計算機アーキテクチャ 149－19
（２００２．８．２３）

投機的コヒーレンス制御にともなうネットワークトラフィックの評価

乗貞由華[†] 鈴木圭介[†] 多田野陽介[†]
古川文人[†] 大津金光[†]
横田隆史[†] 馬場敬信[†]

DSMシステムにおいて、リモートメモリへのアクセスレイテンシの問題と、ネットワークトラフィックの問題の解決は重要な課題である。我々はハードウェアによってデータの一貫性を保つDSMシステムにおいてreadおよびwriteアクセス両方のコヒーレンスオーバーヘッドの削減を目的とした投機的コヒーレンス制御機構SCCMを提案してきた。本機構を適用することにより、投機的コヒーレンス処理が成功した場合には、ネットワークトラフィックの削減が期待できる。本論文ではその投機的コヒーレンス処理にともなうネットワークトラフィックに関して評価した。その結果、メッセージ受信回数がBarnesで0.40%ほど減少し、Raytraceで0.97%増加することがわかった。

Evaluation of Network Traffic Caused by Speculative Coherence Control

Yuka NORISADA,[†] Keisuke SUZUKI,[†] Yousuke TADANO,[†]
Fumihito FURUKAWA,[†] Kanemitsu OOTSU,[†]
Takashi YOKOTA[†] and Takanobu BABA[†]

It is important to reduce the overhead of coherence control caused by memory access and network traffic.In order to reduce the overhead caused by the memory access of both read and write, we proposed the speculative coherence control mechanism for Cache Coherent DSM systems which maintains the consistency of data by hardware. If the speculative coherence control transaction is successful, we predict that the speculative coherence control mechanism reduces network traffic. In this paper, we evaluate network traffic caused by speculative coherence transaction. The results reveal 0.40% reduction of messages in the Barnes program and 0.97% increment in Raytrace.

1. はじめに

分散共有メモリ(以下、DSM)システム[1]は、相互結合網を介した通信処理で実現される。故にDSMシステムではリモートメモリへのアクセス(以下、リモートアクセス)が発行されてから完了するまでのレイテンシ(以下、リモートレイテンシ)がローカルメモリのレイテンシに比べ長い[2)3)]ため、問題となっている。また、リモートアクセスによってネットワークが混雑すると、それがボトルネックとなり性能向上を妨げることが問題となっている。

そこで、この問題の解決方法として、ハードウェアによるCache Coherent DSMシステム(以下、CC-DSMシステム)における投機的コヒーレンス処理に関する研究が行われている。投機的コヒーレンス処理は、データの共有状態を動的に予測し、その予測情報をもとに行われる。予測のための処理と投機的コヒーレンス処理はユーザに完全に透過な形でハードウェアが行う。したがって、ユーザに提供する開発環境への影響はない。

我々はCC-DSMシステムにおいてreadおよびwriteアクセスのリモートレイテンシを短縮することを目的とした新たな投機的コヒーレンス制御機構(Speculative Coherence Contorol Mechanism(以下、SCCM))[4]を提案してきた。本機構では投機を行うことによって投機的コヒーレンス処理が成功した場合、リモートレイテンシが削減でき、ネットワークトラフィックを削減できることが期待される。本論文では、SPLASH-2[5]を用いて本機構のシミュレーションを行いネットワークトラフィックにどのような影響を与えるかを評価する。

2. 投機的コヒーレンス制御機構

本論文では、以下に示すDSMシステムとキャッシュコヒーレンスプロトコルを前提として議論する。本章では、2.1節でモデルとなるDSMシステムについて述

† 宇都宮大学工学部情報工学科
Department of Information Science, Faculty of Engineering, Utsunomiya University

べ、2.2節でキャッシュコヒーレンスプロトコルについて述べる。

2.1 DSMシステムとノード内の構成

DSMシステムとノード内の構成を図1に示す。このDSMシステムは、複数のノードがネットワークで結合された物であり、2Dメッシュ構造である。そのネットワークは、全二重で東西南北にReplyとRequest用に1つずつリンクを持っているので1ノードは合わせて16本のリンクを持っている。リンク幅は1本あたり16ビットである。また、2ノード間のメッセージの到着順序を保証される。各ノードは、1台のプロセッサコア、キャッシュ、DSMコントローラ、ディレクトリおよびメモリ、ネットワークインターフェース(以下、NI)およびSCCMから構成される。また、キャッシュはライトバック型、ディレクトリはフルマップ形式である。

2.2 キャッシュコヒーレンスプロトコル

キャッシュコヒーレンスプロトコルは無効化型である。また、キャッシュラインとメモリブロックは1対1に対応しており、それらの状態遷移はそれぞれ図2と図3に示される。なお、図2と図3でのX/Yは、Xが要求、Yが遷移後の応答を意味する。

キャッシュラインの状態は、Modified、Exclusive、Shared、Invalid、(以下、それぞれM、E、Sc、I)の4状態をとる。Mは、メモリブロックのコピーを専有し、その内容がメモリと一致していないことを示す。Eは、メモリブロックのコピーを専有し、その内容がメモリと一致していることをを示す。Scは、他のノードに同一内容のラインがあることを示す。Iは、そのラインの内容が無効であることを示す。

メモリブロックの状態は、Uncached、Shared、Private、Busy-Shared、Busy-Private(以下、それぞれU、Sm、P、BS、BP)の5状態をとる。Uは、システム内にそのメモリブロックのコピーが無いことを示す。Smは、1個以上のノードのキャッシュがそのメモリブロックのコピーを保持し、その内容がメモリと一致していることを示す。Pは、1つのノードのキャッシュがそのメモリブロックのコピーを保持し、その内容がメモリと一致しない可能性があることを示す。BS、BPは、それぞれS、Pへ遷移するための処理を行っている最中であることを示す。BS、BPのメモリブロックへの要求があった場合、それを管理するDSMコントローラはその要求を全て保持し、その状態がそれぞれS、Pへ遷移した後で要求を受けた順に処理する。

3. 投機的コヒーレンス処理とその適用

本章では、SCCMで行われる投機的コヒーレンス処理について述べる。

3.1 投機的コヒーレンス処理

SCCMは以下の5つの投機的コヒーレンス処理を行う。

(1)speculative self-downgrade
(以下、spec.self-dwg.)

Producer-Consumer共有データ(以下、PC共有データ)へのreadアクセスによって発生するコヒーレンスオーバーヘッドを軽減するための処理である。spec.self-dwg.は、EまたはMの状態のコピーをすでに保持するノードが自分のキャッシュに対して発行し、そのコピーの状態をScに遷移させる。その後、コピーの状態がScに遷移したことをホームに通知するためにメッセージ(SPEC_SELF_DWG)を送信する。また保持しているコピーがMの状態の場合は、その内容をホームに書き戻す。

(2)supeculative self-invalidation
(以下、spec.self-inv.)

PC共有データへのwriteアクセスおよびMigratory共

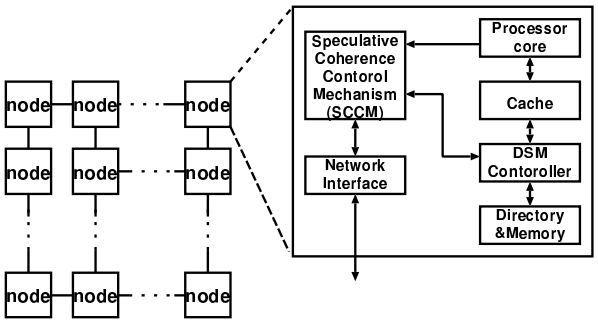

図1 DSMシステムとノード内の構成

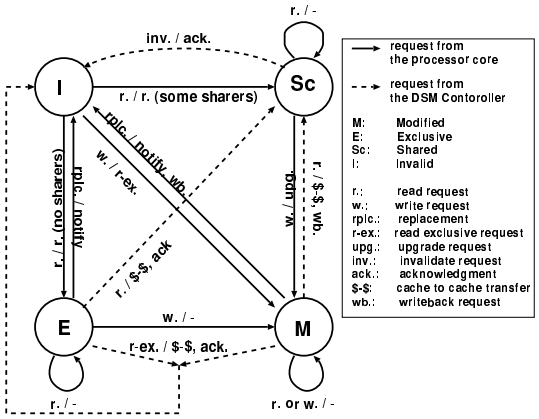

図2 キャッシュラインの状態遷移

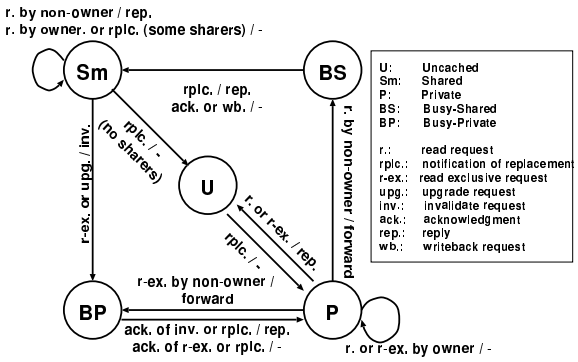

図3 メモリブロックの状態遷移

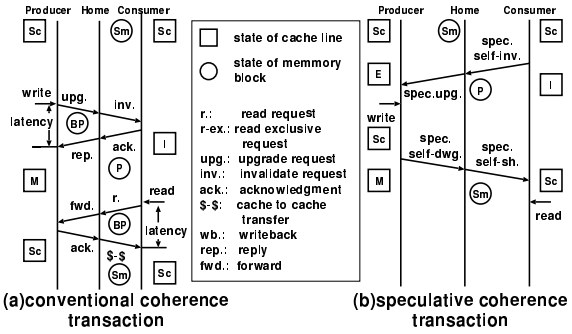

図 4　PC 共有状態における投機的コヒーレンス処理の例

有データ (M 共有データ) への read と write 両方のメモリアクセスによって発生するコヒーレンスオーバーヘッドを軽減するための処理である。spec.self-inv. は、コピーをすでに保持するノードが自分のキャッシュに対して発行し、そのコピーを無効化する。その後コピーを無効化したことをホームに通知するためにメッセージ (SPEC_SELF_INV) を送信する。また、保持しているコピーの状態が M のときは、その内容をホームに書き戻す。

(3)supeculative send block in shared state(以下、spec.send-sh.)

PC 共有データへの read アクセスによって発生するコヒーレンスオーバーヘッドを軽減するための処理である。spec.send-sh. はホームにおいて Sm の状態のメモリブロックに対して read 要求を発行するノードを予測し、そのノードに対して Sc の状態のコピーを保持したメッセージ (REPLY SPEC_SEND_SH) を送信する。それと同時に、予測されたノードが Sc の状態のコピーを保持していることを DSM コントローラに対して通知する必要がある。

(4)speculative send block in exclusive state(以下、spec.send-ex.)

M 共有データへの read および write アクセスによって発生するコヒーレンスオーバーヘッドを軽減するための処理である。spec.send-ex. は、ホームにおいて U の状態のメモリブロックに対して read 要求に引き続く read-exclusive 要求を発行するノードを予測し、そのノードに対して E 状態のコピーを保持したメッセージ (REPLY SPEC_SEND_EX) を送信する。それと同時に、予測されたノードがコピーを専有し、その内容がメモリブロックと一致しない可能性があることを DSM コントローラに対して通知する必要がある。

(5)speculative upgrade (以下、spec.upg.)

PC 共有データおよび M 共有データへの write アクセスによって発生するコヒーレンスオーバーヘッドを軽減するための処理である。spec.upg. は、ホームにおいて Sm の状態のメモリブロックに対して upgrade 要求を発行するノードを Sc の状態のコピーを持つノードの中から予測し、そのノードに対して E の状態への遷移を許可したことを通知するためにメッセージ (REPLY SPEC_UPG) を送信する。それと同時に, 予測されたノードがコピーを専有し, その内容がメモリブロックと一致しない可能性があることを DSM コントローラに対して通知する。

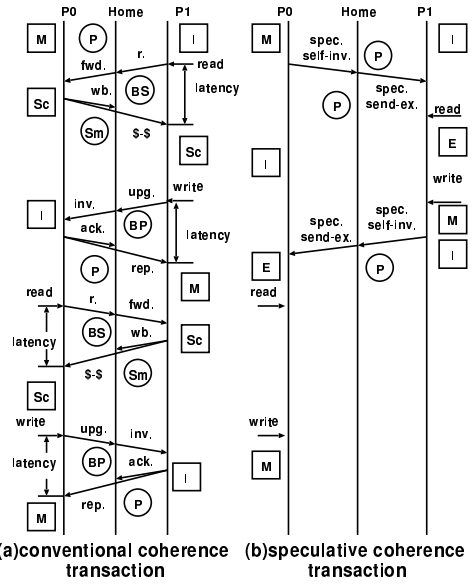

図 5　M 共有状態における投機的コヒーレンス処理の例

3.2　PC 共有状態における処理

図 4は、Producer-Consumer 共有状態 (以下、PC 共有状態) におけるコヒーレンス処理の流れを示したものである。図 4(a) は PC 共有データへのメモリアクセスを投機を行わないコヒーレンス処理で解決する場合の例である。図 4(b)SCCM による投機的コヒーレンス処理によってコヒーレンスオーバーヘッドを完全に除去できたときの例である。

図 4(a) では、Producer の Sc 状態のデータへの write アクセスによって、Home に upgrade 要求 (REQ UPGRADE) が送信される。次に Home から Sc 状態のデータのコピーを持っている Consumer へ invalidate 要求 (COHE INVL) が送信される。Consumer はコピーを無効化し、その旨を Home に通知するためのメッセージ (COHE_REPLY INVL) を送信する。その後、Home は upgrade 要求を許可するためのメッセージ (REPLY UPGRADE) を Producer に送信する。次に Consumer の M 状態のデータへの read アクセスによって、Home に read 要求 (REQ READ_SH) が送信される。次に Home は Producer へ最新のデータを要求するためのメッセージ (COHE COPYBACK) を送信する。Producer は Home へ最新のデータ (COHE_REPLY COPYBACK) を書き戻す。Home は Consumer に要求されたデータ (REPLY SH) を送信する。

図 4(b) では、spec.self-inv. によって Sc 状態のデータのコピーを持っている Consumer が予めコピーを無効化し、その後コピーを無効化したことを Home に通知するためのメッセージ (SPEC_SELF_INV) を送信する。Home は spec.upg. によって Sm 状態のメモリブ

ロックに対してupgrade要求を発行するノードをSc状態のコピーを持つノードの中から予測し、そのノード(Producer)に対してE状態への遷移を許可したことを通知するためのメッセージ(REPLY_SPEC_UPG)を送信する。その後Procucerからのwriteアクセスが発生しているので、リモートレイテンシが削減されている。次に、今度はEまたはM状態のコピーをすでに持っているProducerがspec.self-dwg.によってコピーの状態をScに遷移させる。その後、コピーの状態がScに遷移したことをHomeに通知するためのメッセージ(SPEC_SELF_DWG)を送信する。そして、HomeはConsumerに予めSc状態のコピー(REPLY SPEC_SEND_SH)を送信する。その後、Consumerからのreadアクセスが発生しているので、リモートレイテンシが削減されている。

3.3 M共有状態における処理

図5は、Migratory共有状態(以下、M共有状態)におけるコヒーレンス処理の流れを示したものである。図5(a)はM共有データへのメモリアクセスを投機を行わないコヒーレンス処理で解決する場合の例である。図5(b)はSCCMによる投機的コヒーレンス処理によってコヒーレンスオーバーヘッドを完全に除去できたときの例である。

図5(a)では、P1のM状態のデータへのreadアクセスによってP1からHomeへread要求(REQ READ)が送信される。次にHomeからP0へ最新のデータを要求するためのメッセージ(COHE COPYBACK)が送信される。その後、P0はHomeに最新のデータを書き戻すためのメッセージ(COHE COPYBACK)を送信し、同時にP0からP1に要求されたデータを持つメッセージ(REPLY SH)が送信される。次に、P1からSc状態のデータへwriteアクセスが発生すると、P1からHomeへupgrade要求メッセージ(REQ UPGRADE)が送信される。その後、HomeからSc状態のデータのコピーを持っているP0にinvalidate要求メッセージ(COHE INVL)が送信される。P0はコピーを無効化し、その後、Homeへ無効化したことを通知するためのメッセージ(COHE_REPLY INVL)を送信する。それと同時にP1へ要求されたデータを持つメッセージ(REPLY EXCLDY)を送信する。以下、P0のM状態のデータへのreadアクセスが発生した場合も同様に処理される。

図5(b)では、P0がspec.self-inv.によって自分が保持しているSc状態のデータのコピーを無効化する。そして、コピーを無効化したことをHomeに通知するためのメッセージ(SPEC_SELF_INV)を送信する。HomeはU状態のメモリブロックに対してread-exclusive要求を発行するノードを予測し、そのノード(P1)に対してE状態のコピーを持つメッセージ(REPLY SPEC_SELF_EX)を送信する。その後Producerからのreadおよびwriteアクセス発生しているので、リモートレイテンシが削減されている。次にP1がspec.self-inv.によって自分の保持しているSc状態のデータのコピーを予め無効化し、その後コピーを無効化したことをHomeに通知するためのメッセージ(SPEC_SELF_INV)を送信する。次にHomeはU状態のメモリブロックに対してread-exclusive要求を発行するノードを予測し、そのノード(P0)に対してE状態のコピー(REPLY SELF_SEND_EX)を送信する。その後P0からのreadおよびwriteアクセスが発生しているので、リモートレイテンシが削減されている。

3.4 投機が失敗した場合の処理

投機が失敗した場合の処理を以下に示す。spec.self-inv.およびspec.self-dwg.では、投機が失敗した場合、本来必要ないRequest、Replyメッセージが流れる。

spec.send-sh.、spec.send-ex.およびspec.upg.では、投機が失敗した場合、投機が失敗したことをディレクトリコントローラに通知するためのメッセージ(COHE_REPLY SPEC_SEND_EX、COHE_REPLY SPEC_UPG)を送信する。

4. 評価

本章では、投機を行わない従来のコヒーレンス制御機構(以下、non-spec)を適用したDSMシステム、readアクセスのみを投機の対象とする投機的コヒーレンス制御機構(以下、spec-read)を適用したDSMシステム、readとwriteアクセスの両方を投機の対象とするSCCMを適用したDSMシステムの相互結合網のメッセージトラフィックがどのようになっているかを実験により明らかにする。

4.1 評価方法

評価は、表1に示すDSMシステムのモデル上でSPLASH-2のPC共有データモデルのアプリケーションであるBarnesとM共有データモデルのアプリケーションであるRaytraceをCC-NUMA型並列計算機シミュレータRSIM[6]でシミュレーションすることにより行った。RSIMは実験のためにSCCMとSpec-readの機構をのせ、メッセージの情報を計測できるように改良したものを使用した。Barnes、Raytraceそれぞれのシミュレーションで、表2に示すメッセージの受信回数、メッセージの受信回数×平均メッセージサイズ、平均レイテンシ、平均ブロック時間を計測した。以降はメッセージの表記は表中括弧内のの略称を使用する。ここで、メッセージの受信回数×平均メッセージサイズとは、ネットワークを流れたメッセージの合計のサイズであり、平均レイテンシとは、送信されたメッセージが送信元のノードのNIを出てから送信先のノードのNIに到着するまでにかかったサイクル数の平均であり、平均ブロック時間とは、1つのメッセージが送信元から送信されて送信先に届くまでにネットワークの混雑のためにネットワークを使えず、使えるまで待機していたサイクル数の平均である。

4.2 評価結果

図6、図7にBarnesとRaytraceのそれぞれのメッセージ受信回数、図8、図9に受信回数×平均メッセージサイズ、図10、図11に平均レイテンシ、図12、図13に平均ブロック時間を示す。それぞれの図のグラフの並び方は、図6と同じ順序で並んでいる。

この結果から、Barnesにおいてメッセージ受信回

表1 シミュレーションパラメータ

ノード数			16
プロセッサコア			4-way スーパースカラ
命令キャッシュ			完全ヒット
データキャッシュ(ノンブロッキング)			
ラインサイズ			64バイト
最大未解決ミス数			4
サイズ	L1		64Kバイト
	L2		8Mバイト
アクセス	L1		1サイクル
レイテンシ	L2	タグ	3サイクル
		データ	4サイクル
メモリアクセスレイテンシ			
ローカルメモリ			80サイクル
ネットワーク			
トポロジ			2D-メッシュ
メッセージヘッダサイズ			16バイト
フロー制御方法			ワームホール
レイテンシ /1hop	データなし (2hops以降)		128サイクル (+48サイクル/1hop)
	データ付き (2hops以降)		300サイクル (+48サイクル/1hop)

表2 評価メッセージ一覧

非投機メッセージ
REQ READ_SH(R.R.S.)
REQ READ_EX (R.R.E.)
REQ UPGRADE(R.U)
REPLY SH(R.S.)
REPLY EXCL(R.E,)
REPLY UPGRADE(R.U.)
REPLY EXCLDY(R.ED.)
COHE COPYBACK(C.C.)
COHE COPYBACK_INV(C.C.I.)
COHE INVL(C.I.)
COHE_REPLY COPYBACK(C.R.C.)
COHE_REPLY COPYBACK_INV(C.R.C.I.)
COHE_REPLY INVL(C.R.I.)
COHE_REPLY WRB(C.R.W.)
COHE_REPLY REPL(C.R.R.)
投機メッセージ
SPEC_SELF_DWG(S.S.D)
SPEC_SELF_INV(S.S.I.)
REPLY SPEC_SEND_SH(R.S.S.S)
REPLY SPEC_SEND_EX(R.S.S.E)
REPLY SPEC_UPG(R.S.U.)
COHE_REPLY SPEC_SEND_EX(C.R.S.S.E)
COHE_REPLY SPEC_SEND_UPG(C.R.S.S.U)

数ではSCCMでは全体で0.40%減少している。特にspec-read、SCCMともにR.S.S.S.が発生していることからR.R.S.とR.S.が減少している。またSCCMではS.S.I.が発生していることから、C.I.とC.R.Iが減少している。このことから投機によってメッセージの受信回数を減少させることがわかる。また、これにともないメッセージ受信回数×平均メッセージサイズもSCCMでは全体で1.13%減少している。特にR.R.S.とR.S.が減少している。しかし、平均レイテンシはSCCMでは全体で34%増加している。特にR.S、E.ED.、C.R.CとC.R.C.I.が増加しており、平均ブロック回数は、SCCMでは全体で1.54%増加している。特にR.S.、R.ED.、C.I.、C.R.C.、C.R.C.I、C.R.I.が増加している。このことからネットワークトラフィックは削減されていないことがわかる。次にRaytraceにおいて、メッセージ受信回数ではSCCMでは全体で0.97%増加している。特にspec-read、SCCM ともにR.R.S.、R.S.、R.E.、C.C、C.R.C.が増加している。これにともない、メッセージ受信回数×平均メッセージサイズは、SCCMでは全体では0.65%増加している。特にR.S.、R.E.が増加している。平均レイテンシは、SCCMでは全体で1.39%増加しているが、R.S.、R.E.、R.ED.、C.R.I.が減少しており、平均ブロック時間では、全体で7.35%減少しており。特にR.S.、R.E.、R.E.D.、C.R.C.I.、C.R.I.が減少している。このことからネットワークトラフィックは削減されていることがわかる。

今回使用したBarnesとRaytraceでは投機メッセージの受信回数が少ないことから、投機をほとんど必要としなかったことが考えられる。そのためネットワークトラフィックがあまり削減されなかったのではないかと考える。

5. ま と め

本論文では、SPLASH-2を用いて投機的コヒーレンス制御機構のメッセージトラフィックについて評価した。その結果、メッセージ受信回数は最大で0.40%、メッセージ受信回数×平均メッセージサイズは最大で1.13%、平均レイテンシは最大で1.39%、平均ブロック時間は最大で7.35%減少することがわかった。

今後は、SPLASH-2の他のアプリケーションを用いて評価を行う予定である。

謝辞 本研究は、一部日本学術振興会科学研究費補助金(若手研究(B)14780186、及び基盤研究(B)14380135、(C)14580362)の援助による。

参 考 文 献

1) Keleher, P. J.; Distributed Shared Memory Home Pages. http://www.cs.umd.edu/ keleher/dsm.html.
2) 細見岳生、加納健、中村真章、広瀬哲也、中田登志之; 並列計算機Cenju-4の分散共有メモリ機構、並列処理シンポジウムJSPP'99論文集、pp. 15-22(1999).
3) Landon, J. and Lenoski, D.; The SGI Origin: A cc-NUMA Highly Scalable Server, *Proceedings of the 24th Annual International Symposium on Conputer Architecture* (1997).
4) 古川文人、多田野陽介、乗貞由華、大津金光、馬場敬信; DSMシステムにおける投機的コヒーレンス制御機構の提案と評価、情報処理学会研究報告2001-ARC-142 2001-HPC-85、pp. 169-

174(2001).
5) Woo, S.C., Ohara, M., Torrie, E., Singh, J. P. and Gupta, A.; The SPLASH-2 Programs; Characterizarion and Methodological Considerations, *Proceedings of the 22th Annual International Symposium on Computer Architecture*, pp. 179-190(1998).
6) Pai, V. S., Ranganathan, P. and Adve, S. V.; RSIM; An Execution-Driven Simulator for ILP-Based Shared-Memory Multiprocessors and Uniprocessors, *Proceedings of the Third Workshop on Conputer Architecture Education*(1997).

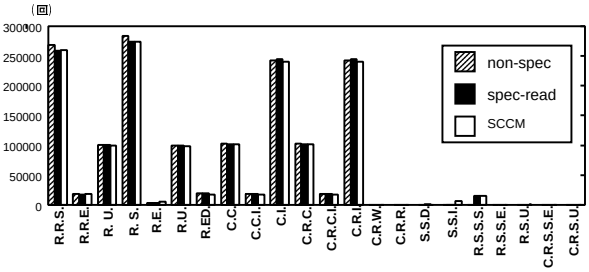

図6 Barnesのメッセージ受信回数

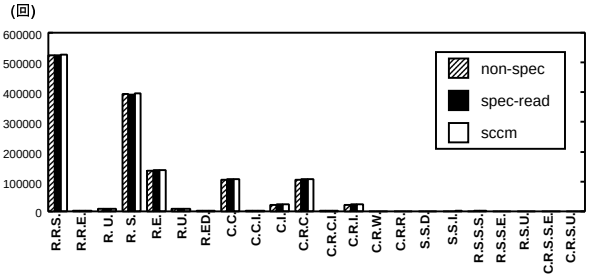

図7 Raytraceのメッセージ受信回数

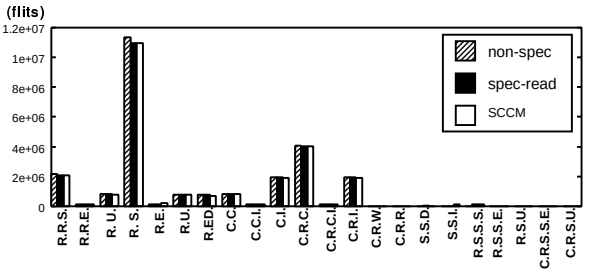

図8 Barnesのメッセージ受信回数×平均メッセージサイズ

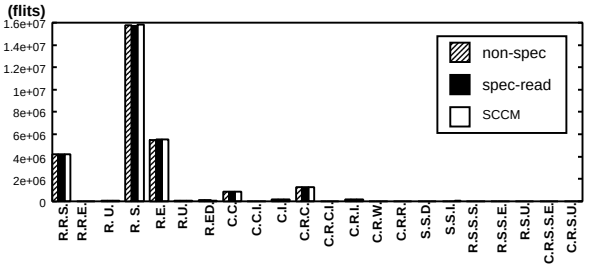

図9 Raytraceのメッセージ受信回数×平均メッセージサイズ

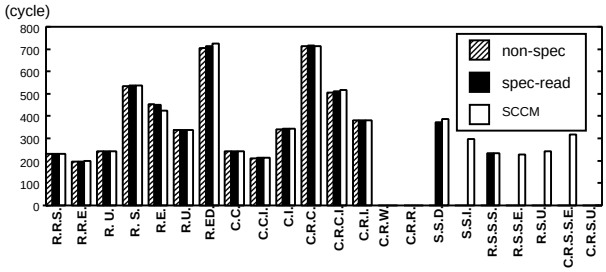

図10 Barnesの平均レイテンシ

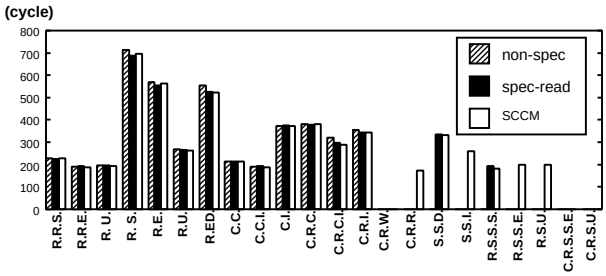

図11 Raytraceの平均レイテンシ

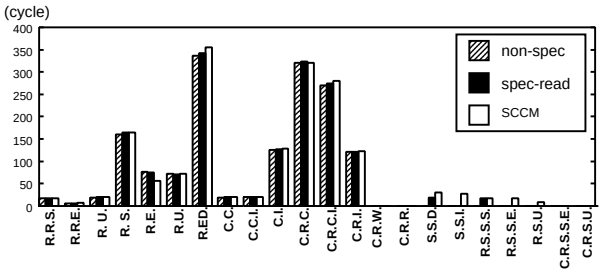

図12 Barnesの平均ブロック時間

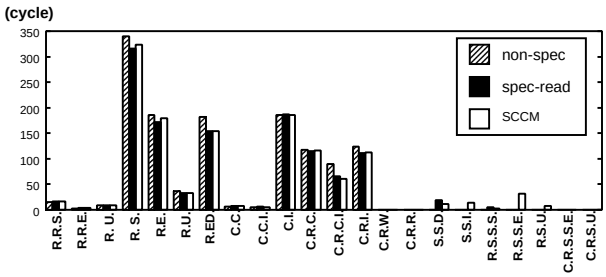

図13 Raytraceの平均ブロック時間