

ローカルメモリを持つマルチプロセッサにおける アプリケーションの最適化*

4B-11

鬼頭 宏幸 寺澤 卓也 山本 淳二 服部 大 天野 英晴†
慶應義塾大学 理工学部‡

1 はじめに

現在、大規模な並列処理を行なうために様々な方法が試みられている。その中で、分散共有メモリ型並列計算機は大規模化の有力な候補ではあるが、分散共有メモリの一貫性を維持するためのハードウェアのコストが大きい。そこで、各プロセッサがローカルメモリを持つと共に、スイッチまたはバスなどによって直接アクセス可能な共有メモリを持つ大規模並列計算機が考えられる。このような構成の並列計算機ではローカルメモリを用いることでアプリケーションのパフォーマンスをあげることができる。

本研究では、共有メモリ型並列計算機ベンチマークプログラム集である SPLASH[1] のうちいくつかを小規模ではあるがこのようなローカルメモリを持つ共有メモリ型並列計算機上へ実装し、さらにローカルメモリを有効に利用することを目的としたプログラムのリストラクチャを行う。また、それらの評価を行う。

2 評価用アプリケーション

今回は、SPLASH うち Water, Cholesky, MP3D を用いた。Water, Cholesky に関しては SPLASH に含まれているアルゴリズムをそのまま用い、計算に用いる主なデータをローカルメモリに配置するようにリストラクチャを行う。また MP3D に関してはアルゴリズムの変更を行う。

Water :

このプログラムは N 個の水分子の液体での相互作用と分子内および分子間のポテンシャルのシミュレーションを行なう。一つの粒子(水分子)の全てのデータを一つの構造体にまとめ、プログラム中では粒子はこの構造体の配列として表現されている。各プロセッサには、N 個の粒子を均等に静的に割り当てる。また、並列に実行する間は同じ粒子のセットの計算を同じプロセッサが行なう。構造体に含まれる一つの粒子のデータのうち実際に並列計算中に複数のプロセッサに共有されるのは一部であり、ほとんどのデータは割り当てられたプロセッサのみアクセスされるローカルデータである。

Local 1 : 粒子のデータは全て共有メモリに配置する。

Local 2 : 粒子のデータのうち実際に共有されるもののみ共有メモリに配置し、共有されない粒子のデータはローカルメモリに配置する。

Cholesky :

このプログラムは正定置疎行列のコレスキー分解を行なう。このプログラムでは symbolic 分解、numerical 分解を行ない、numerical 分解の部分をスーパーノード消去法を用いて並列に実行している。symbolic 分解で行列 L の非ゼロの要素の行列中の位置を決定し、それを元に numerical 分解で非ゼロ要素実際の値を決定する。また、プロセッサへのスケジューリングはタスクキューを持ちいて動的に行なっている。

Local 1 : 行列のデータは全て共有メモリに配置する。

Local 2 : 行列のデータのうち symbolic 分解で決定されたデータをローカルメモリに配置し、その他の行列のデータは write-shared であるため共有メモリに配置する。

MP3D :

このプログラムは障害物を含む直方体トンネルの中に超音速で粒子を流し込み、境界(トンネルの壁や障害物)や他の粒子との衝突のシミュレーションを行なう。トンネル内の空間をさらに細かく分割したものをセルと呼ぶ。主なデータ構造は各粒子毎のデータと測定の対象となる空間の情報を示すセルアレイからなる。

Local 1 : 全ての粒子データを共有メモリに配置した。また、測定の対象となる粒子を各プロセッサに均等に静的に割り当て、その粒子の空間中の位置とは関係なくマッピングする。

Local 2 : 測定の対象となる空間中の粒子データをローカルメモリに配置した。また、粒子の流れに沿って空間をプロセッサ分に分割し、プロセッサは割り当てられたセル内に含まれる粒子を計算するように、粒子をプロセッサに割り当てた。また、セルアレイは共有メモリに配置した。なお、この Local 2 におけるリストラクチャは CASPAR[2] として紹介されているリストラクチャの方法である。

3 ターゲットマシン

今回の実装ではローカルメモリを持つ共有メモリ型並列計算機である ATTEMPT-0 を用いた [3]。

ATTEMPT-0 は慶應義塾大学理工学部と Allumer 株式会社が共同開発したバス結合型並列計算機である [3]。プロトタイプ ATTEMPT-0 は PU(Processing Unit) に MC68030, FPU(Floating Point Unit) に MC68882 を使い、4MB のローカルメモリを持っている。そして最大 20 枚のプロセッサボードが非同期バス Futurebus(IEEE

*An Optimization of Applications on a Shared Memory Multiprocessor with Local Memory

†Hiroyuki KITO, Takuya TERASAWA, Junji YAMAMOTO, Dai HATTORI, Hideharu AMANO

‡Keio University

P896.1) によって結合されており、バスを介して最大 16MByte の共有メモリをアクセスできる。

共有メモリに対しては、ライトスルー方式の 64KB の 2ウェイセットアソシアティブマッピングのスヌープキャッシュが用意されている。また、プロセッサ間の同期には、一種の共有メモリである Synchronizer という専用ハードウェアが装備されており、不可分命令 Fetch&Dec を提供している。

4 評価

各アプリケーションの実行条件は次の通りである。

Water : 64 粒子、 $2 \times (1.50 \times 10^{-16})$ sec 間実行

Cholesky : 問題となる行列は lshp.O を使用

MP3D : 300 粒子、50 タイムステップ間実行

アプリケーション	version	実行時間 (s)
Water	Local1	23.73
	Local2	26.31
Cholesky	Local1	34.18
	Local2	34.11
MP3D	Local1	4.981
	Local2	5.986

表 1: 各バージョンの 1PU での実行時間

図 1から図 3に各バージョンの 1PU での実行速度に対する速度比を示す。また、各バージョンの 1PU での実行速度を表 1に示す。

どのバージョンも同期はシンクロナイザを用いて行なった。これにより同期のために共有メモリへアクセスすることはなくなり、同期によるオーバーヘッドは小さくすることができた。

SPLASH はもともとプログラム中におけるデータのローカリティを高いキャッシュヒット率によって実現しようとしている。実際に今回実装したどのバージョンでもキャッシュのヒット率は 95%以上を示した。

また、並列実行時間中のプロセッサ間の通信時間はプロセッサ台数が大きくなって大きく変化することはない、バスの獲得待ちの時間とキャッシュのリプレース時間がその大半であった。

今回の実装のターゲットとなった ATTEMPT-0 は共有メモリに対してキャッシュを持ち、ローカルメモリへのアクセス時間が 250ns であるのに対してキャッシュにリードヒットした場合のアクセス時間は 150ns と高速であるため、表 1にあるように高いキャッシュヒット率を持つプログラムにおいてはデータをローカルメモリに配置することによるリストラクチャによっては実行時間の向上はできなかった。

また、MP3D で行なったアルゴリズムからのリストラクチャは、キャッシュを持つ並列計算機においてかなり有効な手段であるが [2]、図 3から分かるように、ローカルメモリを持つ並列計算機においてもこのリストラクチャは有効な手段であることがいえる。

5 おわりに

今回の実装では ATTEMPT-0 が共有メモリに対してキャッシュを持つため満足な結果が得られなかった。そこで今後、マルチステージネットワークなどスヌープキャッシュの実装が困難な結合形式の共有メモリ型並列計算機で評価を行なう予定である。

参考文献

- [1] Jaswinder Pal Singh, Wolf-Dietrich Weber and Anoop Gupta : *SPLASH : Stanford Parallel Applications for Shared-Memory*, Technical report
- [2] David R. Cheriton, Hendrik A. Goosen, and Philip Machanick : *Restructuring a Parallel Simulation to Improve Cache Behavior in a Shared-Memory Multiprocessor: A First Experience*, ISSMM April 2-4, 1991
- [3] H. Amano, T. Terasawa, and T. Kudoh : *Cache with synchronization mechanism.*, In Proceeding of IFIP Congress 89, August 1989.

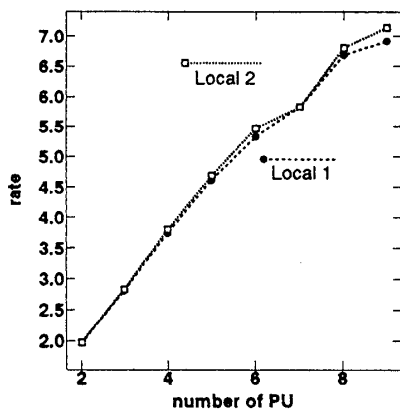


図 1: Water の 1PU での実行速度に対する速度比

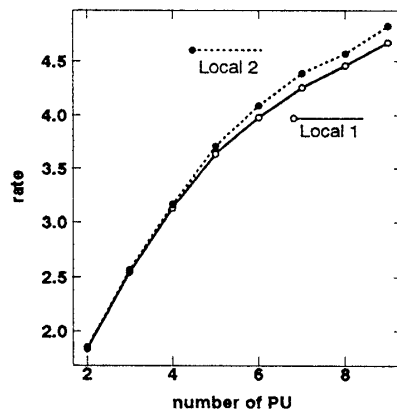


図 2: Cholesky の 1PU での実行速度に対する速度比

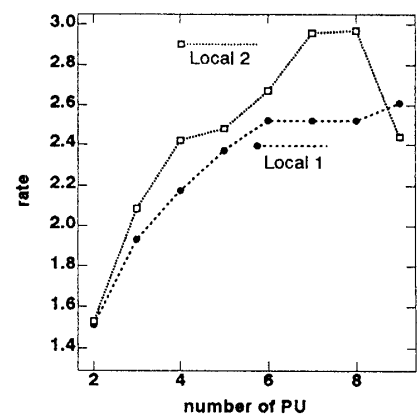


図 3: MP3D の 1PU での実行速度に対する速度比