数理モデル化と問題解決　25－2
（１９９９．７．１６）

文字列領域の問題解決における一階論理表現からのプログラム生成

吉田 忠行† 赤間 清†† 宮本 衛市†

† 北海道大学 大学院 システム情報工学専攻
札幌市北区北 13 条西 8 丁目. Tel. 011-706-6814
†† 北海道大学 情報メディア教育研究総合センター
札幌市北区北 11 条西 5 丁目. Tel. 011-706-6814

問題解決において，正しく，かつ効率的に問題を解くプログラムを作成するのは重要なことであるが，同時に困難なことでもある．本論文では，正しく効率的なプログラムを自動的に作成するための新しい方法を提案する．本論文では，文字列領域における問題の１つとして言語認識問題を取りあげる．一階述語論理表現を用いて定式化した問題仕様から，問題を高速に解くためのプログラムを生成する．ただし，本方法によって生成されるプログラムは，有限オートマトンに相当する単純なプログラムに限定している．

本論文では，問題仕様とプログラムを従来の理論よりも一般的な形で関係づけている，等価変換に基づく計算モデル（等価変換モデル）を採用する．等価変換モデルでは，プログラムは等価変換ルールの集合である．本論文のプログラム生成では，問題仕様から等価変換ルールを次々に生成し，その中から効率的なルールだけを選別する方法を用いる．ルール生成は「メタ計算」と呼ばれる計算によって行い，効率的なルールの評価基準に基づき，生成されたルールを選別する．

等価変換　問題解決　一階論理制約　宣言的記述　ルール生成　文字列領域

Program Generation from First-order Logical Expressions for Problem Solving on a String Domain

Tadayuki Yoshida† Kiyoshi Akama†† Eiichi Miyamoto†

† Division of System and Information Engineering, Hokkaido University
North 13 West 8 Kita-ku Sapporo. Tel. 011-706-6814
†† Center of Infomation and Multimedia Studies, Hokkaido University
North 11 West 5 Kita-ku Sapporo. Tel. 011-706-6814

It is very important but not easy to write correct and efficient programs for solving problems. In this paper, we propose a new method for automatic generation of correct and efficient programs. We assume that we are given a language recognition problem that are formalized by using first-order logical expressions. Correct and efficient programs are then generated from the problem specification. The class of programs generated by the proposed method correspond to the one of finite automatons.

In this paper we adopt a computation model based on equivalent transformation (ET model), which formalizes the relation between specifications and programs more generally than other existing models. In the ET model, a program consists of equivalent transformation rules (ET rules). We generate many ET rules from given specification represented by a first-order logical expression and select "efficient" rules among them. We generate ET rules by "meta computation" and select efficient rules based on an evaluation function.

equivalent transformation, problem solving, first-order logical constriant,
declarative description, rule generation, string domain

1 まえがき

問題解決において，正しく，かつ効率的なプログラムを作成するのは重要なことである．しかし，そのようなプログラムを作成するのは，多くの場合，容易ではない．本論文では，正しく，かつ効率的なプログラムを作成するために，プログラムを理論に基づいて自動生成することを試みる．具体的には，文字列領域における問題のうち言語認識問題を取りあげ，一階述語論理表現を用いて定式化し問題仕様とする．その問題仕様から，問題を効率的に解くためのプログラムを生成する．このプログラムは，有限オートマトンに相当する単純なものである．

本論文では，問題仕様とプログラムを明確に，かつ強力に結びつけるために，等価変換に基づく計算モデル（等価変換モデル）を採用する．等価変換モデルでは，プログラムは等価変換ルールの集合である．したがって，ルールの善し悪しがプログラムの善し悪しを決定する．本論文では，一階述語論理表現による問題仕様から等価変換ルールを生成し，その中から本論文における「効率的なルール」を選別する方法を用いる．問題仕様から等価変換ルールを生成するために,「メタ計算」と呼ばれる計算を行う．また，効率的なルールを明確に判定する評価基準を設定する．プログラム生成システムは，次々に生成されたルールの中からその評価基準を満たすルールの集合を出力する．

一階述語論理表現からプログラムを生成する枠組としては，論理プログラムに基づく方法も存在する [5]．しかし，論理パラダイムにおけるプログラムは，確定節集合などの論理式であり，アルゴリズムの表現力に乏しく，効率的なプログラムを自然に記述することに限界がある．それに対して等価変換パラダイムでは，手続きは表現力の豊かな等価変換ルールで記述され，広範なアルゴリズムを自然に表現することができる．

また，特に一階述語論理式を扱うときには，論理パラダイムでは，計算方法が，一階述語論理式を，標準形に変換するフェーズとその変換された標準形を SLDNF 導出に基づいて推論するという 2 つのフェーズに分離している [6] など，計算の枠組が統一的ではない．いっぽう，等価変換の枠組では,「計算=等価変換」という統一した原理がある．このため，計算に関してすべて単一の原理で行うことができ，プログラム生成の研究も見通しよく行えることが期待できる．

等価変換モデルではすでに，確定節から等価変換ルールを生成する理論的な基礎が与えられている [2]．本論文では，その理論を基にして，文字列領域をターゲットとして，表現レベルを確定節から一階述語論理表現を用いることができるところまで拡大する．これは，等価変換の理論においてこれまで並行して行われていた，一階述語論理表現の扱いに関する研究と，純粋な確定節レベルの仕様から等価変換ルールを生成する研究を結びつけるものである．

2 等価変換による問題解決

2.1 言語認識プログラム作成問題

プログラム作成問題の例として，次の問題を考える．

> Σ をアルファベット $\{a,b,c\}$ とする．Σ 上の言語 L は，次の条件を満たす Σ 上の任意の文字列 δ の集合と定義する．
>
> > δ が もし aa という部分列を含むならば，その aa の右には 必ず ba が出現する．
>
> Σ 上の任意の文字列が与えられたとき，それが 言語 L に属するかどうかを判定するプログラムを作成せよ．

本論文では，この問題を例として，文字列領域でのプログラム生成の方法を議論する．

2.2 宣言的記述による問題の定式化

$s(x)$ は 「x は 言語 L の正しい文字列である」を意味するアトムとする．

任意の Σ 上の文字列 δ が与えられたとき，$s(\delta)$ が成り立つか否かを判定する問題は，

$P = \{\, s(X) \leftarrow$
$\quad \forall\alpha\forall\beta\,([X = \alpha aa\beta] \Rightarrow \exists\gamma[\beta = ba\gamma]).\}$
$P' = P \cup \{yes \leftarrow s(\delta)\}$

として，

$yes \in \mathcal{M}(P')$??

か否かを判定する問題として定式化できる．ただし，$[X = \alpha aa\beta]$ や $[\beta = ba\gamma]$ は 文字列領域上の等式制約，yes は述語である．

3 本論文の問題設定

3.1 対象とする問題クラス

本論文では，プログラム生成の対象クラスを任意のアルファベット Σ と，一階論理制約 $FLC(X)$ のペア全体の集合とする．また，$\forall X\, s(X) \Leftrightarrow FLC(X)$ によって定義された述語を s とする．Σ 上の文字列 δ が任意に与えられたとき，$s(\delta)$ が成立するか否かを判定する効率的なプログラムを生成するのが，本論文の問題である．

プログラムに効率性や停止性を保証するために，生成されるプログラムは，以下のような条件を満たす等価変換ルールから構成されるものに限定する．

条件 1 $atom, atom'$ をメタアトム（??節参照）として，ルールの形は，

(1) $atom \rightarrow \langle true \rangle$.
(2) $atom \rightarrow \langle false \rangle$.
(3) $atom \rightarrow atom'$.

のどれかになっていなければならない．

条件 2 ルールの適用前後において，メタアトムのサイズ[1] は減少していなければならない．

条件 3 生成されるルールの左辺のメタアトムの引数パターンは，全体で Σ 上の文字列をすべてカバーしていなければならない．

条件 4 生成されるルールの左辺のメタアトムの引数パターンは，互いに素になっていなければならない．こうすることによって，同一のアトムに対して，複数のルールが選択されることはなくなる．したがって，制御に依存しないプログラムとなる．

3.2 例題において生成されるプログラム

前章で取り上げた例題は，

$\Sigma = \{a, b, c\}$

$FLC(X) = \forall\alpha\forall\beta\,([X = \alpha aa\beta] \Rightarrow \exists\gamma[\beta = ba\gamma])$

とした問題である．

このようにして与えた問題仕様から，本論文の方法では次のようなプログラム（等価変換ルールの集合）が生成される．$\&X$ は任意の文字列を表す変数である．このプログラムは，全体として言語 L を認識するオートマトンに相当するアルゴリズムを実現している．

r1: $s(\epsilon) \to \langle true \rangle$
r2: $s(b\&X) \to s(\&X)$
r3: $s(c\&X) \to s(\&X)$
r4: $s(a) \to \langle true \rangle$
r5: $s(ab\&X) \to s(\&X)$
r6: $s(ac\&X) \to s(\&X)$
r7: $s(aa) \to \langle false \rangle$
r8: $s(aab) \to \langle false \rangle$
r9: $s(aac\&X) \to \langle false \rangle$
r10: $s(aaa\&X) \to \langle false \rangle$
r11: $s(aabb\&X) \to \langle false \rangle$
r12: $s(aabc\&X) \to \langle false \rangle$
r13: $s(aaba\&X) \to s(a\&X)$

ルールの解釈は，次のとおりである．

例えば，ルール r4 は，$s(a)$ というアトムがボディにあれば，それを除去していよいことを表している．ルール r7 は，$s(aa)$ というアトムがボディにあれば，そのアトムを含む節を除去してよいことを表している．ルール r13 は，$s(aaba\&X)$ というアトムがあれば，それを $s(a\&X)$ というアトムに書き換えてよいことを表している．それ以外のルールについても同様である．

これらのルールは，前節で定めた条件（条件 1〜条件 4）をすべて満たしている．

4 等価変換によるプログラム生成の方法

4.1 プログラム生成の要素

本論文におけるプログラム生成の枠組は，次のような要素から成り立っている．

(1) メタルールの準備
メタ節を変換するためのメタルールを用意する．
(2) ルール生成ルーチン *generate*
メタルールを用いて変換を行い，ルールを出力する．
(3) 評価基準 *eval*
ルールの評価値を与える写像である．
(4) メタルール生成ルーチン *genMR*
ルール生成時に，ルールからメタルールを作る．
(5) アトムパターンの集合の更新ルーチン *specialize*
ルール生成ルーチンに与えるアトムパターンの集合を更新する．

以降の節では，これらの各要素について説明し，最後にプログラム生成アルゴリズムを提案する．

4.2 プログラム生成システム

これまで定義した，*generate*, *eval*, *genMR*, *specialize* を用いて，本稿で対象としている問題クラスに対する，プログラム生成の手続きを提案する．

ルールの集合，メタルールの集合，アトムパターンの集合をそれぞれ，GR，MR，Pat とする．

まず，GR, MR, Pat をそれぞれ初期化する．そのあと，ターゲットとするアトムパターンがなくなるまで生成の処理を続ける（while ループ）．生成処理は，現在のパターン Pat を Pat' に保存したあと，Pat' の各要素（アトムパターン）p に対して，メタルール MR を使ってルールを 1 つ生成する．これを r とする．生成処理がうまくいかなかったとき（r =null）は次のパターンにうつる．生成処理がうまくいったときは，そのルール r を評価して，基準を満たしていれば（$eval(r) = T$）r を GR に追加し，生成に用いたアトムパターン p を Pat から除去する．生成したルールが評価基準を満たさないとき（$eval(r) = F$）は，そのルールから作られたすべてのメタルール（$genMR(r)$）を，メタルール集合 MR に追加し，以降のルール生成処理に用いることができるようにする．Pat' のすべての要素を処理したら（$Pat' = \emptyset$），その時点でのアトムパターンの集合 Pat を更新し（*specialize*），while ループを繰り返す．

生成するパターンがなくなったとき（$Pat = \emptyset$）は，すべてのパターンが生成できたので，その時点で得られているルール集合 GR を出力して終了する．

[1] メタアトムにおける変数と定数の総出現数．

5 ルールの正当性と停止性および高速性

5.1 ルールの正当性

本稿で述べた方法で生成されたルールの正当性は，文献 [2] と類似の理論で保証することができる．

5.2 ルールの停止性

生成されたルールの停止性は，次の 2 つの事実から容易に導かれる．

- 条件 4 により，常にちょうど 1 つのルールだけが適用可能になる．しかも条件 3 により，基礎 s アトム全体の被覆になっているので，計算の途中でルールが適用できなくなるようなことはない．
- 条件 2 により，ルールを適用する際のアトムのサイズは減少列をなす．さらに，この減少列の下限は 0 である．

したがって，基礎 s アトム は，ルールを複数回適用することにより，必ず $\langle true \rangle$ または $\langle false \rangle$ に置き換えられるので，L に関する言語認識問題は すべて本論文で生成したルールだけで 有限ステップで解くことができる．

5.3 ルールの効率性

本稿で生成したルールを用いて，??節での文字列を変換すると，次のようになる．

p1: $yes \leftarrow s(caaba)$
p2: $yes \leftarrow s(aaba)$
p3: $yes \leftarrow s(a)$
p4: $yes \leftarrow$

この変換過程で用いた等価変換ルールを以下に示す．

p1 → p2　r3: $s(c\&X) \rightarrow s(\&X)$
p2 → p3　r13: $s(aaba\&X) \rightarrow s(a\&X)$
p3 → p4　r4: $s(a) \rightarrow \langle true \rangle$

このように，プログラム生成を行うことによって，大幅に変換ステップ数を減らすことが可能になるだけでなく，ひとつひとつの変換ステップにかかる計算コストも非常に少なくなっている．したがって，計算効率が大きく改善されていることがわかる．

6 おわりに

本研究は，等価変換の理論においてこれまで行われていた，一階述語論理表現の扱いに関する研究と，純粋な確定節レベルの仕様から等価変換ルールを生成するという研究を結びつけて得られた．

本論文では，生成するプログラムはオートマトンに相当するレベルに限定して議論したが，この方法は，それよりもっと広範な表現力をもつプログラムを生成する目的にも拡張することができる．

参考文献

[1] 赤間清, 繁田良則, 宮本衛市：論理プログラムの等価変換による問題解決の枠組，人工知能学会誌，Vol.12, No.2, pp.90–99 (1997).

[2] 赤間清，小池英勝，宮本衛市：等価変換ルールの生成方法の理論的基礎，情報処理学会，SIG-PRG (1999).

[3] 吉田忠行, 赤間清, 宮本衛市：一階論理制約の等価変換を用いた問題解決の枠組，電子情報通信学会研究報告，KBSE98-6, pp.17–24 (1998).

[4] Futamura,Y.: Partial evaluation of computation process - an approach to a compiler-compiler, *Systems, Computers, Controls*, vol.25 pp.45-50 (1971)

[5] Lloyd, J.W.: *Foundations of Logic Programming*, Second edition, Springer-Verlag (1987).

[6] Lloyd,J.W. and R.W.Topor: Making Prolog More Expressive, J.Logic Programming 1,3, pp225–240 (1984).

[7] Takeuchi,A. and Fujita,H. : Competitive Partial Evaluation, *Workshop on Partial Evaluation and Mixed Computation*, pp.317-326, October (1987)

[8] 玉木 久夫：論理型言語におけるプログラム変換，プログラム変換，共立出版 (1987)

[9] Tamaki,H. and Sato,T : Unfold/fold Transformation of Logic Programs, Proc. of 2nd ILPC, pp.127-138 (1984)