逐次型戦略に基づく TRS コンパイラの構築

関 康夫

北陸先端科学技術大学院大学 情報科学研究科

1997 年 2 月 14 日

キーワード： 項書換え系，関数型言語，コンパイラ，正規化戦略.

関数型言語のシステムが開発されだしてから今日に至るまで，実用的なアプリケーションの構築が色々と試みられている．このような関数型言語のシステムとして Elan や Sml, Gofer，OBJ などが存在する．これらの言語システムの特徴として遅延評価，高階関数，多相型の導入などが挙げられる．これらのポピュラーなシステムの他にシンプルなシステムとして酒井らの Cdimple がある．Cdimple は TRS を効率よく実行する手続き型のプログラムを生成する．これは TRS から C プログラムへのコンパイラであり，生成されたプログラムは最内戦略で項を書換えて，これ以上書換えられない項 (正規形) を求める．また，Cdimple はユーザ定義の C プログラムとの接続も可能としている．しかしながら，Cdimple では書換え戦略が最内戦略であるため正規化戦略ではない．一般に正規形が存在すれば必ずそれを求めること (正規化戦略) が望ましい．また他の言語においても正規化戦略となる TRS のクラスははっきりしていない．本研究では関数型言語の計算モデルである項書換え系 (TRS) の効率のよい実現を考え，Cdimple を FB と呼ばれる TRS のクラスに対して正規化戦略になるように改良する．

Orthogonal と呼ばれるクラスにおいては，正規形を求めるためには必ず書換えなければならない場所 (インデックス) を書換えることにより，必ずその正規形を求めることができることが知られている．Huet らは Strongly Sequential(SS) という Orthogonal のサブクラスにおいて，与えられた項からインデックスを探索するオートマトンを示している．しかし，このオートマトンでは一度書換えると次のリデックスを発見するために，項のルートから再びチェックしなければならず効率が悪い．Huet らの SS のクラスのサブセットである Forward-Branching(FB) に限定した場合，インデックスを効率的に発見する手法として Strandh らの FB インデックスオートマトンがある．FB のクラスでは書換え後，次に書換えるべきインデックスは前回に書換えられた項の位置から探索すればよい．

本研究ではこの FB インデックスオートマトンを手続き型言語のプログラムに埋め込むことにより，高速に実行するための TRS コンパイラが構築できると考える．本研究ではこのような手法を用いて Cdimple の改良を行った．

実装において，項はセルのダグ構造で表現し，セルの共有を行っている．このため最外型の戦略で効率を悪くする原因である項を複製することなく書換えを実行できるようにしている．また，ゴミ集めの手法として参照カウンタをそのまま利用した．参照カウンタ方式はリアルタイムでゴミ集めができる反面，参照カウンタの増減による手間がオーバヘッドの問題になっている．最外戦略で書換えて行くことは項の頭正規形をトップダウンに求めて行くことになる．ある項が頭正規形であるならば項の頭のセルにマークを付けることにした．項が頭正規形であるかどうかはこのマークを見て行けばよい．セルに付けられたマークの有無により，更に書換え可能かどうかを判定し，マークが無ければ再び頭正規形を求めようとする．これによりインデックス探索の手間が減少する．

競合する他のコンパイラシステム (Cdimple,Elan compiler,Sml-NJ compiler, Gofer compiler) が生成した実行プログラムとの比較評価を行った．デフォルトの書換え戦略は 本コンパイラ はインデックス戦略，Gofer compiler は最左最外戦略，後の Cdimple, Elan compiler, Sml-NJ compiler は最内戦略である．最内戦略による書換えが有利な問題では本コンパイラによるプログラムは最内戦略を採用するシステムと比較して，数倍以上に遅くなる．また，最外型の戦略である Gofer と比較すれば常に本システムの方が高速であることがわかった．最外戦略による書換えが有利な問題では本コンパイラで生成したプログラムが Cdimple や Elan，Sml-NJ などが生成したプログラムより速い．しかし，Gofer と比較すると若干遅い結果となった．本コンパイラは正規化戦略を実現しているため，正規形を持つ問題を与えても Elan や Sml-NJ，Gofer などでは正規形が求まらないが本コンパイラを使うと正規形が求まる例が存在する．

表 1: results

	Our Compiler	Cdimple	Elan	SML-NJ	Gofer
fact 8	586ms	456ms	74ms	290ms	684ms
fib 20	509ms	478ms	113ms	120ms	722ms
naive reverse 1000	7712ms	9760ms	566ms	1230ms	9132ms
quiq sort 20	[a] 0ms	14086ms	2912ms	1850ms	2 ms
quiq sort 1000	543ms	[b] —	[c] △	[c] △	518ms
fbexamp	[a] 0ms	[d] ∞	[d] ∞	[d] ∞	[d] ∞

[a] $0 < 1$ms
[b] memory error
[c] long time $< \infty$, terminate
[d] non terminate