

重なりのある等式に対するパターン照合法の実現*

5F-2

近江 義智

栗原 正仁

大内 東†

北海道大学工学部‡

1 はじめに

項書き換えシステムを計算モデルとする等式プログラムは、記述性、読解性に優れている。このことは実際にプログラミングをおこなう立場から見ても注目すべき点であり、人間とプログラムとの間のセマンティクス・ギャップを低減するための有効な手段でもある。

等式プログラムは等式の集合であり、等式は書換え規則とみなす。また各々の等式に制約を課すことにより左正規項書換え系になる。したがって、等式プログラムは合流性を持ち、正規化戦略も定められている[1]。しかし、実際にプログラムを行う場合、制約条件をあまり顧慮しなくてもよいことが望ましい。

本稿では等式プログラムの処理系の一つとして著者が Common Lisp 上に実現した等式インタプリタについて概説し、その各等式の左辺における重なりは許されないという、制約についての問題点を示す。さらに、そのときに生じる問題点およびその解決に関して考察をおこない、その制約条件を緩めることへの可能性について調べる。

2 等式インタプリタ

等式インタプリタは等式プログラムから直接そのインタプリタを作成することによって実現されている。すなわち、等式インタプリタは等式プログラムから作られる項書き換えシステム (term rewriting system: TRS) とも言える。

2.1 等式プログラム

等式プログラムとは等式の集合であり、等式を左辺から右辺への書き換え規則として用いることによりプログラミング言語として扱う。

等式とは関数記号と変数記号から構成される2つの項 (term) P, Q を等号 (=) で結んだ式 $P = Q$ のことである。各関数記号はあらかじめ定められた項数を持っており、項数0の関数記号を定数とする。

また、等式 $P = Q$ とある項 M の部分項 M' において、変数を項で置き換える代入 θ により $M' \equiv P\theta$ となるとき、等式 $P = Q$ を書換え規則として項 M に適用できる。このとき M' を M のリデックスと呼び、項 M の部分項 M' を $Q\theta$ で置き換えた項 $M[Q\theta] \equiv N$ が得られる。項 N にリデックスがない場合、 N は正規形と呼ばれる。等式プログラムにおいてはこの正規形が計算結果である。

さらに等式の左辺に、以下の制約を課す。

2.2 制約条件

- (1) 等式の左辺は変数だけであってはならない。また、等式の右辺に現れる変数は、必ず左辺にも現れる。
- (2) 各々の等式の左辺では、同じ変数が繰り返し現れることなく (左線型性: *left-linear*)、また変数が関数記号や定数の左側に現れることもない (左正規性: *left-normal*)。
- (3) 異なる任意の2つの等式の左辺 P_i, P_j において、 P_i のいかなる部分項 (変数は除く) も P_j と単一化されない、つまり重なりがない。

すべての等式の左辺が上述の制約条件を満たすような等式プログラムは合流性を満たす[1]。

2.3 簡約戦略

等式プログラムに対する正規化戦略は最左最外戦略 (*leftmost-outermost-strategy*)、項の最も左側でかつ外側に位置するリデックスを書き換える戦略、である[1]。そこで、等式インタプリタにおける簡約戦略は最左最外戦略である。

3 制約条件に対する問題点

等式の左辺の重なりを修正する方法[1]はいくつかあるが、例えば等式の後ろに条件式を付加することが考えられる。すなわち、等式を

$$term_1 = term_2 \quad \text{where} \quad qualification$$

と表記することも許すことにする。しかし、この方法ですべての重なりが避けられるわけではない。

等式の左辺に重なりが生じる場合は多々ある。そのたびごとに特別なテクニックを用いたり、等式に工夫をほどこさなければならぬというのはかなりわずらわしい[2]。また、プログラムを等式のみで書き表すという利点や美しさを損なうことにもなる。このような点からも現在の等式プログラムに課せられた制約、特に重なり、を緩めるように改良を加える必要を感じる。

4 重なりを許す場合の問題点

等式の左辺に重なりを許すと、危険対が発生することになる。そのため合流性が失われる可能性が生じてくる。このとき一般に可簡約項が複数個あるため簡約戦略、つまり正規化戦略はどうなるのかということも問題になる。

*Implementation of pattern matching for overlapping equations
†Yoshitomo OUMI, Masahito KURIHARA and Azuma OHUCHI
‡Faculty of Engineering, Hokkaido University

4.1 合流性の判定

これまで様々な TRS についてその合流性が示されているが、左線型な TRS については Huet により次の定理が証明されている。

すべての書換え規則が左線型であるような TRS において、すべての危険対 s, t に $s \rightarrow^* t$ となるような関係があるとき、その TRS は合流性を満たす。ここで $\rightarrow^*$ は並列簡約 (parallel reduction) を表わしている。□

この定理により等式プログラムの合流性が判定が可能である。また、重なりのある等式を用いても合流性を持ったプログラムを書けることになる。

4.2 正規化戦略

重なりのある TRS に対する正規化戦略についても、研究が進められている [4]。左線型な TRS R が

- 強逐次性 (Strong sequentiality)

正規形でない各々の項は必ずインデックス (Index 正規形を得るために書き換えなければならないリデックス) をもっている。

- 平衡曖昧性 (balanced ambiguity)

すべての危険対 s, t が 0 回以上の同じステップ数で、ある同じ項 u に簡約される。

の二つの条件を満たすとき、 R は常にインデックスを書き換えるような戦略が正規化戦略となり、また UN 性 (Uniform Normal form property) をもつ。左線型な TRS が強逐次性を持つかどうかは決定可能であることも証明されている。

さらに左線型性、左正規性、平衡曖昧性を持つ TRS については正規化戦略は最左最外戦略であり、また UN 性を持つことも証明されている。

従って、平衡曖昧性を保つような重なりであれば、現在の等式インタプリタは項の照合法を改良するだけで等式の重なりを許せるようになる。

5 重なりのある項に対するパターン照合法

ここで扱うパターン照合問題を、項の有限集合 $P = \{p_1, \dots, p_n\}$ ともうひとつの項 q を与え、 q が $p_i \in P$ のインスタンスであるかどうかを決定する、として扱う。項は線型である。

この問題を解決するために今回採用したのはある種の照合オートマトンを作り出すという方法 [3] であり、構築法が従来のものとは少し異なる。これはまず、パターン語頭の (prefix) 間の重なりを決定するために語頭単一化 (prefix unification) という概念を用い、パターンの語頭の可能な組み合わせをすべて同時に単一化することにより、パターン集合の閉包 (closure) を計算し、この閉包から決定性照合オートマトンの一種を演繹するものである。

5.1 語頭単一化

以下で、 Σ は有限の空でないシンボルの集合、 Σ_n はランク n のシンボルの集合で $\Sigma = \bigcup_{n \in \mathbb{N}_0} \Sigma_n$ である。 Σ^* は Σ 上の全てのシンボル列の集合、 ε は空のシンボル、 $\text{Pref}(\nu)$ はシンボル列 $\nu (\in \Sigma^*)$ の語頭の集合 (ε および ν 自身も含む)、 $\omega \in \Sigma_0$ を変数シンボル (引数なし)、とする。このとき、 $\nu, \mu \in \Sigma^*$ の語頭単一化は以下のようにして得られる。

- $\nu \wedge \mu = \alpha(\nu' \wedge \mu')$ if $\nu = \alpha\nu', \mu = \alpha\mu', \alpha \in \Sigma$.
- $\nu \wedge \mu = \omega(\nu' \wedge \mu')$ if ν and μ have different head symbols and $\nu = \omega\nu', \mu = \omega\mu'$.
- $\nu \wedge \mu = \varepsilon$ in all other cases.

この定義により任意のパターン集合を同じ語頭を持つパターンの部分集合の和へと変換できる。

5.2 閉包の計算

同じ語頭 $\lambda \in \Sigma^*$ に対する項の語尾 (suffix) の集合を $M \subseteq \Sigma^*$ 、語尾集合 M と $\alpha \in \Sigma$ に対し、先頭のシンボル α を各項から除いて得られた語尾集合を $M/\alpha = \{\mu \in \Sigma^* | \alpha\mu \in M\}$ とする。このとき M に対する閉包 $\overline{M}$ は次のようにして得られる。

- $M \subseteq \{\varepsilon\} \Rightarrow \overline{M} = M$.
- $M \not\subseteq \{\varepsilon\} \Rightarrow \overline{M} = \bigcup_{\alpha \in \Sigma} \alpha \overline{M_\alpha}$ where

$$M_\omega = M/\omega$$

$$M_\alpha = \begin{cases} M/\alpha \cup \omega M/\omega & M/\alpha \neq \phi \\ \phi & \text{otherwise} \end{cases}$$

$\forall \alpha \neq \omega$

この定義を用いることで任意のパターン集合からその閉包が求められる。

例として、パターン集合 $P = \{faw, fwb, fgwa\}$ を考え、その閉包 $\overline{P}$ を求める。

$$\begin{aligned} \overline{P} &= f \overline{P/f} = f \overline{\{aw, wb, gwa\}} = f \overline{M}, \quad \text{但し,} \\ \overline{M} &= a \overline{M_a} \cup \omega \overline{M_\omega} \cup g \overline{M_g} \\ &= a \overline{\{w, b\}} \cup \omega \overline{\{b\}} \cup g \overline{\{wb, wa\}} \\ &= \{aw, ab, wb, gwb, gwa\}. \end{aligned}$$

このようにして得られた閉包から照合オートマトンを構築すると、重なりのある項に対するパターン照合が可能になる。

6 おわりに

等式インタプリタにおいて等式の左辺の重なりが禁止されているということは等式でプログラムを行なっていく上で支障になることがたびたびある。また、本稿で紹介してきたように重なりのある TRS に関する合流性や正規化戦略についてもかなりの研究がなされてきている。このような現状を考えると等式インタプリタに重なりを許すよう改良を加えるという研究は意義のあることだと思われる。

今後はさらに諸問題について検討を推し進め、重なりのある項に対するパターン照合法 [3] を等式インタプリタに取り入れていく予定である。

参考文献

- [1] M.J.O'Donnell, "Equational Logic as a Program Language", The MIT Press, 1985.
- [2] M.J.O'Donnell, "Term-Rewriting Implementation of Equational Logic Programming", *Proceeding on Rewriting Techniques and Applications*, LNCS 256, pp1-12, 1987.
- [3] Gräf, Albert, "Left-to-Right Tree Pattern Matching", *Proceeding of the 4th Inf. Conf. on Rewriting Techniques and Applications*, Springer LNCS 488, pp323-334.
- [4] Yoshihito Toyama, "Strong Sequentiality of Left-Linear Overlapping Term Rewriting Systems", NTT Laboratories.