計算機アーキテクチャ 134－6
（1999．8．2）

ソフトウェアシミュレータ上での SPLASH-2ベンチマークの挙動に関する研究

高木 将通　　松本 尚　　平木 敬
{takagi-m, tm, hiraki}@is.s.u-tokyo.ac.jp
東京大学大学院理学系研究科情報科学専攻

概要

RT レベルソフトウェアシミュレータ (MISC[1] シミュレータ) を用いて、メモリシステムのパラメタに対する SPLASH-2[4] ベンチマークの実行時間の変動を評価する。共有バスを使用した集中共有メモリマルチプロセッサシステムの動作を忠実にシミュレートするシミュレータ上で評価する。

Execution Time Behaviour of SPLASH-2 Benchmark on Software Simulator

Masamichi Takagi, Takashi Matsumoto, and Kei Hiraki
Department of Information Science, Faculty of Science, University of Tokyo

Abstract

We evaluate the execution time behaviour of SPLASH-2[4] benchmark programs with RT-level simulator(MISC[1] simulator). We focus the behaviour with change of the parameters of the memory system. The simulator simulates shared bus based cenralized shared-address space multiprocessors, and reflects the delay and interaction of the memory system.

1 はじめに

SPLASH-2[4] (Stanford ParalleL Applications for SHared memory-2) は、cache-coherent な集中あるいは分散共有 address space を持つ並列計算機のためのベンチマークであり、広く用いられている。

このベンチマークの使用、結果の参照の際には、ベンチマークプログラムが本来持つアーキテクチャに依存しない性質と、それらの性質が組み合わさった結果として現れる、多数のパラメタを持つシステム上での実行時間の振る舞いといった性質を知る必要があり、S.C.Woo らの論文 [4] は主に前者について述べている[1]。本稿では後者に主眼を置く。

本稿では共有バスモデルを採用した、集中共有メモリのシステム上でベンチマークプログラムを動作させる。プロセッサ、メモリシステムはおのおの実機に近いパラメタを持ち、実機に近い振る舞いをする。それゆえ、メモリシステムのもつ latency、bandwidth といったパラメータや、それに伴うバスの contention の影響が考慮される。

それらを考慮した上で、実行時間ベースにおいて、メモリシステムの L2 Cache のサイズ、ラインサイズ、共有バスの幅、メインメモリのバンク数、レイテンシといったパラメタに対する、ベンチマークの振る舞いを調べる。

本稿の構成は以下の通り。2 章で SPLASH-2 について説明する。3 章で MISC シミュレータについて説明する。4 章で 評価環境を述べる。5 章で評価結果を示す。6 章で 主張をまとめる。

2 SPLASH-2

SPLASH-2[4] は、並列計算機を対象としたベンチマークであり、cache-coherent な集中あるいは分散共有 address space を持つシステムのためのプログラムである。

4 つのカーネルプログラムおよび 7 つのアプリケーションからなっている。今回使用したもののみ挙げる。Kernel: FFT(高速フーリエ変換)、LU(LU 分解)、Radix(Radix ソート)、Application: Ocean(海洋の移動問題の解法)、Water-Nsquared(水分子の力とポテンシャルの計算 ($O(n^2)$ のアルゴリズムを使用))。全て C で記述されている。デフォルトの問題サイズがそれぞれに与えられている。

各プログラムの実行命令数等の性質は S.C.Woo らの論文 [4]、J.P.Singh らの論文 [3] に示されている。

3 MISCシミュレータ

MISC シミュレータは本来生産者–消費者問題を解決するハードウェア機構を持つ MISC[1] の機能を評価するためのものであるが、MIPS R3000 の RT レベルのシミュレータとしての機能も持っている。メモリシステムはバスの request、reply のレベルでシミュレートされている。

ISA は R3000+R3010(FPU)+MISC 命令[2]である。整数演算命令、浮動小数点演算命令のレイテンシ、スループットは R3000、R3010 の値と等しい。同時命令 issue 数は 1 でパイプライン構造をもつ。

[1] そこでは評価に用いられたメモリモデルは PRAM モデルであり、メモリシステムは命令の実行タイミングに影響を与えないという条件であった。また浮動小数点、メモリアクセスを含む全命令は 1 サイクルで終了するという条件であった。

[2] ベンチマークプログラムで使用したものは RREQ(ハードウェアバリア命令 [2]) のみである。

レジスタへの書きこみ命令は実際の書きこみ終了を待たずに retire し、書きこみが終了していないレジスタに対する読み込みのみ block するのでロード、ストアの隠蔽ができる。L1 instruction キャッシュ、L1 data キャッシュ、共用 L2 キャッシュを持つ。L1、L2 間に一本のバスがあり、これはアトミックトランザクション方式である (L2 が L1 を invalidate する際にもこのバスを使用。)。L2、メインメモリ間に共有バスがありこれはスプリットトランザクション方式である。このバス上で L2 キャッシュがバススヌープを行う。メインメモリは 4K byte ページ単位の複数バンク構造をとっている。

4 評価環境

デフォルトのシステムのパラメタは以下の通りである。L1 instruction cache、L1 data cache は共に 2K byte で、それぞれ 2-way set associative、direct-map である。line size は共に 16 byte。L1 data cache は write-through である。L2 shared cache は 8K byte、2-way set associative、line size は 64 byte。L1、L2 間のバス幅は 16 byte、L2、メインメモリ間のバス幅は 8 byte である。CPU クロックとバスクロックは同一としている。キャッシュヒット/ミス時のペナルティは L2 ヒット時は 8 クロック、L2 ミス時は 24+(L2 Line Size)/(Bus Width) クロックである。L2 キャッシュを 8K byte にしたのは Radix、FFT、Ocean に関して実行時間に関して重要なワーキングセットが L2 キャッシュに収まらない場合を想定し scale したためである。8K byte は scale された結果の代表である。Water-Nsquared、LU に関しても第 1 ワーキングセットが L1 に収まり、第 2 ワーキングセットが L2 に収まらないケースを想定して scale したサイズに 8K が含まれる。

L2、メインメモリ間の split-transaction のバスにおいて、read request、invalidate コマンドの長さは 2 クロック分、reply、write 時は最初の 1 ブロックが 2 クロック、以降のバースト転送はそれぞれ 1 クロックを占める。read 時に request コマンドがバスに出終わってから、reply コマンドが出るまでの間 (この間はバスが空いていて他のコマンドが入ることができる) は最低 12 クロック。

SPLASH-2 ベンチマークの内、LU-CB 及び NCB(contiguous and non-contiguous block allocation)、FFT、Radix、Ocean-CP(contiguous partition allocation)、Water-Nsquared を使用した。

ベンチマークプログラムに与えた問題サイズ、S.C.Woo らの論文 [4] に述べられているベンチマークプログラムの第 1 ワーキングセット (WS1)、同じく第 2 ワーキングセット (WS2) [3]、シミュレータ上での 16 プロセッサ時の実行クロック数を表 4 に示す。括弧内はデフォルトの問題サイズおよびワーキングセット (16 プロセッサ時) を示す。

[3] これらはキャッシュサイズ対キャッシュミス率のグラフの Knee から得たものである。

5 評価結果

5.1 メインメモリバンク数

Radix はメモリシステムに与える負荷がベンチマーク中最も高い。この Radix のアクセス頻度の程度および実行時間に対するメモリバンクの影響を見るためメインメモリバンク数を 1 から 16 にしたときの Radix の 16 プロセッサ時の実行時間を測定した。結果を図 1 に示す。

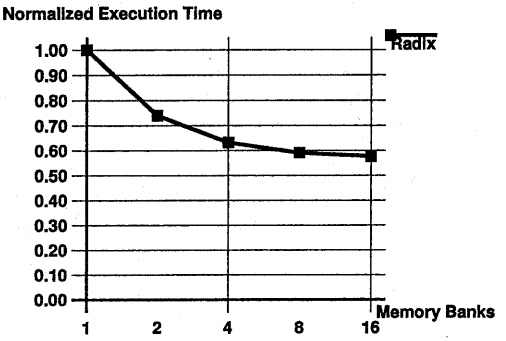

図 1: Execution time versus memory banks.

実行時間がバンク数増大と共に減ることからバスアクセスの頻度がスプリットトランザクション、複数メモリバンクが働く程度には高いことが分かる。また 4 バンクから 16 バンクに増やしたとき実行時間が減ることから、4 バンクのときに同時アクセスに対する retry が起きていることが分かる。

同時に Radix のスピードアップに与える影響を調べた。メインメモリバンク数を 4、16 としたときの Radix のスピードアップのグラフを 2 に示す。

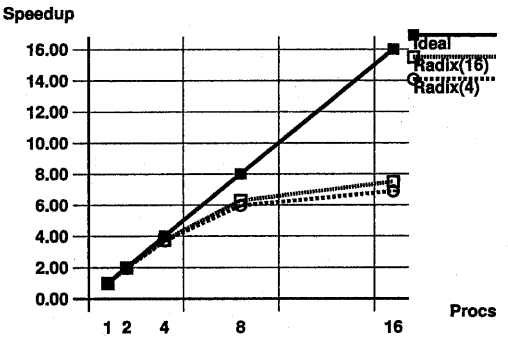

図 2: Speedup versus memory banks.

Radix においてはプロセッサ数が増大すると key、local histogram のアクセスの同時要求頻度が高まるため、バンク数の効果がより高まる。

5.2 request、reply 間のバスの空き時間

Radix のバスの同時アクセス頻度及び共有バスにおける request と reply のペアの占めるバス時間の実行時間に対する影響を調べるために、プロセッサのメインメモリアクセス時のペナルティを変えずにバスの request コマンドと対応する reply コマンド

表 1: SPLASH-2ベンチマークに対する問題サイズ、第1ワーキングセット (WS1)、第2ワーキングセット (WS2)、16プロセッサにおける実行クロック数。括弧内はデフォルト (16プロセッサ時)。CB/NCBは Contiguous/Non-contiguous block allocation。

プログラム	問題サイズ	WS1(byte)	WS2(byte)	実行クロック数
FFT	65536(65536) 点	8K(8K)	512K(512K)	42M
LU	256×256(512×512) の行列	4K(4K)	64K(256K)	CB:47M NCB:62M
Radix	radix 1024, $0 \leq N \leq 1048576$ の 512K 個 (1M 個) の整数のキー	16K(16K)	128K(256K)	19M
Ocean	130×130(258×258) Ocean	8K(4K)	256K(1024K)	71M
Water-Nsquared	6^3=216(8^3=512) 個の水分子、 1(3) step	2K(2K)	32K(64K)	101M

の間にバスが空く時間を1クロックから11クロックまで変化させてみた。結果を図3に示す。

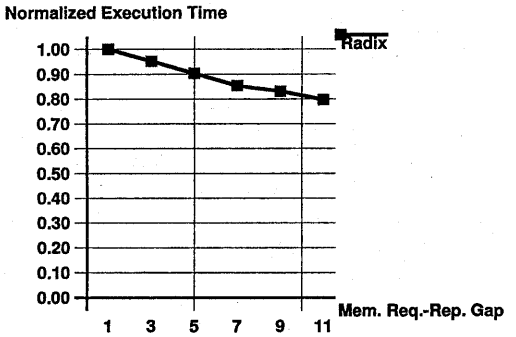

図 3: Execution time versus request-reply gap size.

バスが空く時間が短くなるにつれ実行時間が増大することから、同時アクセスの頻度が分かる。

5.3 メインメモリのレイテンシ

メインメモリのレイテンシの Radix の実行時間に対する影響を調べた。

共有バス上に request が出終わってから reply が出るまでの時間を2クロックから256クロックまで変化させたときの Radix、Ocean-CP の実行時間のグラフを図4に示す。

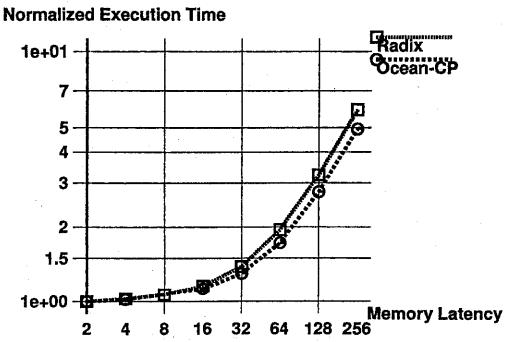

図 4: Execution time versus memory latency.

間隔が256クロックにおいても実行時間は6倍程度である。これにより、request と reply の間のバスの空き時間が有効利用されていることが分かる。また、ロード、ストアの隠蔽の度合いが分かる。

5.4 L2キャッシュサイズ

L2キャッシュサイズを 1K byte から 512K byte まで変化させたとき (L1キャッシュのサイズは 1K byte) の各プログラムの実行時間を図5に示す。

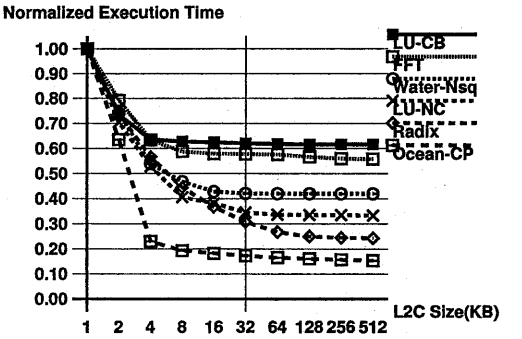

図 5: Execution time versus L2 cache size.

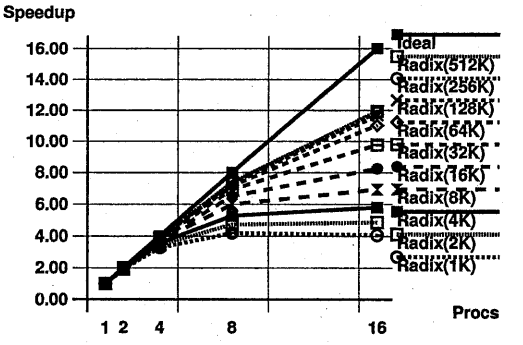

図 6: Speedup versus L2 cache size.

Knee は FFT では 8K、Water-Nsq では 16K、LU-NCB では 8K、LU-CB では 4K、Ocean-CP では 4K 付近にあると考えられ、これは表4における値と大きくは異ならない。LU-NCB が LU-CB に比較してキャッシュサイズに対する実行時間の減少の度合いが大きいのは、LU-CB の 4-D array 表現に対し、LU-NCB の 2-D array 表現は conflict miss を起こしやすいためである。Radix には Knee らし

きものが見られない。これはRadixが、第1ワーキングセットとなり得るlocal histogramを頻繁にアクセスするが、同時に第2ワーキングセットとなり得るkeyにもアクセスするため、前者のキャッシュ上での再利用率が低くなるためである。Ocean-CPではメモリアクセス頻度が高く、4K以下でミス率が非常に高くなって実行時間が急増している。

また、各キャッシュサイズにおけるRadixのスピードアップを図6に示す。

Radixに関してはデータセットがキャッシュに収まっていないのでL2キャッシュサイズがスピードアップに大きく関わってくる。

5.5 問題サイズ

Water-Nsquaredはシミュレーション時間の制限から、小さな問題サイズが選ばれることが多い。そこで問題サイズを5^3=125、6^3=216、7^3=343(デフォルトは8^3=512)としたときのWater-Nsquaredのスピードアップのグラフを図7に示す。

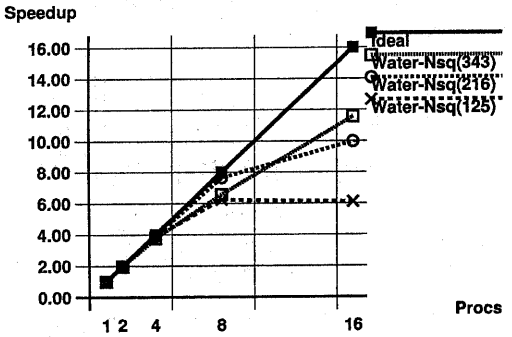

図7: Speedup versus problem size.

問題サイズ125は小さすぎ、台数効果が出ないことが分かる。また、8プロセッサのときは問題サイズが216のときの方が343より効率が良く、16プロセッサでそれは逆転している。これは、プロセッサ間通信の増加率が、問題サイズが小さいほど大きく、また、容量ミス率はプロセッサ数と共に減少するが、データセットが大きいほど1プロセッサ時の値が大きいので、それらの和がほとんどを占めるメモリアクセス時間の減少率が異なるためである。

5.6 共有バスの幅

バス幅を増大させると、L2ミス時のペナルティ、バス使用時間が減少する。バス使用量が多いほど、またバス使用が集中しているほどバス幅増大の影響は大きくなる。共有バスの幅を8byteから64byteまで変えてみた際の各アプリケーションの16プロセッサ動作時の実行時間の変化を図8に示す。

16プロセッサ時の、メモリアクセスに要する時間対計算時間の比が大きいものほど影響が大きいことがわかる。

つまり、Radixはこの比が突出して大きく、Ocean-CP、LU-NCBは比較的大きく、Water-Nsquared、FFTは比較的小さく、LU-CBは非常に小さいと言える。

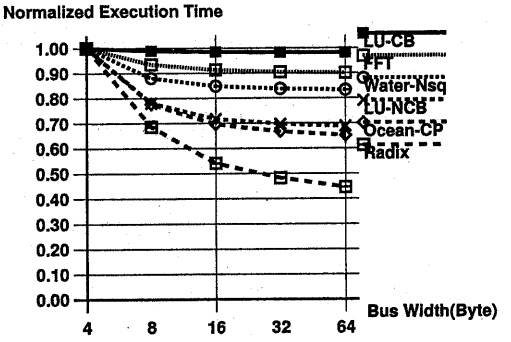

図8: Execution time versus shared bus width.

Radix、Ocean-CPのキャッシュミス率は他に比べ非常に高い(25～40%)[4]。トラフィックに関してOceanにおいては容量ミスの割合が多く、Radixにおいては容量ミスとプロセッサ間通信の両方が多い[4][4]。

これらから以下のように考えられる。Radixはキャッシュの容量ミスの計算時間に対する比が大きい。プロセッサ間通信の時間の計算時間に対する比も大きい。Ocean-CPはキャッシュの容量ミスの計算時間に対する比が大きい。プロセッサ間通信の時間の計算時間に対する比は小さい。LU-CB、FFT、Water-Nsquaredは容量ミス及びプロセッサ間通信の時間の計算時間に対する比が小さい。

LU-NCBは先述の理由でキャッシュが有効利用されないため、LU-CBと比べ容量ミス、プロセッサ間通信の時間の割合が増大していると考えられる。

次に、共有バスの幅を8byteから64byteまで変えて見た際のスピードアップについて見る。Radix、Ocean-CP、LU-NCB、Water-Nsquared、LU-CB、FFTのグラフを図9、10、12、11、13、14に示す。括弧内は共有バスの幅を示す。

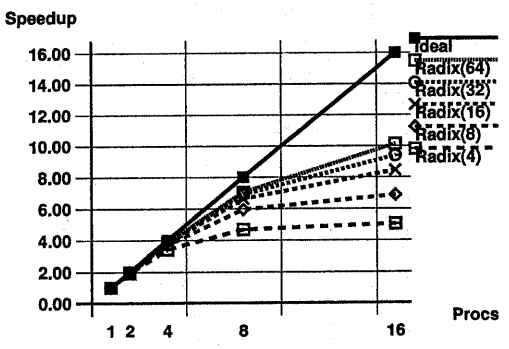

図9: Speedup versus shared bus width.

スピードアップに対する影響は、プロセッサ数が増大したときメモリアクセスに要する時間対計算時間の比が増大するものほど影響が大きい。また、プロセッサ数が増えたとき、同時メモリアクセスの頻度が高くなるものは、この点でプロセッサ数が増

[4]論文[4]における remote writeback もプロセッサ間通信とみなしている。

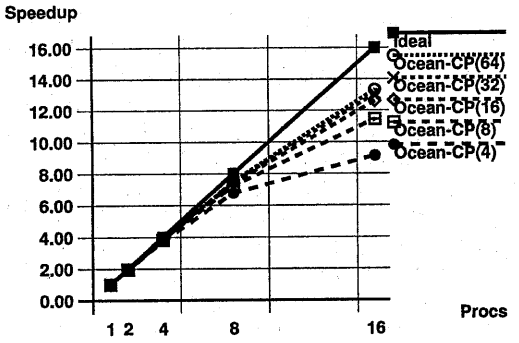

図 10: Speedup versus shared bus width.

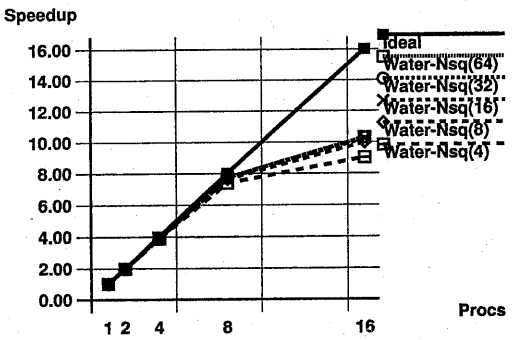

図 11: Speedup versus shared bus width.

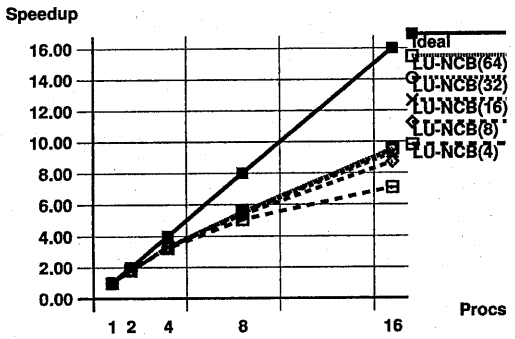

図 12: Speedup versus shared bus width.

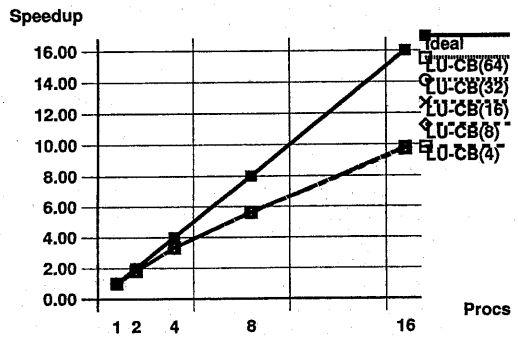

図 13: Speedup versus shared bus width.

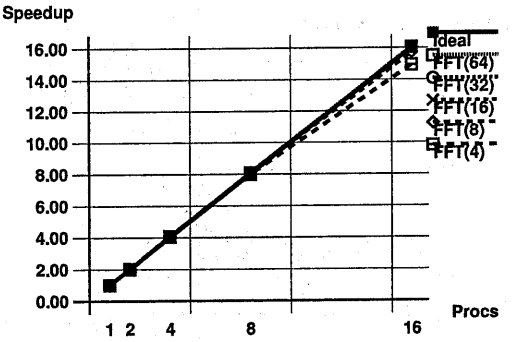

図 14: Speedup versus shared bus width.

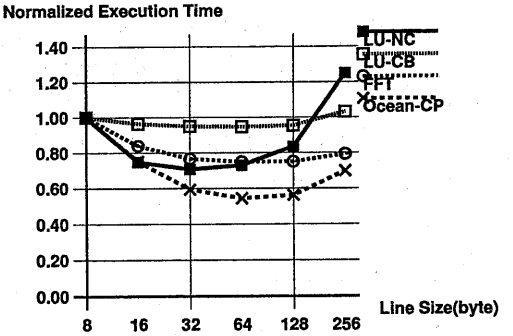

図 15: Execution time versus L2 line size.

えるほどバス幅の効果は大きくなる。

FFT、Ocean-CP、Radix はプロセッサ数増大に対するトラフィック増大が比較的少なく、Water-Nsquared、LU は比較的大きい [4]。また Radix、Ocean-CP のキャッシュミス率は他に比べ非常に高い (25 ～ 40%) [4]。

これから以下のように考えられる。LU、Water-Nsquared はもともとメモリアクセスに要する時間対計算時間の比が小さいので影響が小さい。FFT はこの比の増大が小さいため影響が小さい。Ocean-CP、Radix はワーキングセットがキャッシュに収まっていないので、メモリに同時にアクセスすることの影響が大きい。

5.7 L2 ラインサイズ

ラインサイズの増大に伴ってプリフェッチとしてのプログラムの spatial locality の有効利用の効果と、false sharing の影響、また、ライン数の減少による影響、キャッシュミス時のレイテンシの増大といった影響がある。

この影響をみるために、L2 キャッシュのラインサイズを 8 byte から 256 byte に変えた時 (L1 キャッシュのラインサイズは 8 byte) の実行時間のグラフを図 15、図 16、図 17 に示す。

Radix の括弧内は共有バスの幅 (byte) である。Water-Nsquared の括弧内は問題サイズと共有バスの幅である。

L2 キャッシュのキャッシュラインの replace や invalidate の際にそのアドレスに相当する L1 キャッ

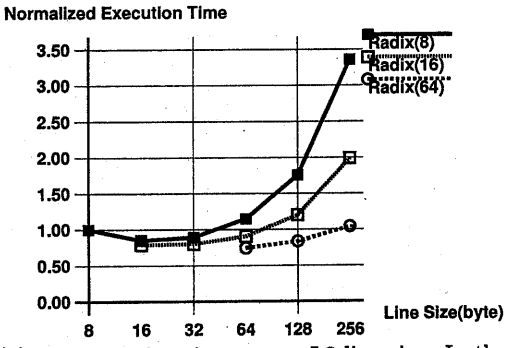

図 16: Execution time versus L2 line size. In the bracket is the shared bus width.

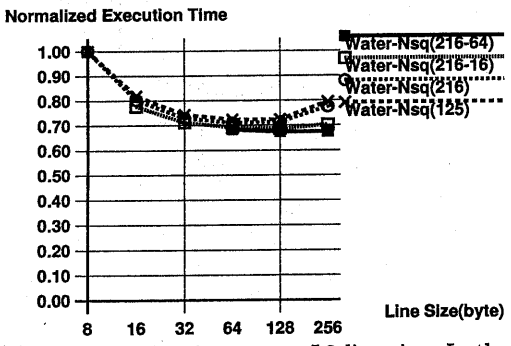

図 17: Execution time versus L2 line size. In the bracket is the problem size and shared bus width.

シュのキャッシュラインを invalidate する。そのために、L2 のラインサイズが 256 byte のときはサイズが 2k byte の L1 の 1/8 が invalidate され得る。このために L2 キャッシュのラインサイズが大きいときに L1 キャッシュの機能が低下し、L1 キャッシュのミス率が増大し、実行時間が増大する。この効果はプログラムを問わず起こる。

LU-CB は行列をブロックに分割していて、そのブロックが連続領域に割り当てられているので spatial locality が比較的長いラインサイズにおいても有効利用できる。

LU-NCB においてはブロックに分割されたブロックは全体の大きな二次元配列としての行列の一部となっているのでそのブロック (16×16=128 byte × 16) の 1 行よりもキャッシュラインが大きくなってしまうと false sharing が頻発する。

Ocean-CP も分割された領域は連続領域に割り当てられているので spatial locality が有効利用できる。キャッシュラインサイズが 128 byte 以上のとき実行時間が増大しているのは、Multigrid 法において行方向の隣接ブロックのデータを利用する際に長いラインのため必要でないデータを読んでしまうためと考えられる。また、実行時間対 L2 キャッシュサイズのグラフから推定できるように、Ocean-CP に関しては L1 キャッシュのミス率は重要であるため、先に述べた L1 キャッシュのミス率増大のためとも考えられる。

FFT は転置の phase において、キャッシュ及びキャッシュライン使用効率を上げる工夫として、8×8 (=128 byte×8) の部分行列単位で転置を行う。このとき他のノードの部分行列を 128 byte の行単位で読むので 128byte まではラインサイズと共に実行時間が減るといった減り方をしている。

Radix に関しては、32byte 以降は実行時間が減らず、128 byte を超えるあたりから急激に実行時間が増大する。これは以下のように説明される。

一つのプロセッサがある bin に複数個書きこむが、この個数は $n/(r \times p)$(n は key 数、r は radix、p はプロセッサ数) に比例する。512K の整数値 (4byte) のキー、16 プロセッサのとき、この値は 128 byte に相当する。それゆえ、ラインサイズが 128 byte 以上のとき invalidate が頻繁に起こることになる。

この結果は S.C.Woo らの論文 [4] にも述べられており定性的に一致する。

Water-Nsquared に関してはあるプロセッサが使用する粒子の構造体の配列の要素 (600byte) のフィールド (例えば重心位置:8 byte×3) がキャッシュラインの整数倍でないのでプロセッサ間の値のやり取りの際 spatial locality が最大限利用できなかったり、キャッシュラインに粒子の構造体が複数入ってしまい、複数のプロセッサがそこを使用するようになって false sharing が起こることがあり [3][4]、256 byte 付近で実行時間がやや増大していてこの効果が見て取れる。Radix、Water-Nsquared のラインサイズが大きいときの実行時間の増加が共有バス幅を大きくすることで抑えられる様子もわかる。

6 まとめ

SPLASH-2[4] ベンチマークについて、共有バスを使用した集中共有メモリマルチプロセッサシステムのシミュレータ上で実行時間を評価した。メモリシステムを忠実にシミュレートすることによって様々なパラメータの組み合わせの結果としての実行時間を調べることができた。ベンチマークの実行時間の、メモリシステムのパラメタに対する振る舞いについて調べた。

参考文献

[1] T.Matsumoto, T.Tanaka, T.Moriyama, and S.Uzuhara : MISC: A Mechanism for Integrated Synchronization and Communication Using Snoop Caches. *Proc. of the 1991 Int. Conf. on Parallel Processing* , Vol. 1, pp.161–170 (August 1991).

[2] T.Matsumoto : Elastic Barrier: A Generalized Barrier-Type Syncronization Mechanism. *Transactions of Infomation Processing Society of Japan* , Vol. 32, No. 7, pp.886–896 (July 1991) (in Japanese).

[3] J. P. Singh, W.-D. Weber, A. Gupta : SPLASH: Stanford Parallel Applications for Shared Memory. *Computer Architecture News* , 20(1), pp.5–44 (March 1992).

[4] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, A. Gupta : The SPLASH-2 Programs: Characterization and Methodological Considerations. *Proc. of the 22nd Annual Int. Symp. on Computer Architecture* , pp.24–36 (June 1995).