

論文の内容の要旨

論文題目 Research on Retargetable Code Generators
(リターゲットブルコード生成系の研究)

氏名 阿部正佳

リターゲットブルなコンパイラとは、さまざまな言語に対してさまざまなターゲットマシンのコードを生成することが可能なコンパイラである。リターゲットブルなコンパイラを実現するためには、コンパイラを構成する数多くのモジュールの独立性を高め、特定の言語やターゲットマシンに依存したモジュールを、他の言語やターゲットマシンのためのモジュールに自由に入れ換えられなければならない。また、言語やターゲットマシンに依存しない部分は汎用のモジュールとして実現できなければならない。特に、コンパイラのバックエンドはさまざまな最適化モジュールから成り、ターゲットマシンに依存した処理も多く、コンパイラのフロントエンドに比べて高いモジュール性を得ることは難しい。

本研究は、リターゲットブルなコンパイラのバックエンドのモジュール性を高め、そのコード生成系の生産性を向上させるとともに、コード生成技術の適用範囲を広げることが目的としている。コード生成系のモジュール性を高めるために、特に中間言語の設計が重要である。本研究では、まず最初に、リターゲットブルなコンパイラのバックエンドで共通に使うことができる汎用的な中間言語 LIR の設計とそのコード生成系の実装を行った。このような中間言語の設計には次の三つの基準が重要であると考え。ま

ず、さまざまなターゲットマシンの命令を表現できる柔軟さが必要である。そしてその表現力を失わない範囲で可能な限り単純でなければならない。最後に、その中間言語の意味は信頼性の高い実装のために明確に定義されているべきである。

LIR は単なる中間言語としてではなく、独立したプログラミング言語として設計されており、上記の三つの基準を満たしている。LIR は、リターゲットブルコンパイラを開発する COINS (COmpiler INfraStructure) プロジェクトの中間言語として設計され、実際に COINS コンパイラにおいて現在も広く使われており、様々な最適化モジュールを自由に組み合わせる基盤となっている。また、そのコード生成系 TMD は、GCC 風の簡潔なマシン記述を DP マッチングのための書き換え規則に変換する、という方法により、GCC と Burg 系コンパイラの両者の良い点を取り入れて実装された。TMD は簡潔なマシン記述を可能とし、理論的に最適なコード選択を行う。実際に SPARC 及び X86 を記述し性能を評価した結果、GCC や Burg 系の記述と比較して TMD の記述はより簡潔であり、COINS コンパイラ全体として GCC に劣らないコードを出力する。

本論文の第 3 章から第 5 章までにおいて、LIR の特徴的な部分に加え、TMD の設計と実装について述べる。

LIR はコンパイラのバックエンドに共通に使える中間言語として設計されたが、命令選択以後のレジスタアロケーションや命令スケジューリングにおいては、このような中間言語は過剰に具象的であり、中間言語のプログラムは余分な情報を多く含んでいる。これは、命令選択以後のモジュールが、プログラムの意味にほとんど依存せずに、主として、リソース割り当てと命令スケジューリングを行っているためである。

よって本研究では、命令選択以後の処理に適した中間言語を設計することにより、命令選択以後のモジュールの汎用性と独立性を高めることを試みた。実際に、リソース割り当てと命令スケジューリングのための中間言語 ASL (Allocation and Scheduling Language) の設計を行い、そのコード生成系を実装した。レジスタアロケーションと命令スケジューリングの最適化は互いに排他的であり、この最適化問題は NP-HARD であることが知られているが、本研究では最適解を得るために整数線形計画法に基づく手法を利用した。

ASL はリソース割り当てと命令スケジューリングにとって適切な抽象度で設計されているため、通常の CPU 以外にも、同様の問題を含む幅広い分野に適用可能である。実際に本研究では、DNA 実験のために作られたロボットに対する最適なコード生成に応用することができた。このことから、命令選択以後に中間言語を適切な抽象度のものに切り替え、レジスタアロケーション以降のモジュールを独立にすることで、リターゲットブルコンパイラの適応範囲を広げられるということが示された。

本論文の第 6 章において、ASL とそのコード生成系について述べる。第 7 章において、DNA 実験ロボットへの応用について述べる。