを $\sigma(v)$ の代りに $v\sigma$ と表す. また, $x, F \in \mathrm{dom}(\sigma)$ のとき,

$x\sigma \neq x$ かつ $F\sigma \neq F(w_1,\cdots,w_n)$ と仮定する．$\mathrm{dom}(\sigma)$ が $\{v_1,\cdots,v_m\}$ かつ v_i を s_i に変換する代入 σ を $\{s_1/v_1,\cdots,s_m/v_m\}$ と表す．以降では，代入を $\theta,\sigma,\pi,\cdots$ と表す．L^* 項 t に代入 $\sigma=\{s_1/v_1,\cdots,s_m/v_m\}$ を施した結果を $t\sigma$ と表し，以下のように定義する:

1. $t\in \mathrm{IC}_L\cup\mathrm{IV}_L\cup\mathrm{PM}_L$ かつ $t\notin\{v_1,\cdots,v_m\}$ ならば，$t\sigma=t$.

2. $t\in\mathrm{IV}_L$ かつ，ある i に対して，$t=v_i$ ならば，$t\theta=v_i$.

3. $t=d(t_1,\cdots,t_n)$, $d\in\mathrm{FC}_L\cup\mathrm{FV}_L$ かつ $d\notin\{v_1,\cdots,v_m\}$ ならば，$t\sigma=d(t_1\sigma,\cdots,t_n\sigma)$.

4. $t=F(t_1,\cdots,t_n)$ かつ $F=v_i$ $(1\leq i\leq n)$ ならば，$t\sigma=s_i[t_1\sigma,\cdots,t_n\sigma]$.

任意の代入 σ に対して，t が L 項ならば，$t\sigma$ も L 項となる．代入 θ, σ の合成を $\theta\sigma$ と表し，任意の $v\in\mathrm{IV}_L\cup\mathrm{FV}_L$ に対して，$v(\theta\sigma)=(v\theta)\sigma$ と定義する．また，代入 θ を変数の集合 $V\subseteq\mathrm{IV}_L\cup\mathrm{FV}_L$ で制限した代入を $\theta|_V=\{s/v\mid v\in\mathrm{dom}(\theta)\cap V,\ s=\theta v\}$ と表す．特に，$\theta|_{\{v\}}$ を $\theta|_v$ と表す．

直観的に，L 項はラムダ計算におけるラムダ抽象を含まない項を表しており，L^* 項は，最も外側にだけラムダ抽象の現れる項を表している．$F\sigma=t$ となる代入 σ と，n 引数関数変数 F に対して，t はラムダ項 $\lambda w_1\cdots w_n.t$ を表す．また，L^* 項言語における代入は，ラムダ計算における β 簡約の役割も果たす．

3 一変数単一化

項言語 L における**単一化** (*unification*) とは，与えられた二項 t, s に対して，$t\theta=s\theta$ となる代入 θ を見つける操作であり，このような θ が存在するとき t と s は**単一化可能** (*unifiable*) という．このとき θ を t と s の**単一化代入** (*unifier*) という．さらに，与えられた二項に対して，単一化代入が存在するか否かを決定する問題を**単一化問題** (*unification problem*) という．

本節では，関数変数の個数を，ただ一つに制限した単一化問題の決定可能性について議論する．すなわち $\mathrm{FV}_L=\{F\}$ とする．まず，与えられた二項のうち一方は閉じており，他方は変数の出現がただ一つの場合について，単一化代入をすべて挙げる手続きを示す．次に，与えられた二項にそれぞれちょうど一度ずつ変数が出現する場合について，単一化問題の決定可能性を示す．最後に，与えられたに項のうち一方には変数が一回出現し，他方にはちょうど二回出現する場合について，単一化問題の決定可能性を示す．

単一化を二項の組の有限集合に拡張する．二項組の有限集合を**不一致集合** (*disagreement set*) という．与えられた不一致集合 D に対して，ある代入 θ が存在し，D の各要素 $(t,s)\in D$

```
function simple(t,s) /* 二項 t, s に対して, その不一致集合を返す */
  if s,t ∈ IC_L then
    if t = s then simple := φ else simple := Fail
  else if t ∈ IV_L or s ∈ IV_L then simple := {(t,s)}
  else /* ここで, t = d(t_1,···,t_n), s = d'(s_1,···,s_m) とする. */
    if d,d' ∈ FV_L then simple = {(t,s)}
    else if d = d' then /* このとき n = m */
      S := φ ;
      for i := 1 to n do
        S' := simple(t_i, s_i) ;
        if S' = Fail or S = Fail then S := Fail
        else S := S ∪ S'
      endfor;
      simple := S
    else/* d ≠ d' */
      if d,d' ∈ FC_L then simple := Fail
      else simple := {(t,s)}
```

図 1: 手続き *simple*

が $t\theta = s\theta$ となるとき, D は単一化可能であるといい, このときの代入 θ を D の単一化代入という.

与えられた二項 t, s に対して, 不一致集合を返す手続き *simple*を図 1に示す. 手続き *simple*には, 次の性質が成り立つ.

命題 1 与えられたに項 t, s の単一化代入の集合を $U(t,s)$ と表す. このとき, $simple(t,s) \neq$ Fail ならば,

$$\bigcap_{(p,q)\in simple(t,s)} U(p,q) = U(t,s)$$

となる. また, $simple(t,s) =$ Fail のとき, またそのときに限り $U(t,s) = \phi$ となる.

まず, 与えられた二項 t, s に対して, s は閉じていて, t における変数の出現がただ一つの場合について, 単一化代入をすべて挙げる手続きを示す.

F を n 引数関数変数とする. また, t を, F がちょうど一回現れる L 項とし, s を閉じた項とする. このとき, $simple(t,s) \neq$ Fail ならば, (t,s) の不一致集合は $\{(F(t_1,\cdots,t_n), s')\}$ の形をしている. したがって, t における変数の出現を頭部に制限しても一般性は失われない. そこで, 以降では t を頭部に変数がちょうど一度現れる L 項とする. このとき, L^* 項の集合 $M_t(s)$ を以下のように帰納的に定義する:

1. $s \in \mathrm{IC}_L$ ならば $M_t(s) = \{s\} \cup \{w_i \mid s = t_i\ (1 \leq i \leq n)\}$.

2. $s = f(s_1, \cdots, s_m)$ ならば,
$$M_t(s) = \{w_i \mid s = t_i\ (1 \leq i \leq n)\} \cup \{f(r_1, \cdots, r_m) \mid r_i \in M_t(s_i)\ (1 \leq i \leq m)\}.$$

$M_t(s)$ は直観的には, s の任意の部分項の出現を w_i $(1 \leq i \leq n)$ で置き換えた L^* 項 s' のうち $s'[t_1, \cdots, t_n] = s$ となる集合である. このとき, 以下の補題が成り立つ.

補題 1 $t = F(t_1, \cdots, t_n)$ を頭部に変数がちょうど一度だけ現れる L 項とし, s を閉じた項とする. $U(t, s)$ を t と s の単一化代入の集合とする. このとき,

$$\{\theta|_F \mid \theta \in U(t, s)\} = \{\{r/F\} \mid r \in M_t(s)\}$$

が成り立つ.

証明 s の構成に関する帰納法による. □

補題1より, $M_t(s)$ から t と s の単一化代入の集合を簡単に作ることができる.

次に, 与えられた二項にそれぞれちょうど一度ずつ変数が出現する場合について, 単一化問題の決定可能性を示す.

言語 L^* に特別な記号 □ を加え, L^* 項を以下のように拡張する:

1. 各 $d \in \mathrm{IC}_L \cup \mathrm{IV}_L \cup \mathrm{PM}_L \cup \{\square\}$ は L^* 項である.
2. $d \in \mathrm{FC}_L \cup \mathrm{FV}_L$ が引数 $n \geq 1$ をもち, $t_1, \cdots, t_n$ がそれぞれ L^* 項のとき, $d(t_1, \cdots, t_n)$ は L^* 項である.

このように拡張された言語 L^* 上の二項演算 $t \cdot s$ を以下のように定義する:

1. $t = \square$ ならば $t \cdot s = s$.
2. $t \in \mathrm{IC}_L \cup \mathrm{IV}_L \cup \mathrm{PM}_L$ ならば $t \cdot s = t$.
3. $d \in \mathrm{FC} \cup \mathrm{FV}$ かつ $t = d(t_1, \cdots, t_n)$ ならば $t \cdot s = d(t_1 \cdot s, \cdots, t_n \cdot s)$.

演算 $\cdot$ には結合則が成り立つ. また, t_i $(1 \leq i \leq n)$ を L^* 項とすると, $(t \cdot s)[t_1, \cdots, t_n] = (t[t_1, \cdots, t_n]) \cdot (s[t_1, \cdots, t_n])$ となる. 以降では, 特に混乱ない限り, 拡張された L^* 項を L^* 項という.

例 2 $t_1 = f(a, \square, b)$, $t_2 = g(c, \square)$, $t_3 = F(a, g(a, b), c)$ とする. このとき, 以下の等式が成り立つ.

$$\begin{aligned} t_1 \cdot t_2 \cdot t_3 &= f(a, \square, b) \cdot g(c, \square) \cdot F(a, g(a, b), c) \\ &= f(a, g(c, \square), b) \cdot F(a, g(a, b), c) \\ &= f(a, g(c, F(a, g(a, b), c)), b) \end{aligned}$$

与えられた二項を t, s とする. $simple(t,s) \neq \mathrm{Fail}$ のとき, $simple(t,s) = D$ とする. このとき, 明らかに関数変数 F を含む項の組 $(t,s) \in D$ は次のいずれかの形となる:

1. $(F(t_1,\cdots,t_n),\ F(s_1,\cdots,s_n))$,
2. ある閉じた項 s', t' に対して, $(F(t_1,\cdots,t_n),\ s'),\ (t', F(s_1,\cdots,s_n))$,
3. $(F(t_1,\cdots,t_n),\ s' \cdot F(s_1,\cdots,s_n))$,
4. $(t' \cdot F(t_1,\cdots,t_n),\ F(s_1,\cdots,s_n))$.

1 の場合は, 明らかに単一化可能となる.

2 の場合は, $t'' = F(t_1,\cdots,t_n)$, $s'' = F(s_1,\cdots,s_n)$ とすると, $M = M_{l''}(t') \cap M_{s''}(s')$ が空でなければ t と s は単一化可能であり, $r \in M$ に対して, $\{r/F\}$ が単一化代入となる.

3 の場合は t と s を入れ換える事により 4 の場合に帰着する. 4 の場合は $t_F = F(t_1,\cdots,t_n)$, $s_F = F(s_1,\cdots,s_n)$ とする. ここで, t' を以下のように表すことができる.

$$\begin{aligned}
t' &= r_1 \cdot \cdots \cdot r_k \cdot \cdots \cdot r_m, \\
r_1 &= f_0(r_{1,1},\cdots,r_{1,i_1-1},\square,r_{1,i_1+1},\cdots,r_{1,n_1}), \\
&\vdots \\
r_k &= f_{k-1}(r_{k,1},\cdots,r_{k,i_k-1},\square,r_{k,i_k+1},\cdots,r_{k,n_k}), \\
&\vdots \\
r_m &= f_{m-1}(r_{m,1},\cdots,r_{m,i_m-1},\square,r_{m,i_m+1},\cdots,r_{m,n_m}).
\end{aligned}$$

t と s 単一化代入は以下のように表される. $r'_{h,j} \in M_{t_F}(r_{h,j}) \cap M_{s_F}(r_{h,j})$ $(1 \leq h \leq m,\ 1 \leq j \leq i_{h-1},\ i_{h+1} \leq j \leq n_h)$,

$$\begin{aligned}
r'_1 &= f_0(r'_{1,1},\cdots,r'_{1,i_1-1},\square,r'_{1,i_1+1},\cdots,r'_{1,n_1}), \\
&\vdots \\
r'_k &= f_{k-1}(r'_{k,1},\cdots,r'_{k,i_k-1},\square,r'_{k,i_k+1},\cdots,r'_{k,n_k}), \\
&\vdots \\
r'_m &= f_{m-1}(r'_{m,1},\cdots,r'_{m,i_m-1},\square,r'_{m,i_m+1},\cdots,r'_{m,n_m}).
\end{aligned}$$

とする. このとき, $r'_1 \cdot \cdots \cdot r'_k \cdot \cdots \cdot r'_m[t_1,\cdots,t_n] = r'_1 \cdot \cdots \cdot r'_k \cdot \cdots \cdot r'_m[s_1,\cdots,s_n] = t'$ となる.

補題 2 t と s が単一化可能なとき, ある k, l $(1 \leq k \leq m,\ 1 \leq l \leq n)$ が存在して, $A = r'_1 \cdot \cdots \cdot r'_k$ かつ $B = r'_{k+1} \cdot \cdots \cdot r'_m$ のとき, t と s の単一化代入は $\{A \cdot (B \cdot A)^* w_l/F\}$ と表される.

証明 背理法による. □

補題 3 $A = r'_1 \cdot \cdots \cdot r'_k$ $(1 \leq k \leq m)$ とすると, $\{A \cdot w_l/F\}$ が t と s の単一化代入となる k, l $(1 \leq l \leq n)$ が存在しないとき, t と s は単一化不能となる.

以上の考察から次の定理が成り立つ.

定理 1 与えられた二項のそれぞれに関数変数が高々一回現れるとき, 単一化問題は決定可能となる.

4 まとめ

本論文では, 二階単一化のうち, 関数変数の個数を 1 に制限した場合, すなわち $\mathrm{FV}_L = \{F\}$ に制限した場合について, まず, 与えられた二項のうち一方にだけに関数変数が出現し, その出現回数も高々一回のとき, すべての単一化代入を求める手続きを示した. 次に, 各項に関数変数が高々一回ずつ現れる場合, 単一化問題は決定可能となることを示した.

今後の課題として, 本論文の結果を類推による定理証明系や, スキーマを用いたプログラム検証に応用することが考えられる.

参考文献

[1] Farmer, W. M.: *A unification algorithm for second-order monadic terms*, Annals of Pure and Applied Logic **39**, pp.131–174, 1988.

[2] Farmer, W. M.: *Simple second-order languages for which unification is undesidable*, Theoretical Computer Science **87**, pp.25–41, 1991.

[3] Goldfarb, W. D.: *The undecidability of the second-order unification problem*, Theoretical computer Science **13**, pp.225–230, 1981.

[4] Miller, D.: *A logic programming language with lambda-abstraction, function variables, and simple unification*, Lectur Notes in Artificial Intelligence **475**, pp.253–281, 1990.

[5] Huet, G. P.: *A unification algorithm for typed λ-calculus*, Theoretical Computer Science **1**, pp.27–57, 1975.

[6] Prehofer, C.: *Decidable higher-order unification problems*, CADE-**12** Proceedings, pp.635–649, 1994.

[7] Snyder, W. and Gallier, J.: *Higher-order unification revisited: Complete sets of transformations*, Journal of Symbolic Computation **8**, pp.101–140, 1989.