

8 J-9

入出力例から既存のプログラムを組み合わせて プログラムを合成する一手法

森 弘昌 永田 守男[†]慶応義塾大学理工学部管理工学科[‡]

1993年1月25日

1 はじめに

近年、コンピュータによるプログラムの自動合成や、知的エディタ、またプログラム自動デバッガシステムなどのプログラム開発支援を行うプログラミングツールの研究が盛んに行われており、ソフトウェア開発全体を支援するCASE (Computer Aided Software Engineering) も徐々に実用化されつつある[1]。また、プログラムの中には同じような機能を有するものがたくさんあり、部分的には同じプログラムが使えるのではないかとという観点から、プログラムの再利用のための研究も再び盛んになってきている。しかし、プログラムの抽象化やその具体化などのいくつかの困難な課題が残っていて、完全な自動再利用に至るまでにはなっていない。

プログラムの自動合成の手法の1つとして、直観的に分かりやすいということから、入出力例を用いたものがこれまでも研究されている[2]。しかし、今までの入出力例を用いた研究では、簡単なプログラムを合成するようなものばかりであった。そこで本研究では、リスト処理プログラムの範囲ではあるが、入出力例から従来のシステムよりも複雑なプログラムを合成できる手法を提案し、実験を通してその有効性を示す。

2 概略

複雑なリスト処理を行なうプログラムには、入力リストを何らかの形に変形し、その結果をさらに処理するという複数の処理が含まれているものと考えた。そこで、まず第一段階で与えられた入出力例を比較し、入力例から必要なリスト(最小変形リスト)だけを抽出する。次に第二段階で、最小変形リストと出力例の間で成立する制約を抽出し、これを用いてライブラリの中か

ら合成のための部品プログラムの候補を見つける。これらの処理の後、第三段階で、どのような組み合わせ方が良いかを探索して合成プログラムを作り上げる。次に、最小変形リストと入力例との間の制約を見つけ、これを用いて抽出プログラムを作成する。最後に2つのプログラムを合成して目的のプログラムを作成する。このように入力からの方向と出力からの方向の両方向から探し出すことで、複数の処理の流れを見つけ出す。以上の流れを図に示す。

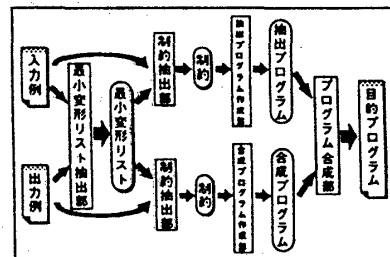


図 1: 本システムの処理の概略

2.1 最小変形リストと制約

前述の処理の中で使う「最小変形リスト」と「制約」の内容をまず示す。

2.1.1 最小変形リスト

入力例と出力例を比較し、入力例から変形したと思われる最小のリストのリスト。たとえば、次のような例が与えられた時、

入力例 : ((a b c) (d e f) (g h i))

出力例 : (c b a i h g)

リスト(c b a i h g)はリスト(a b c), (g h i)が変形してできたものであると考え、

最小変形リスト : ((a b c) (g h i))

となる。

A Method for Synthesizing Programs by Using Existing Programs from Input-Output Examples

[†]Hiromasa Mori, Morio Nagata

[‡]Keio University

2.1.2 合成のための制約

最小変形リストを合成して出力例にするために制約を抽出する。制約というのは例えば次のようなものである。

- ・全ての要素が使われている
- ・リスト中の先頭の要素が使われている

複数例からこれらの制約を抽出し、共通の制約を合成制約とする。

2.1.3 抽出のための制約

必要な要素を入力例から分解、抽出するための制約として次のようなものがある。

- ・リストにおける先頭のリストまたはアトム
- ・リストにおける最後のリストまたはアトム

これらの制約を複数例から抽出し、共通制約を抽出制約とする。

2.2 抽出プログラムと合成プログラムの作成

2.2.1 制約を基にした最小変形リストの合成プログラムの作成

最小変形リストと出力例との間の制約を基にライブラリ中より必要なプログラムを探索し Prolog プログラムを合成する。前出の例でいえば、まず、最小変形リスト内の2つのリストが1つのリストになっているのでリストをつなげる述語が使われるであろうことを推定する。つぎに最小変形リスト中の要素が出力例の中で逆順に使われていることから、要素を逆順にするプログラムが選ばれ、最後にこれらの述語をどのように最小変形リストに対して適用するかについて組合せを考えた探索を行なう。また、再帰的なプログラムであれば分割統治法の基本型に述語を当てはめて探索を行う。

2.2.2 制約を基にした入力例からの要素の抽出プログラムの作成

制約を基に、入力リストに対するプログラムをライブラリ中より探索し入力例から必要な要素を抽出するプログラムを合成する。また、再帰的な取り出しであれば分割統治法的基本的な再起プログラムに述語を当てはめて探索を行う。例えば、次のような例が与えられたとき、

入力例 : ((a b c) (d e f) (g h i))

最小変形リスト : ((a b c) (g h i))

first と last によって (a b c) と (g h i) が抽出されたと考えられる。

3 抽出プログラムと合成プログラムの合成

抽出プログラムと合成プログラムの、2つのプログラムを合成し、目的プログラムとして可能なものであれば1つのプログラムにする。

4 評価と考察

上記の方法に従い、Macintosh 上で Prolog を用い、Prolog のリスト処理プログラムを合成する試作システムを作成し、評価を行った。

Prolog のライブラリ集と Lisp と Prolog のプログラミング入門書の例題を使い、どの程度の述語が合成可能であるかを調べた。

表 1: 合成可能性の評価

種類	対象外	合成可能	合成不可能	合計
個数	317	71	16	404
%	78	18	4	100

対象とすべきプログラムのうち 88% ものプログラムが合成可能であった。この数字は過去の研究と比べても優れているが、計算時間に関しては簡単なもので数 10 秒、少し複雑なものになると数 10 分もかかってしまった。しかし、最近のコンピュータの処理速度の上昇は早く、近い未来には実用的な速度になるであろうことが予測される。また探索問題においては並列処理が可能であるのでアルゴリズム次第でさらに速くなる可能性を秘めている。

4.1 結論

最小変形リストと制約の抽出という2つのアイデアを順に適用することによって、これまでの合成システムよりも複雑なプログラムを自動合成できることを示した。リスト処理以外のプログラムや手続き型のプログラムへの応用などは今後の課題である。

参考文献

- [1] Norman, R.J. and Chem, M.: Working together to integrate CASE, IEEE Software, Vol. 9, No.2, 1992, pp.12-16
- [2] 小泉, 永田: 類推を用いた論理プログラムの合成に関する研究, 修士論文, 慶應義塾大学 (1990)