並列推論マシン PIM/i における KL1 言語処理系の改良

加藤 研児* 久野 英治 六沢 一昭 武田 浩一 大原 輝彦
沖電気工業株式会社

1 はじめに

PIM/i[1, 2] は第五世代コンピュータプロジェクトの一環として開発した並列推論マシンであり、並列論理型言語 KL1 で記述されたプログラムを並列に高速実行する高級言語マシンである。KL1 の言語処理系の実装は ICOT（新世代コンピュータ技術開発機構）で開発された VPIM 処理系 [3] を基に、まず動作することを目的にした実装をし、次いで改良するというステップで行なった。

VPIM 処理系とは仮想マシン (Virtual PIM) 上の処理系であり、実際の計算機上でそれを実現するためには実装する計算機の特性を活かした改良が必要である。本稿では VPIM 処理系を PIM/i に実装する上で行なった改良の内容及びそれによる効果について述べる。

2 PIM/i の構成及び KL1 処理系の概要

PIM/i ハードウェア及び、KL1 言語処理方式の概要を以下に示す。

1. ハードウェア構成の概要
   8 台の要素プロセッサ (以下 "PE" と略す) と 64M-word の共有メモリが共有バスで結合されている。並列キャッシュは更新型プロトコルを用いるため、共有状態のデータに書き込むとブロードキャストメッセージが流れる。また各 PE は 32k-word のローカルコードメモリ (以下 "LCM" と略す) 及び 16k-word のローカルデータメモリを持つ。

2. プロセサ命令の特徴
   命令は 40 ビット固定長の LIW 型であり、最大 3 操作（演算、メモリアクセス、分岐）の同時実行が可能である。また、パイプラインの段数は 3 段である。

3. KL1 処理方式
   KL1 プログラムはまず抽象機械語命令（Prolog における WAM に相当）である kl1-b 命令にコンパイルし、次に PIM/i 機械語命令コード列に展開して実行する。コード量を抑えるため、共有される処理はランタイムルーチンとして実現した。

*Optimization of the KL1 processing system for Parallel Inference Machine PIM/i
Kenji KATO, Eiji KUNO, Kazuaki ROKUSAWA, Koichi TAKEDA, Teruhiko OOHARA,
Oki Electric Industry Co., Ltd.

3 処理系の改良内容

「動作することを目的として実装した KL1 処理系」に対し以下の改良を行なった。

1. 複合 kl1-b 命令の追加
   kl1-b 命令間のコード最適化を行った。例えば表 1のように頻繁に現れる kl1-b 命令列（この場合は klb_mark と klb_write）を新たに作成した一つの kl1-b 命令（klb_mark_write）に置き換え、効率の良い（この場合 1 命令短縮した）機械語コードを生成するように変更した。

2. ランタイムルーチンのコードの改良
   デレファレンス、ユニフィケーション、回収処理等の頻繁に実行されるルーチンに対し、コード最適化を行った。

3. 自動負荷分散機構の改良
   負荷を単純に隣の PE に要求する方法から、1) アイドル状態の PE には負荷を要求しない、2) 既に負荷要求がどれかの PE により出されている場合には要求を出さない、等の変更を行い、負荷分散に伴う不要なバストラフィックを減少させた。

4. 一部ランタイムルーチンの LCM への割り付け
   約 220k-word で構成されるランタイムルーチンの中で頻繁に実行される部分 (約 6%) を LCM へ割り付けてコードキャッシュミスを減少させた。

表 1: kl1-b 命令列の改良の例

kl1-b 命令列	PIM/i 機械語コード列
klb_mark klb_write (改良前)	mark!reg0 = 1. adress = reg1. write(reg0).
klb_mark_write (改良後)	adress = reg1. mark!reg0 = 1, write.

表 2: 実機における実行時間 (単位：秒)

Bench-mark	Queen 11		Pentomino 5×8		Bestpath 100×100	
PE台数	1台	8台	1台	8台	1台	8台
改良前	154	25	496	101	129	79
改良後	104	15	405	76	102	54
後 / 前	68%	60%	82%	75%	79%	68%
R/C	71%	12%	52%	5%	46%	-22%
L/C	—	37%	—	64%	—	34%

R/C　実行時間減少分の中でランタイムルーチンの改良による分が占める割合
L/C　実行時間減少分の中で負荷分散方法の変更分が占める割合

4　処理系の性能向上

改良前及び改良後の処理系を用いてクイーン問題 (Queen)、詰め込みパズル (Pentomino)、最短経路問題 (Bestpath) の 3 つのベンチマークを PIM/i 実機で実行した時の実行時間を表 2に示す。1 台及び 8 台実行のどちらにおいても約 25% 性能が向上した。性能向上の理由の内訳を見ると、1 台実行の場合にはランタイムルーチンの改良によるもの (R/C) が約半分を占め、8 台実行時の場合では負荷分散方法の改良によるもの (L/C) が約 40% を占めた。ランタイムルーチンのみを改良して Bestpath を 8 台で実行したところ、Bestpath のコードと負荷分散ルーチンのコードが偶然にキャッシュミスを起こして実行時間が長くなり、R/C の値が負となった。しかし負荷分散ルーチンを LCM に移すことでこのキャッシュ競合は避けられ、改良後の実行時間は短くなった。

性能向上の要因を詳しく分析するため、前述のベンチマークを規模を小さくしてシミュレータ (1 台及び 8 台) で実行した。その結果を解析した値を表 3に示す。1 台実行における実行サイクルの減少は分岐時のディレイスロットの有効利用による減少分 (D/C) が約 20% を占め、実行命令数の減少についてはランタイムルーチンの改良による減少分 (R/I) が支配的 (約 90%) であった。また、8 台実行にける実行サイクルの減少は共有バスの待ち時間減少の占める割合 (W/C) が約 20% を占めた。なお、改良の結果ディレイスロットの有効利用率 (DS) は約 65% に向上し、1 命令当たりの平均実行操作数 (OP) もわずかながら向上した。また、実行されるランタイムルーチンの 6 割以上、プログラムによっては 99% が LCM 上のコードで実行された。

表 3: シミュレータ実行時における各種の値

Bench-mark	Queen 8		Pentomino 4×4		Bestpath 4×4	
改良	前	後	前	後	前	後
DS	57%	65%	57%	61%	57%	66%
OP	1.38	1.45	1.35	1.36	1.35	1.36
LCM	0%	99%	0%	63%	0%	72%
D/C	17%		18%		23%	
R/I	87%		90%		99%	
W/C	10%		25%		32%	

DS　分岐時のディレイスロット実行率
OP　一命令当たりの実行操作数
LCM　ランタイムルーチンのコードの中で LCM 上のコードをの実行した割合 (改良前は全て共有メモリに置いていた)
D/C　実行サイクル減少分の中でディレイスロットの有効利用分が占める割合
R/I　実行命令減少分の中でランタイムルーチンの減少分が占める割合
W/C　実行サイクル減少分の中でバス待ちサイクルの減少分が占める割合
注)　W/C は 8 台実行の際の値であり、それ以外は 1 台実行の際の値である。

5　おわりに

VPIM 処理系を PIM/i へ実装するにあたり行なった改良とそれによる効果について述べた。1 台実行の場合はランタイムルーチンの最適化によるものが約半分を、8 台実行の場合は負荷分散方法の改良によるものが約 40% を占めた。

謝辞

日頃からお世話になっている新世代コンピュータ技術開発機構 (ICOT) の皆様に感謝します。

参考文献

[1] 佐藤 正俊, 武田 浩一, 大原 輝彦 : 並列推論マシン PIM/i プロセッサの設計, 情報処理学会論文誌, Vol.33, No.3, pp.278–287, 1992.

[2] Taki, K. : Parallel Inference Machine PIM, International Conference on Fifth Generation Computer Systems 1992, pp.50–72, 1992.

[3] 今井 明 : VPIM 処理方式解説書, ICOT Technical Memorandom, TM-1044, 1991.