1K−8

並列推論マシン PIM/k の開発 (2)
— KL1 処理系の予備評価 —

丹野 尚, 沖 健治, 酒井 浩†, 仲瀬 明彦†, 武脇 敏晃‡
東芝ソフトウェアエンジニアリング(株)　†(株)東芝 総合研究所　‡情報通信システム技術研究所

1 はじめに

著者らは通産省第五世代コンピュータプロジェクトの一環として，並列推論マシン PIM/k とその上で動作する KL1 処理系の研究開発を行なっている．今回，試作した KL1 処理系の性能を改善するために，KL1 処理系をソフトウェアシミュレータ [1] 上で動作させて得られたデータを基に、KL1 処理系の性能改善について検討した．本稿では，この内容について述べる．

2 目的

試作した KL1 処理系の性能を下記の観点から調べ，改善を試みる．

- 単一プロセッサの場合の実行プロファイルを収集し．1リダクション内の処理の高速化をはかる．
- 複数プロセッサの場合の実行プロファイルを収集し．負荷分散など並列処理特有の処理の高速化をはかる．

従って，今回は，データ構造の変更やそれに伴う処理アルゴリズムの見直しは行わなかった．

3 データ収集方法

KL1 クラスタのソフト・シミュレータ [1] 上で，KL1 処理系及びサンプルプログラムを実行させて，実行プロファイルを収集する．サンプルプログラムにはレイヤードストリーム法による8クイーンプログラムを用いた．このプログラムでは，リスト処理が多いこと，ゴール分散がしやすいことなどが知られている．なお，ピープホールオプティマイザを適用してオブジェクトレベルでの最適化を行なうことができるが，今回の測定では適用しないオブジェクトを使用した．

シミュレータによる要素プロセッサの並列実行では，すべての命令が1クロックで終了するものとし、各プロセッサ間のスケジューリングはラウンドロビンで1クロックごとに制御が次のプロセッサに渡されるようにした．実行プロファイルでは，共有メモリの初期化，実行結果の表示処理等本来の処理でない部分は除いた．

4 単一プロセッサでの処理に関する検討

レイヤードストリーム法による8クイーンの実行過程の中から，ある1リダクションについて実行トレースを採取した。このトレース中で KL1 処理系の各部の命令実行数の割合を図1の上側のグラフに示す．

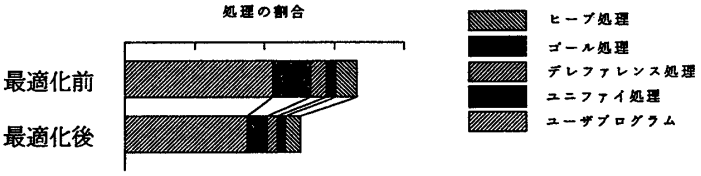

図1　8queenを 1台のプロセッサで実行した時のある1リダクションの各処理内容

図中の各処理の内容は次のとおりである．ヒープ処理では，KL1 処理系で扱う各種のサイズのデータセルの割り付け，回収 (実時間 GC) を行なう．ゴール処理では，ゴールスタックやゴールレコードの操作を行なう．デレファレンス処理では，データへのポインタをたぐり，データセルの回収 (実時間 GC) も行なう．ユニファイ処理では，2引数のユニファイを行なう．

この実行トレースについて，最適化の可能性を検討した結果，下記のような最適化の余地と効果があることが確かめられた．

ピープホール・オプティマイザで最適化可能なもの

(1) 不要な nop 命令の除去

要素プロセッサには，例えば，ストア命令の直後にロード命令を実行してはならないといった制約があるため，それらの命令の間に nop 命令を挿入することがある．この処理をマクロ命令で記述すると，不必要な nop 命令が現れることがある．この最適化は各処理について有効であり，全体の実行命令数をを約0.9%減らすことができる．

(2) 無意味な命令の除去

マクロが連続している場合，両方で同じレジスタ間の move 命令等が現れることがある．この最適化はユーザプログラムについて有効であり，全体の実行命令数を約1.5%減らすことができる．

(3) 分岐命令のディレイドスロットへの命令の埋め込み

分岐命令の直後の命令スロットを有効に使用するため，分岐先の命令をディレイスロットに埋め込む．この最適化は各処理について有効であり，全体の実行命令数を約1.5%減らすことができる．

Development of Parallel Inference Machine PIM/k (2) – Evaluation of its KL1 System –
Hisashi TANNO, Kenji OKI, Toshiba Software Engineering Co. Ltd.,
Hiroshi SAKAI, Akihiko NAKASE, Toshiaki TAKEWAKI, Toshiba Corporation

KL1-B テンプレートの改良

(1) KL1-B テンプレートの改良 (1)

サブルーチンコール及び，それに伴うレジスタのセーブ / リカバーを減らすために， KL1/B 命令のテンプレートの中で，処理の軽いものはマクロで展開し，重いものはサブルーチンで行なう．この最適化は使用頻度の高いヒープ処理，ユニファイ処理について有効であり．全体の実行命令数を約 18.9% 減らすことができる．

(2) KL1-B テンプレートの改良 (2)

例えば，(デレファレンス命令) と (整数であるか否かを判定する) という 2 つの連続するテンプレートについて (整数であるか否かを判定する) の処理を先に行ない，判定が偽であれば (デレファレンス命令) を行なったほうが効率が良い．このようなメインパスを優先にするテンプレートを作成する．この最適化はユーザプログラムについて有効であり，全体の実行命令数を約 1.5% 減らすことができる．

KL1 処理系内部の個別の最適化

KL1 処理系のサブルーチンの内部でさらに別のサブルーチンを呼んでいる場合に、その一部を展開することで，サブルーチンコール及びそれに伴うレジスタのセーブ / リストアを減らす。この最適化は今回の実行トレースについては適用できないが．ユニファイ処理などについて有効であると考えられる．

このような最適化の効果を，全体の実行命令数の違いとして図 1 に示す。

5 複数プロセッサでの処理に関する検討

図 2 はレイヤードトリーム法による 8 クイーンプログラムをプロセッサ台数を変えて実行し，ユーザプログラムと KL1 処理系の各処理別の実行命令数を示したものである．但し， 2 台以上の場合については，複数プロセッサの処理の合計で表している．

図中に現れる各処理の内容は，次のとおりである．里親処理では，里親レコードのメンテナンスを行なう．プロセッサ間ゴール送受処理では，複数プロセッサ間で送られてきたゴールを自分のゴールスタックに積み込んだり，ゴール要求があった時に，自分のゴールスタックから，適当なゴールを選び出す．サスペンド処理では未束縛の変数があった場合，それが束縛されるまで，そのゴールをフックする． d-code 処理は，プロセッサが処理できるゴールがなくなった時 (アイドル状態) や，ゴールの実行に失敗した時，実行される．

この図から下記のことが分かる．プロセッサ台数に注目した場合の各ルーチンの総ステップ数見ると，ヒープ処理，デレファレンス処理，ユニファイ処理，ユーザプログラム処理はほぼ一定のステップ数である．それに対してゴール処理，里親処理，プロセッサ間のゴール送受処理，

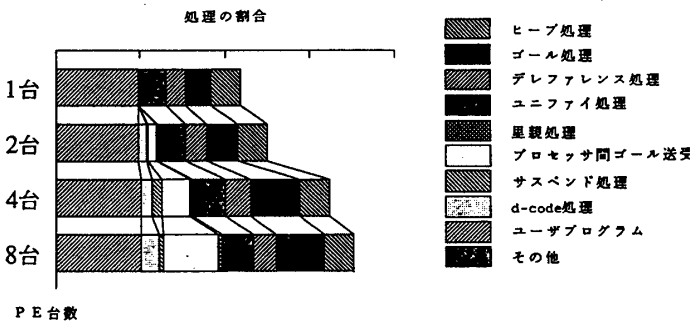

図 2 8queenを複数台のプロセッサで実行した時の各処理内容

サスペンド処理のステップ数は増加している．特にプロセッサ間のゴール送受処理の実行命令数の増加が著しい．従って，プロセッサ間での負荷分散の方式について再検討する必要がある．また，サスペンド処理については，プロセッサ台数が 8 台の時より， 4 台の時の方が実行命令数が多いが，これは実際にサスペンドするゴールが多いためである．

なお，プロセッサが 1 台の場合について単一プロセッサでの処理に関する検討で述べた最適化を適用し，各処理について同等の効果があると仮定すると全体の実行命令数が約 24% 減るものと期待される．

6 結論

試作した KL1 処理系について性能改善の検討を行ない以下のことが分かった．まず，要素プロセッサの実行トレースを基に最適化の余地と効果を検討したところ，実行命令数が，ピープホール・オプティマイザにより約 4%, KL1-B テンプレートの改良により約 20%，それぞれ減少することが分かった．また，実行トレースには現れなかったが， KL1 処理系のサブルーチンの内部を最適化できることも分かった．

プロセッサ台数を変えてレイヤードストリーム法による 8 クイーンを実行し，ユーザプログラムと KL1 処理系の各処理別の実行命令数を調べたところ，特にプロセッサ間ゴール送受処理で，実行命令数がプロセッサ台数の増加にともない，著しく増加していることが分かった．

謝辞

日頃，御指導いただいている ICOT 第一研究室の方々に感謝いたします．

参考文献

[1] 沖ほか，"並列推論マシン PIM/k の開発 (1)KL1 処理系のデバッグ手法とツール"，第 42 回情報処理全国大会講演論文集