研究課題別評価

1. 研究課題名：効率的で正しいプログラムの自動生成

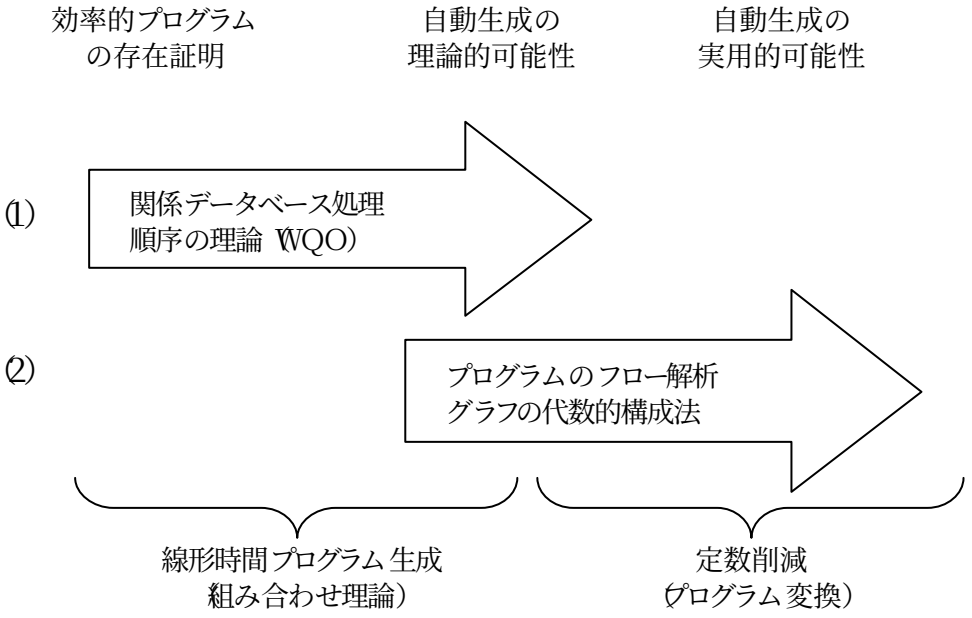
2. 研究者氏名：小川瑞史

3. 研究の狙い：

効率的で正しいプログラムを得るために、計算機が可能なサポートには
(1) 人間の書いたコードの解析に基づく最適化とエラー検出、および
(2) 仕様に基づく自動生成
の二つがある。後者ではプログラムの正しさは仕様と自動生成系が正しければ自動的に保証される。

本研究では、後者の立場をとり、「効率的で正しいプログラムの自動生成」というテーマを設定した。しかし、一般的なプログラムを対象に自動生成を試みれば、構成的証明からのプログラム抽出の研究でも広く知られるように簡単に決定不能性に陥る。したがって、有用な応用領域への適切な制限が必要になる。また、理論計算機科学では論理の枠組みを用いることが多いが、その中にもるべき内容を数学（特に組み合わせ理論）から借りてくることで、対象とする領域では人間のコーディングを凌駕する自動生成を狙いとした。

具体的なアプローチは、応用領域の制限として、
(1) 関係データベースやデータマイニング、
(2) プログラムのフロー解析
を、また組み合わせ理論としてそれぞれ
(1) 順序の理論（Well-Quasi-Order や Kruskal 型定理の構成的証明）
(2) グラフの代数的構成法（グラフの木分割や木幅）
を用いた（図）。これらの組み合わせ理論は、線形時間プログラムの生成、および生成されたプログラムの定数の削減において用いられる。



4. 研究結果：

本研究では、有用な応用領域と計算機の計算可能性の微妙なバランスの上に、上記の図にあげた

(1) 関係データベースにおける順序の理論の応用

(2) グラフの代数的構成法に基づくコントロールフロー解析

の二つの研究課題を設定した。つまり効率的なプログラムの自動生成には、存在証明、自動生成の理論的可能性、自動生成の実用的可能性、の3つの段階がある。効率的アルゴリズムの存在証明はできても実際に生成できない状況は、Kruskal 型の定理を用いた場合にしばしば生じる。これを構成的証明を借りてきて乗り切きろう、というのが第一の課題である。

また自動生成の理論的可能性が示されたとしても、それは実用的可能性をそのまま示している訳ではない。典型的には、自動生成されたプログラムの計算量評価は良くても、実際には定数が爆発して現実的でない状況である。ここでは、実際のコントロールフローグラフは比較的良い構造をもっているという観察を出発点として、グラフの代数的構成法、およびその上の関数型プログラミング＋プログラム変換を用いよう、というのが第二の課題である。

関係データベースにおける順序の理論の応用

当初は課題 1.について進めた。具体的には、関係データベース処理において van der Meyden 現ニューサウスウェールズ大)が 1993 年に提示した未解決問題、

時間概念をもつ不定データベース上の選言 (∨を含む)単項質問処理の線形時間アルゴリズム

を対象とした。この問題は線形時間アルゴリズムの存在は示されていたが、その具体的構成法が知られていなかったものである。

線形時間アルゴリズムの存在証明は、Higman の補題を用いて、充足可能な (∨を含まない)単項質問が多い」という不定データベース間の関係≧が well-quasi-order (WQO) であることから得られる。≧が WQO であることは、任意のデータベースの集合に対し極小元が有限個しかないことを意味しているので、選言単項質問を満たす極小不定データベース $\{D_1, \ldots, D_k\}$ を先に計算しておけば、各 D_i との比較は線形時間でできるので、全体として線形時間で質問処理がなされる。

しかし極小不定データベースが有限個であることがわかっていても、具体的にそれらを決定することは容易ではない。たとえば、単項質問 A<B<C ∨ B<C<A ∨ C<A<B において、その極小不定データベースの集合は以下のようになる。

{A<B<C}, {B<C<A}, {C<A<B}, {A<B, B<C, C<A},

{A<B<A, B<C}, {B<C<B, C<A}, {C<A<C, A<B}, {A<C<A, B<C}, {B<A<B, C<A}, {C<B<C, A<B},

{A<B<A<B, C}, {B<C<B<C, A}, {C<A<C<A, B}, {B<A<B<A, C}, {C<B<C<B, A}, {A<C<A<C, B}.

一番上の行が含まれるのは明らかだが、それ以外についてはパズルを解くような考察を要する。

本研究では、Higman の補題の構成的証明から極小元の計算法を抽出することで、自動的に極小不定データベースの計算、すなわち線形時間質問処理プログラムの自動生成が可能となった。この結果は 2001 年の秋、国際会議 TACS01 で発表し、続いて幸いにも特集号に招待され採録となった。

しかし定数の削減の良いアイデアがみつからなかったこと(fold/unfold 変換は一つの可能性である)、また問題として自然なものがあまり設定できなかったため、中断して課題 2.に力をそそぐことにした。

グラフの代数的構成法に基づくコントロールフロー解析

プログラム解析の自動生成の一般的な手法として、

プログラム解析＝抽象化＋モデル検査

というパラダイムにのっとり、SMVやSPINなどの既存の効率的な実装を適応するアプローチがある。これについては、Java からJimple という3 アドレスコードの中間言語に既存ツールのJ3Cコンパイラとモデル検査系SMVを組み合わせた実装を試みた。既存ツールの活用により、わずか500 行程度の記述でプログラム解析の自動生成系が実際に構成され、素朴な実装であってもかなり実用的な処理時間で実行可能なことを実証した。実際、Java 1.3.1 のjava.math.* クラスの解析を行い、30Kbytes程度のプログラムを数分で解析し、10 数個の不要変数を発見した。しかしプログラムサイズの増大とともに処理時間も急速に増大する傾向も観察され、数万行レベルが限界になりそうな感触がある。

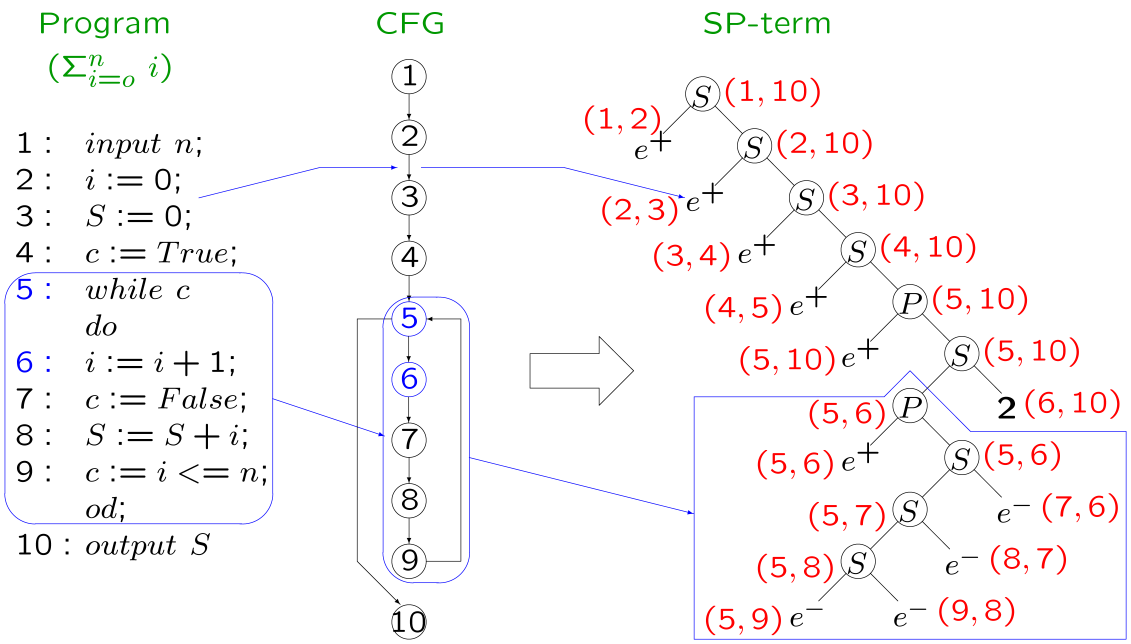
モデル検査は優れた手法であるが、異なるアプローチとして、Thorup (現ATT Labs)が1998年に示した実際のプログラムのコントロールフローグラフは比較的良い構造を持っているという観察（具体的には、GOTO-free C プログラムならば木幅は6以下、Java プログラムではほとんど木幅が3以下などが知られる)に基づき、プログラムのフロー解析のアルゴリズムレベルからの革新を目指した。

もともとグラフの木幅が有界なとき、さまざまな性質の判定が線形時間でできることが知られていた（80年代からのCourcelle、Arnborg や Borie などの研究)。これは論理式による仕様を出発点として、動的プログラミングの手法を適用することで得られている。しかし、論理式を変換する途上で、∀やヨの限量子が現れるごとに指数関数的に状態数（定数係数に対応する)が増加し、線形時間といっても定数が爆発し、理論的な計算量評価にとどまっていた。

本研究では、仕様を論理式のかわりに直接関数型プログラミングで記述し(必要に応じて)融合・組化変換を施すことで劇的な定数の改善が可能となることを示した。関数型プログラムによる仕様記述を可能させるために、多項式型によるグラフ構造の代数的表現が必要となる。木幅の上限 k を制限したグラフの代数的表現として、SP項 SP_k という概念を新たに提案した。SP項は

$$SP_k = e_k(i,j) \mid \mathrm{k} \mid s_k\ SP_k \cdots SP_k \mid p_k\ SP_k\ SP_k$$

として定義される。たとえば、単純なループ構造は k=2 の場合に帰着し、SP項による構成は以下のようになされる。



過去にもいくつかの代数的表現が提案されていたが、SP項の特徴は定数以外の関数記号が s_k（逐次結合）、p_k（並列結合）の二つに限られる点であり、その単純さによりSP項上のプログラミングを容易にしている。具体的なフロー解析の例としては、不要変数解析、ならびに最適レジスタ割当の新しいアルゴリズムを提示し、国際会議 ICFP03において発表した。

5. 自己評価：

当初の理論と実践の溝を埋めたい、という望みは十分にかなえられたとはいえないが、3 年間の研究を経て、ようやくその燭光が見えてきたという状態である。その一方、適切な応用領域を切り出して組み合わせ理論を応用するというアプローチ自体は理論面でかなりの成功であったと考えている。特に、上記の課題 2. において提案したSP項の概念は、試行錯誤の末、最終的に非常に簡潔なものとして提案することができた。このような現実的なグラフの制限を出発点とする研究は、1970 年前後の可約フローグラフ以外にあまり類を見ないものであり、しかもここで提案したSP項は可約フローグラフと独立な概念である。可約フローグラフは深さ優先探索を基本にするのに比べ、SP項ではそれに加え動的プログラミングなどの手法も適用可能とする。そのためより複雑な解析の記述に適している感触を持っている。

SP項に基づくプログラム解析の自動生成は、さきがけの 3 年間の研究期間の最終期に到達したものであり、まだ研究は緒に就いたばかりである。今後、実装も含め、現実的な有効性を実証していきたいと考えている。

6. 研究総括の見解：

正しいプログラムの自動生成はプログラム開発の究極の姿であるが、これまでは生成されたプログラムの実行効率が非常に低いために、特別な領域を除いては一般的には実用化されていない。小川研究者は、この問題にチャレンジし、プログラムのフロー解析を対象に新しい手法「グラフの代数的構成法、その上の関数型プログラミング＋プログラム変換、SP項の導入」という非常に有望な方法を開発した。時間的な制約で研究期間内にこの方法の実用性を示すことは出来なかったが、今後が大いに期待できるものである。

7. 主な論文等：

海外論文誌(4件)

1. Ken Mano, Mizuhito Ogawa. Unique Normal Form Property of Compatible Term Rewriting Systems - A New Proof of Chew's Theorem -, Theoretical Computer Science, 258 (1-2), pp.169-208, 2001.
2. Zurab Khasidashvili, Mizuhito Ogawa, Vincent van Oostrom. Perpetuality and Uniform Normalization in Orthogonal Rewrite Systems. Information and Computation, 164 (1), pp.118-151, 2001.
3. Mizuhito Ogawa. A Linear Time Algorithm for Monadic Querying of Indefinite Data over Linearly
4. Ordered Domains. Information and Computation, 186(2), pp.236-259, 2003, Fourth International Symposium on Theoretical Aspects of Computer Science special issue.
5. Mizuhito Ogawa. Well-Quasi-Orders and Regular ω-languages. Theoretical Computer Science 掲載予定, Third International Colloquium on Words, Languages and Combinatorics special issue.

国内論文誌（4件）

1. 篠埜 功, 胡 振江, 武市 正人, 小川 瑞史. ナップサック問題およびその発展問題の統一的解法, コンピュータソフトウェア 18 (2), pp.59-63, 2001.
2. 篠埜 功, 胡 振江, 武市 正人, 小川 瑞史. 最大重み和問題の線形時間アルゴリズムの導出. コンピュータソフトウェア 18 (5), pp.1-17, 2001.
3. Isao Sasano, Zhenjiang Hu, Masato Takeichi, Mizuhito Ogawa. Derivation of Linear Algorithm for Mining Optimized Gain Association Rules. コンピュータソフトウェア 19 (4), pp.39-44, 2002.
4. 山岡裕司, 胡振江, 武市正人, 小川瑞史. モデル検査技術を利用したプログラム解析器の生成ツール. 情報処理学会論文誌:プログラミング 44 (SIG13/PRO18), pp.25-37, 2003.

査読付国際会議（5件）

1. Zurab Khasidashvili, Mizuhito Ogawa, Vincent van Oostrom. Uniform Normalization beyond Orthogonality. Proceedings of the 12th International Conference on Rewriting Techniques and Applications (RTA01), Lecture Notes in Computer Science 2051, pp.122-136, May 2001, Springer-Verlag.
2. Mizuhito Ogawa. Generation of a Linear Time Query Processing Algorithm Based on Well-Quasi-Orders. Proceedings of the Fourth International Symposium on Theoretical Aspects
3. of Computer Software (TACS2001), Lecture Notes in Computer Science 2215, pp.283-297, October 2001, Springer-Verlag.
4. Mizuhito Ogawa. Call-by-Need Reductions for Membership Conditional Term Rewriting Systems.
5. The 3rd International Workshop on Rewriting Strategies in Rewriting and Programming (WRS03), June 2003, Electronic Notes in Theoretical Computer Science 86(4), Elsevier (http://www.elsevier.nl/gej-ng/31/29/23/135/49/show/Products/notes/index.htt).
6. Mizuhito Ogawa, Zhenjiang Hu, Isao Sasano. Iterative-Free Program Analysis, Proceedings of the 8th ACM SIGPLAN International Conference on Functional Programming (ICFP03), pp.111-123, August 2003, ACM Press.
7. Mizuhito Ogawa. Complete Axiomatization for an Algebraic Construction of Graphs. The Seventh International Symposium on Functional and Logic Programming (FLOPS04) 採録 (Lecture Notes in Computer Science, Springer-Verlag 掲載予定)

学位論文

1. 小川瑞史. 関数型プログラムの自動解析・検証・生成. 東京大学大学理学系研究科（情報科学）, 2002.4.8 授与.