数理モデル化と問題解決　26－13
（1999．9．22）

問題解決のための効率的な等価変換ルールの探索

小池英勝[†]　赤間清[††]　宮本衛市[†]

等価変換による問題解決では，等価変換ルールの集合で問題解決の手続きを記述する．等価変換ルールを用いると，正当で効率的な問題解決が可能である．論理プログラミングでは，関係を正しく記述しても解を得られない場合がある (reverse の例題など) が，そのような問題に対してもこの方法では正しく解を与えることができる．これまでに，確定節の集合から多数の正当な等価変換ルールの集合を生成するための基礎理論が提案されている[2]．本論文では，等価変換ルールの集合から探索によって，問題を効率的に解決するための等価変換ルールを発見する方法を提案する．本方法は，正当で効率的なプログラムを自動生成するための基礎を与える．

Searching for Efficient Equivalent Transformation Rules for Problem Solving

HIDEKATSU KOIKE,[†] KIYOSHI AKAMA[††] and EIICHI MIYAMOTO[†]

In problem solving based on equivalent transformation(ET), a procedure is represented by a set of ET rules, which enables us to solve problems correctly and efficiently. There exist some problems that can be solved correctly by the ET paradigm, but cannot be solved by natural logic programs, though they define problems correctly. A theoretical foundation for generating a large class of correct ET rules has been proposed[2]. In this paper we develop a method of searching the class of rules for efficient ET rules. This method lays a foundation of synthesizing correct and efficient programs.

1. はじめに

人にとってわかり易い問題記述から，その問題を解くための効率的なプログラムを自動的に合成することができれば，問題解決における人の負担を大きく減らすことができる．本研究では，等価変換による問題解決の枠組[4]を基礎として，プログラム合成を行う．この合成法は，問題を宣言的に記述した確定節の集合を入力として，そこから等価変換ルールの集合を出力する．等価変換による問題解決では，等価変換ルールの集合がプログラムである．等価変換ルールは，問題記述の宣言的意味[4]を保存したまま変換するルールである．

正当な等価変換ルールを多数生成する方法[1]とその理論的基礎[2]が既に提案されている．多数の正当な等価変換ルールの集合から効率的なものだけを選びプログラムを構成すれば正当で効率的なプログラムを合成できる．よって等価変換の枠組みで行うプログラム合成は，多数の正当な等価変換ルールの生成と，そのルールの集合から効率的なルールを探索することが中心になる．本プログラム合成方法は，問題を正しく解くプログラムの合成と，合成されるプログラムの効率化という2つの側面を持っている．現在このプログラム合成を自動的に行うシステムが試作されている[3]．本論文では等価変換ルールの生成と探索を中心に，プログラム自動合成システムの実現について議論する．

2. 本プログラム合成の特徴

2.1 プログラムの部品としての等価変換ルール

与えられた仕様を満たすプログラムは一般に多数存在する．プログラム自動合成を行う際，その中から効率的なものを出力したい．しかし，プログラムを1つずつ合成して評価するのでは計算量が大きくなりすぎる．そこで，1つのプログラムを合成する問題を，そのプログラムを構成する多数の部品を生成する問題に置き換える．そして多数のプログラムを評価する代わ

† 北海道大学大学院 工学研究科 システム情報工学専攻
Division of System and Information Engineering, Hokkaido University
†† 北海道大学 情報メディア教育研究総合センター
Center of Information and Multimedia Studies, Hokkaido University

りに，部品の候補を1つずつ評価選択して全体を作る．このことによって複雑なプログラム合成の問題は，比較的小規模で簡単な問題に分解されたことになる．

等価変換ルールは，追加による全体への正当性の影響がない．しかも個々のルールが効率的であれば全体としても効率的になる[5]．このことから，本研究では「プログラムの部品＝等価変換ルール」と捉えることにより，プログラム合成を行う．

本プログラム合成は以下のような特徴を持つ．

(1) プログラムの部品を順次生成して追加することによってプログラム合成を行う
(2) 高いアルゴリズム記述能力を持つプログラムのクラスを採用している
(3) 多数の候補の中からプログラムを探索する

3. プログラム合成の概要

3.1 入力と出力

本プログラム合成では問題記述を入力とし，等価変換ルールの集合を出力する．問題記述は，関係を表す確定節の集合と質問を表す節からなる．問題記述の具体例を以下に示す．

関係を表す確定節集合

$reverse([X|Y], Z)$
$\leftarrow reverse(Y, R), append(R, [X], Z).$
$reverse([\,], [\,]) \leftarrow .$
$append([\,], X, X) \leftarrow .$
$append([A|X], Y, [A|Z]) \leftarrow append(X, Y, Z).$

質問

$ans(X) \leftarrow reverse(X, [1, 2]).$

問題記述 $(p1)$

この問題記述を与えると，プログラム合成システムは以下のような等価変換ルールの集合を生成する．

$(r1)$ $reverse(\&X,\ [\&A|\&Y])$
$\rightarrow \{\&X = [\#B|\#W]\},$
$reverse(\#W, \#Z),$
$append(\#Z, [\#B], [\&A|\&Y]).$

$(r2)$ $reverse(\&X,\ [\,]) \rightarrow \{\&X = [\,]\}.$

$(r3)$ $append(\&A, [\&B], [\&C, \&D|\&E])$
$\rightarrow \{\&A = [\&C|\#F]\},$
$append(\#F, [\&B], [\&D|\&E]).$

$(r4)$ $append(\&A, [\&B|\&C], [\&D])$
$\rightarrow \{\&A = [\,], \&C = [\,], \&B = \&D\}.$

これらのルールは、入力として与えられた質問とそれに類似した質問に解を与えることができる．また，単に問題を解けるだけではなく効率的に問題が解けるルールが出力されている (4節，6節)．

3.2 プログラム合成の手順と要素

プログラム合成の大まかな手順の流れ図が $Fig.1$ である．

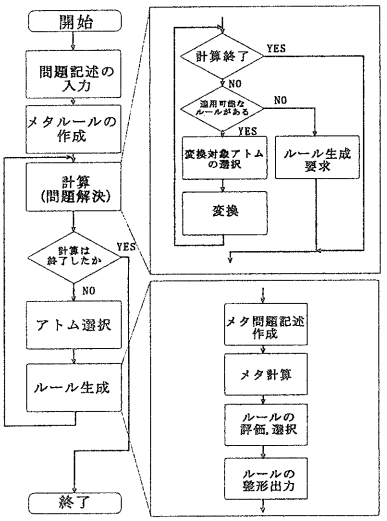

Fig.1 プログラム合成の手順

本プログラム合成システムは，大きく分けて問題解決とルール生成の2つのサブシステムから成る．以下に，システムの入出力とサブシステムの構成要素を列挙する．

プログラム合成システムの入出力

- **問題記述** 関係を宣言的に記述した確定節の集合と，質問を表す節の集合から成る．
- **プログラム** 本システムの出力で，等価変換ルールの集合．

問題解決

- **変換** 等価変換ルールで問題記述を変換☆することにより，質問に対する答えを計算する．
- **ルール生成要求** 必要な等価変換ルールをアトムを使って要求する．

ルール生成

- **メタ問題記述** 多数の問題記述を1つの表現で表したもの．
- **メタルール** メタ問題記述を等価変換するためのルール．
- **メタ計算** メタ問題記述をメタルールで変換する．
- **探索** 生成可能なルールの集合から効率的なルールを探索する．
- **ルールの評価基準** 探索で個々のルールを評価

☆ 実際に変換されるのは質問の節だけである．

するときに使う.

4. 等価変換ルールと問題解決

等価変換パラダイムでは，等価変換ルールで問題記述の質問を表す節を変換していくことによって解を求める.

たとえば，3.1節の問題記述の質問の節は，3.1節の等価変換ルール $(r1)$ ～ $(r4)$ を用いて以下のように変換される.

1) $ans(X) \leftarrow \mathbf{reverse(X, [1,2])}$.

2) $ans([B1|W1]) \leftarrow reverse(W1, Z1),$
$\mathbf{append(Z1, [B1], [1,2])}$.
$(r1)$ より

3) $ans([B1|W1]) \leftarrow reverse(W1, [1|F1]),$
$\mathbf{append(F1, [B1], [2])}$.
$(r3)$ より

4) $ans([2|W1]) \leftarrow \mathbf{reverse(W1, [1])}$.
$(r4)$ より

5) $ans([2, B2|W2]) \leftarrow reverse(W2, Z2),$
$\mathbf{append(Z2, [B2], [1])}$.
$(r1)$ より

6) $ans([2,1|W2]) \leftarrow \mathbf{reverse(W2, [])}$.
$(r4)$ より

7) $ans([2,1]) \leftarrow$.
$(r2)$ より

ここでは，1) から 7) の 7 ステップの変換で解が得られている．太字で強調されたアトムがルール適用によって書き換えられている．変換の過程について説明する．等価変換ルール $r1$ によって 1) から 2) に変換される．以下同様に 2) から 7) まで変換される．7) の節で解が $[2,1]$ であることを表している.

5. ルール生成の基礎

5.1 メタ問題記述

メタ問題記述とは，問題記述の集合を表したものである．メタ問題記述はメタ節の集合である．メタ節は，ダミーヘッドとメタアトムからなる．メタ節，メタアトムはそれぞれ，共通の性質を持った節，アトムの集合を表す．メタ節の具体例は以下のようなものである.

(mc) $h \leftarrow reverse(\&A, [1,2])$.

このメタ節は，第 1 引数が任意の項で，第 2 引数が $[1,2]$ であるような $reverse$ アトムがボディに現れる全ての節を代表している．ここで，ダミーヘッド h は任意のヘッドを表す.

5.2 メタルール

メタルールは，与えられた問題記述から機械的な書き換えで作ることが出来る．3.1節の問題記述 $(p1)$ が与えられたとする．この問題記述から得られるメタルールは以下のようなものである.

$(m1)$ $reverse(*X, *Z)$
$\rightarrow equal(*X, [\%Xs|\%Y]),$
$reverse(\%Y, \%R),$
$append(\%R, [\%Xs], *Z);$
$\rightarrow equal(*X, []), equal(*Z, [])$.

$(m2)$ $append(*X, *Y, *Z)$
$\rightarrow equal(*X, []), equal(*Y, *Z);$
$\rightarrow equal(*X, [\%A|\%Xs]),$
$equal(*Z, [\%A|\%Zs]),$
$append(\%Xs, *Y, \%Zs)$.

5.3 メタ問題記述の変換 (メタ計算) とルール生成

具体例で説明する．ここで示す例は，3.1節の等価変換ルール $(r2)$ の生成の例である．次のような 1 つのメタ節からなるメタ問題記述が与えられたとする.

$h \leftarrow reverse(\&A, [])$.

そして，メタルールで以下のように変換していく.

1) $h \leftarrow \mathbf{reverse}(\&\mathbf{A}, [])$.

2) $h \leftarrow equal(\&A, [\#X|\#X1]),$
$reverse(\#X1, \#R),$
$\mathbf{append}(\#\mathbf{R}, [\#\mathbf{X}], [])$.
$h \leftarrow equal(\&A, []), equal([], [])$.
$(m1)$ より

3) $h \leftarrow equal(\&A, [\#X|\#X1]),$
$reverse(\#X1, \#R),$
$equal(\#R, []),$
$\mathbf{equal}([\#\mathbf{X}], [])$.
$h \leftarrow equal(\&A, [\#X|\#X1]),$
$reverse(\#X1, \#R),$
$equal(\#R, [\#A, \#X2]),$
$\mathbf{equal}([], [\#\mathbf{A}, \#\mathbf{Zs}]),$
$append(\#X2, [\#X], \#Z)$.
$h \leftarrow equal(\&A, []), \mathbf{equal}([], [])$.
$(m2)$ より

4) $h \leftarrow equal(\&A, [])$.
(等式制約に関するメタルール) より

太字で強調されたメタアトムがメタルールによって書き換えられている．3) から，4) への変換には等式制約に関するメタルールが適用される．このメタルールはあらかじめシステムに用意しておく．3) の 1 番目と 2 番目のメタ節の equal は矛盾しているので節そのものが消える.

ここで示した変換の列 1) ～ 4) は，多数の中の一例に過ぎない．例えば，2) には，再び $(m1)$ を適用する

ことが可能である．このようにメタ節 (メタ問題記述) の変換は一般に非決定的であるので，多数の変換列が得られる．

5.3節の 1) から 4) への変換の列はどの段階も等価変換により得られたものである．よって，1) から 4) への直接の書き換えも等価変換であるので，1) と 4) から新しい等価変換ルールを作ることが出来る．このようにして ($r2$) が得られる．変換列は多数生成されるので，多数の等価変換ルールが生成可能である．この等価変換ルールの集合ひとつひとつがプログラムの部品の候補である．

6. 等価変換ルールの生成例と効率

本プログラム合成法で可能な効率化は

(1) unfold/fold 変換による効率化

(2) 新述語の定義による効率化

(3) 再帰呼び出しの除去

などがある．

以下に，新述語の定義による再帰呼び出し除去の例を示す．問題記述として ($p1$) を与える．ただし，質問の節の $reverse$ の第 1 引数と第 2 引数を逆にしたものを与える．本システムは自動的に以下のような等価変換ルールの集合を生成する．

($r10$) $reverse([\&A|\&X], \&Y) \rightarrow auto(\&X, [\&A], \&Y)$.

($r11$) $auto([\&A|\&X], \&Y, \&Z) \rightarrow auto(\&X, [\&A|\&Y], \&Z)$.

($r12$) $auto([], \&Y, \&Z) \rightarrow \{\&Y = \&Z\}$.

このルールの集合は，スタッキングの時間，記憶領域が改善されている．

7. 比　　較

本論文のプログラム合成を論理プログラムのプログラム合成，プログラム変換などと比較する．ここでは，2節で挙げた本プログラム合成法の特徴 (1) ～ (3) に関して比較する．

特徴 (1) は，プログラム合成をルール生成という小さな問題の集合に分解しているということである．これには，出力するプログラムの構成要素 (等価変換ルール) の相互の独立性が必要である．しかし，論理の枠組ではプログラムの構成要素 (確定節など) の独立性が不十分なので，本合成法のような小問題への分解は行えない．よって，論理プログラムのプログラム合成やプログラム変換はより複雑で難しい問題にならざるを得ない．

特徴 (2) は，出力されたプログラムが高いアルゴリズム記述能力を持つということである．論理プログラミングの解法は等価変換の立場から見ると，unfold 変換のみを使って計算していることに相当する．これは，ルールの発火条件が緩いので計算の制御が難しい．このために生成されたプログラムは非効率になったり，無限ループに陥る場合がある．

特徴 (3) は，本プログラム合成法が，仕様を満たす多数のプログラムを生成可能であり，その中から評価の高いものを選択できることを意味している．論理の枠組でのプログラム合成やプログラム変換の研究は，与えられた仕様やプログラムから 1 つのプログラムだけを生成するものが多く，探索でより良いプログラムを探す有効な方法は知られていない．

以上により本プログラム合成法の特徴 (1) ～ (3) は，本論文の新規性と考えられる．

8. おわりに

本研究では，効率的なプログラムの合成のために等価変換ルールを探索する方法を提案した．また，問題記述から効率的なプログラムを合成するシステムを構築した．本論文で示したシステムは，現在様々な例題を与え実行し，そこから新たな問題を発見してその問題に対応できるように改善している段階にある．今後は，問題記述に一階述語論理を導入し，更に適用可能な問題のクラスを拡大する予定である．

参考文献

1) 小池英勝, 赤間清, 宮本衛市：仕様からの等価変換ルールの生成法, 電子情報通信学会技術研究報告 KBSE98-8, pp.33-40 (1998)

2) 小池英勝, 赤間清, 宮本衛市：等価変換ルールの生成方法の理論的基礎, 情報処理学会研究報告 98-ICS-144, pp.13-18 (1998)

3) 小池英勝, 赤間清, 宮本衛市：等価変換ルールの探索に基づくプログラム合成, 電子情報通信学会技術研究報告 SS99-20, pp.41-48 (1999)

4) 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).

5) 畑山満美子, 赤間清, 宮本衛市：等価変換ルールの追加による知識処理システムの改善, 人工知能学会誌 Vol.12, No.6, pp.861-869 (1997)