計算機アーキテクチャ　99－17
(1993. 3. 12)

ＡＳＵＲＡの解析モデル

城 和貴

(株) クボタ
コンピュータ事業推進室
大阪市浪速区敷津東１丁目２番４７号

Abstract

分散共有メモリ型マルチプロセッサＡＳＵＲＡに対するセミ・マルコフ過程を利用した解析モデルを提案する。提案されるモデルはコヒーレンス制御やネットワーク競合による待ち状態を容易に記述できるため、大規模で複雑なアーキテクチャを持つ並列計算機の性能評価に利用できる。モデル化の目的は、ＡＳＵＲＡのキャッシュ・ヒット率及びデータのリード／ライト比が、キャッシュ・コヒーレンス制御を含むリクエストや、それに対する待ち状態に、どのような影響を与えるかを解明し、性能評価を与えることである。さらに、構築されたモデルを用いて、プロセッサ利用率、通常データ／コヒーレンス制御リクエスト及びそれらの待ち時間について実際の評価を行なった。

セミ・マルコフ過程、並列計算機、解析モデル、性能評価

An Analytic Model for the ASURA

Kazuki JOE

Office of Computer Business, KUBOTA Corporation
1-2-47, Shikitsuhigashi, Naniwaku, Osaka, Japan
E-mail: joe@kocb.astem.or.jp

Abstract

This paper presents a discrete time model of memory and network interference for the cache coherence mechanism in the ASURA, which is a caching cluster-based distributed shared memory multiprocessor computer. It differs from earlier models in its ability to model variable waiting time both for normal requests and cache coherent requests. Actually, the aim of our model is to analyze how the ratio of cache hit and read/write requests can affect the performance from the view point of normal/coherent requests and their waiting states. Furthermore, we evaluate the performance, IO request rates, processor utilizaiion etc., of the ASURA using this analytic model.

SemiMarkov Processing, Parallel Computer,
Analytic Model, Performance Evaluation

1 はじめに

プロセッサ・クラスタ方式の並列計算機は将来のスーパーコンピューティングへの有力な一方式として、研究開発が推進されてきた。特に共有メモリを持つものは[8]並列コンパイラ開発の観点からも望ましいものである。さらにキャッシュ制御方式の研究[1]に伴い、キャッシュを持ったクラスタ方式の共有メモリ型並列計算機が精力的に研究されている[7][5]。いずれもネットワークの性能を補うものとしてキャッシュを有効利用しており、スケーラブルなクラスタ型共有メモリ方式の並列計算機という新しい方向に向かっている。

このような大規模な並列計算機アーキテクチャの妥当性を確かめるのに、シミュレーションは有効な手法である反面、数万数十万規模のプロセッサの動きを検証するのに要するコストは新たな問題になるであろう。

一方、並列計算機の性能評価を行なうのに、確率モデルや確率過程を用いた解析モデルは低コストで有効な手法である。しかしながら、これまでに提案されてきた並列計算機システムの解析モデル[4][6][10][2][3][9]では、キャッシュを持ったアーキテクチャは対象にされておらず、また、リクエストに対するネットワーク競合等の待ち状態を記述しているモデルも少ない。

我々は既に大規模な並列処理計算機への第一歩として、クラスタ・ベースの階層型マルチプロセッサ・システムＡＳＵＲＡを提案し[15]、ＡＳＵＲＡ全体を確率モデルで表した性能評価[13]、ＡＳＵＲＡクラスタに着目した解析モデル[14]について報告している。

そこで本稿では、マルコフ連鎖の各状態で任意の時間滞在出来るセミ・マルコフ過程[11]を階層的に構成し、クラスタ型並列計算機に適用することに着目し、ＡＳＵＲＡ全体を表現した解析モデルを提案する。

この解析モデルの目的は、ネットワークで結合されたキャッシュを持つクラスタ型分散共有メモリ方式の並列計算機、ＡＳＵＲＡに対する、定性的な解析とその評価である。

本稿では、ＡＳＵＲＡ全体のモデル化のための仮定を述べ、モデル化のための状態記述の定義を行ない、具体的な計算手順を示した後、性能評価を行なう。

2 準備

2.1 ＡＳＵＲＡの定義

性能評価のモデル化のために、ＡＳＵＲＡを次のように定義する。P個のプロセッサ（プライベート・キャッシュ（ＰＣＨ）を持つ）と１つの[1]ローカル・メモリ（ＬＭ）、ネットワーク・インターフェイス（ＮＩＦ）がバス結合され、プロセッサ・クラスタ（ＰＣ）を形成する。ただし、ＮＩＦはグローバル・メモリ（ＧＭ）とグローバル・キャッシュ（ＧＣＨ）、ＰＣ間の通信を制御するコントローラからなる。（ＰＣ間の通信はＰＣ内のバスに影響を与えないことに注意）M個のＰＣはリング結合されてクラスタ・グループ（ＣＧ）を構成し、K個のＣＧがクロスバ結合することによりＡＳＵＲＡ全体が定義される。従って、全体では$P \times M \times K$個のプロセッサとＰＣＨ、それぞれ$M \times K$個のローカル・メモリ・モジュール、グローバル・メモリ・モジュール、ＧＣＨ、$M \times K$本のバス、K個のリング、１個のクロスバによって並列計算機システムを構成することになる。メモリ階層性を示すために、メモリ・ロケーションをローカル・メモリ（ＬＭ）、ＰＣ内グローバル・メモリ（ＰＣＧＭ）、ＣＧ内グローバル・メモリ（ＣＧＧＭ）、リモート・グローバル・メモリ（ＲＧＭ）で区別する。なお、ＰＣＨ、ＧＣＨのキャッシュ・コヒーレンス・プロトコルはそれぞれ、イリノイ型の変形[12]、Synapse[15]に従う。

2.2 セミ・マルコフ過程

セミ・マルコフ過程（以後ＳＭＰと呼ぶ）に関する詳細は文献[11]に譲るとして、ここではＳＭＰについての簡単な説明を行う。

ＳＭＰとはK個の状態からなる確率過程である。状態iにおいては平均η_iの間滞在し、状態jにp_{ij}の確率で遷移する[2]。もしＳＭＰがエルゴード的な既約なエンベデッド・マルコフ連鎖（以後ＥＭＣと呼ぶ）を持つなら、状態iの定常分布P_iは次式で表される。

$$P_i = \frac{\pi_i \eta_i}{\sum_{j=1}^{K} \pi_j \eta_j} \quad (1)$$

ただし、$\{\pi_i\}$はＥＭＣの定常分布である。本稿におけるモデルでは、後に述べるようにエルゴード的な成分が１つだけからなるＥＭＣで表されているので、式1が利用できる。また、ＳＭＰの状態iから脱出する確率λ_iは次のようにして求められる。

$$\lambda_i = \frac{P_i}{\eta_i} = \frac{\pi_i}{\sum_{j=1}^{K} \pi_j \eta_j} \quad (2)$$

2.3 モデルの仮定

本稿ではＡＳＵＲＡのキャッシュ・コヒーレンス制御を考慮に入れた定性的評価を行なうために、プロセッサおよびＰＣの状態遷移をマルコフ連鎖によってそれぞれ表し、これらをＥＭＣと見なすことにより、ＳＭＰを定義し、任意のリクエストの待ち時間を考慮した解析モデルを構築する。モデルを簡略化するために、次のような仮定を導入する。

1. 全てのプロセッサからのリクエストが全て独立であるような並列プログラムが稼働中である。言い換えれば特殊な同期操作は本モデルでは記述出来ないことになる。
2. プロセッサとＰＣの状態は、計算（アイドル）状態、ネットワークを握ってリクエストを実行している状態、その状態への待ち状態。
3. プロセッサ内オン・チップ・キャッシュにヒットした状態は計算状態と解釈。
4. ＰＣＨにヒットした状態は、バスを使わないリクエストと解釈し、同時に計算も行なう。
5. 他のプロセッサの状態に影響を与えるようなリクエストは、その作用が自分自身に影響を与えるものとする。

[1] 実際にはインターリーブされているが、バスとメモリのサイクル比が同じと考えた方がモデルが簡単であるためこのような仮定を用いる

[2] 各状態の滞在時間がすべて１である時、そのＳＭＰは通常のマルコフ連鎖と等価なことに注意されたい。

6. プログラムは安定稼働している状態、すなわち、起動時のページングやキャッシュ・ヒット率の悪さは考慮しない状態を仮定する。またキャッシュは既に使い切った状態で、それに伴うリプレースについては考慮する。
7. ファイルＩ／Ｏ等によるバスへのリクエストは考慮しない。

3 モデル化

3.1 モデル化の概要

ＡＳＵＲＡは階層的なネットワークおよび複雑なメモリ・システムを持つため、全体を一つのモデルで表すことは困難である。本稿ではモデルの簡略化のため、まず、ＰＣ内バスの影響を直接受けるプロセッサとＰＣＨを一つのモデル（以後Ｐモデルと呼ぶ）として表し、ＰＣ間ネットワークに直接影響を受けるＮＩＦ（ＧＣＨを含む）を別のモデル（以後Ｇモデルと呼ぶ）として表す。

Ｐモデルは [14] で構築したＡＳＵＲＡクラスタのモデルの自然な拡張となる。つまり、ＡＳＵＲＡクラスタのモデルにＰＣ間通信のための状態を追加することで、Ｐモデルが形成される。また、Ｇモデルにおけるプロセッサ間通信のメモリ階層に応じたリクエスト率は、[13] によるものとする。

ＰモデルにおいてＧＣＨおよび共有メモリに対してなされるリクエスト（通常のデータ・リクエストとＭＬＩ保証のためのＧＣＨ制御のリクエスト）はＰモデルからＧモデルへの出力として、また、Ｇモデルにおいて ＭＬＩ保証のためになされるＰＣＨへの制御はＧモデルからＰモデルへの出力として定義される。

ＰモデルおよびＧモデルを使ってＡＳＵＲＡ全体を表すためには、まずＰモデルに対する適当な初期値（このときＰＣ間通信はないものとする）を与え、Ｇモデルへの出力を得る。次に得られたＧモデルへの入力と適当な初期値によって、Ｇモデルからプロセッサモデルへの出力を得る。以後、Ｐモデルおよびモデルが安定状態に達するまでこの操作を繰り返すことにより、ＡＳＵＲＡ全体の定常状態を知ることができる。

3.2 Ｐモデル

3.2.1 状態の定義

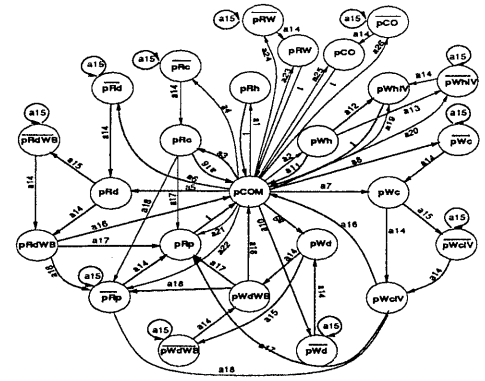
Ｐモデルのモデル化を行なうために、状態定義を行なう。ＰモデルはＡＳＵＲＡクラスタのモデルにクラスタ間通信のための状態を追加したものであるため、[14] に、グローバル・データに対するＰＣＨのミス・ヒットに起因するリクエスト（この場合、ＧＣＨもしくは共有メモリが参照される。）およびその待ち状態、他のＰＣへのコヒーレント制御のためのリクエストおよびその待ち状態、の４状態を追加することとする。

$pCOM$	計算状態、もしくはプロセッサ内キャッシュのアクセス
pRh	ＰＣＨからの読み込み
pWh	ＰＣＨへの書き込み
$pWhIV$	pWh により発生するインバリデーション、もしくはＭＬＩ保証のためＧＣＨから要求されるインバリデーション
$\overline{pWhIV}$	$pWhIV$ への待ち状態
pRc	そのライン属性が $Dirty$ ではないローカル・データのキャッシュ・ミスによる読み込み
$\overline{pRc}$	pRc への待ち状態
pRd	そのライン属性が $Dirty$ であるローカル・データのキャッシュ・ミスによる読み込み
$\overline{pRd}$	pRd への待ち状態
$pRdWB$	$DirtypRd$ により発生するライトバック
$\overline{pRdWB}$	$pRdWB$ への待ち状態
pWc	そのライン属性が $Dirty$ ではないローカル・データのキャッシュ・ミスによる書き込み
$\overline{pWc}$	pWc への待ち状態
$pWcIV$	pWc により発生するインバリデーション
$\overline{pWcIV}$	$pWcIV$ への待ち状態
pWd	そのライン属性が $Dirty$ であるローカル・データのキャッシュ・ミスによる書き込み
$\overline{pWd}$	pWd への待ち状態
$pWdWB$	pWd により発生するライトバック
$\overline{pWdWB}$	$pWdWB$ への待ち状態
pRP	ラインのリプレースにより発生するライトバック、もしくはＭＬＩ保証のためグローバル・キャッシュから要求されるライトバック
$\overline{pRP}$	pRP への待ち状態
pRW	グローバル・データのＰＣＨミス・ヒットによる読み書き
$\overline{pRW}$	pRW への待ち状態
pCO	他のＰＣへのコヒーレント制御命令
$\overline{pCO}$	pCO への待ち状態

図 1は定義された状態間の遷移を示す。図 1において、HはＰＣＨのヒット率、Rはプロセッサからの通常のリクエストに対するリード・リクエストの割合（従って、$1-R$ はライト・リクエストを意味することになる。プロセッサのオペレーションに対するリクエスト率ではないことに注意。）、Dはリクエスト要求のあるデータを含むラインが $Dirty$ である確率、Cはリクエスト要求のあるデータを含むラインが自分以外の少なくとも一つのキャッシュで $Valid$ である確率、Wは（バスに対して）リクエストを出した時に待ち状態に陥る確率、Xはキャッシュが既に一杯である確率を意味する。この時、プロセッサの各サイクル毎のバスに対するリクエスト率φが必要となる。ここで言うバスに対するリクエストとは、ＰＣＨにヒットしないリクエストを意味する。また、L は通常のデータ・リクエストのアドレスがローカル・メモリにある確率、Vは他のＰＣにコヒーレント制御のためのリクエストを要求する確率を表す。図 1からも明らかなように、このＳＭＰをＥＭＣと見るとこれはエルゴード的であるので、式 1を使って定常分布を求めることが出来る。

ＥＭＣの遷移確率行列は図 1のように表される。

先に述べたＥＭＣを用いてＳＭＰを形成するには、各状態の平均滞在時間 $\{\eta_i\}$ が必要である。まずＡＳＵＲＡクラスタの仕様から定まるものとしては、表 1がある。ただし、各値はプロセッサのサイクル数を表す。状態ＣＯＭでの滞在時間は、先に述べたように通常の計算時間に加えて、プロセッサ内キャッシュに対するアクセス時間も含めたもの



$a1 = (1-Y)(1-V)HR$, $a2 = (1-Y)(1-V)H(1-R)$, $a3 = (1-Y)(1-V)(1-H)LR(1-D)(1-W)$,
$a4 = (1-Y)(1-V)(1-H)LR(1-D)W$, $a5 = (1-Y)(1-V)(1-H)LRD(1-W)$, $a6 = (1-Y)(1-V)(1-H)LRDW$,
$a7 = (1-Y)(1-V)(1-H)L(1-R)(1-D)(1-W)$, $a8 = (1-Y)(1-V)(1-H)L(1-R)(1-D)W$,
$a9 = (1-Y)(1-V)(1-H)L(1-R)D(1-W)$, $a10 = (1-Y)(1-V)(1-H)L(1-R)DW$, $a11 = 1-C$, $a12 = C(1-W)$,
$a13 = CW$, $a14 = 1-W$, $a15 = W$, $a16 = 1-XD$, $a17 = XD(1-W)$, $a18 = XDW$, $a19 = Y(1-Z)(1-W)$,
$a20 = Y(1-Z)W$, $a21 = YZ(1-W)$, $a22 = YZW$, $a23 = (1-Y)(1-V)(1-H)(1-L)(1-W)$,
$a24 = (1-Y)(1-V)(1-H)(1-L)W$, $a25 = (1-Y)V(1-W)$, $a26 = (1-Y)VW$

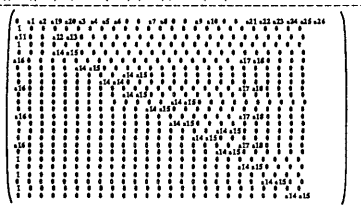


図 1: Pモデルの状態遷移図とそのEMCの遷移確率行列

とするので、20サイクル程度が妥当かと思われる。一方、待ち状態に関してはこのような明確な値は出てこない。なぜなら、通常のデータ・リクエストに加えてキャッシュ・コヒーレンス制御のためのリクエストも各待ち状態に影響を与えるからである。あるいは、PC外部へのリクエストもGモデルの状態によってその遅延時間が動的に変化する。従って、各待ち状態の滞在時間（WTとする）と外部へのリクエスト時間（GRW, CO）は適当な初期値を与え、以後バスへのアクセスを行なう各状態の確率をもとに繰り返し計算することによって収束させることとする[3]。

pRh	pWh	$pWhIV$	pRc	pRd	$pRdWB$
10	10	33	121	163	33
pWc	$pWcIV$	pWd	$pWdWB$	pRP	$pCOM$
121	33	163	33	33	20

表 1: 各状態の平均滞在時間

SMPの定常分布は、EMCの定常分布$\{\pi_i\}$と滞在時間$\{\eta_i\}$を使って式1から求められる。EMCの定常分布$\{\pi_i\}$は

$$\begin{pmatrix} \frac{1-W}{HR(1-V)(1-W)(1-Y)} \\ H(1-R)(1-V)(1-W)(1-Y) \\ (1-W)(CH(1-R)(1-V)(1-Y)+Y(1-Z)) \\ W(CH(1-R)(1-V)(1-Y)+Y(1-Z)) \\ (1-D)(1-H)LR(1-V)(1-W)(1-Y) \\ (1-D)(1-H)LR(1-V)W(1-Y) \\ D(1-H)LR(1-V)(1-W)(1-Y) \\ D(1-H)LR(1-V)W(1-Y) \\ D(1-H)LR(1-V)(1-W)(1-Y) \\ D(1-H)LR(1-V)W(1-Y) \\ (1-D)(1-H)L(1-R)(1-V)(1-W)(1-Y) \\ (1-D)(1-H)L(1-R)(1-V)W(1-Y) \\ (1-D)(1-H)L(1-R)(1-V)(1-W)(1-Y) \\ (1-D)(1-H)L(1-R)(1-V)W(1-Y) \\ D(1-H)L(1-R)(1-V)(1-W)(1-Y) \\ D(1-H)L(1-R)(1-V)W(1-Y) \\ D(1-H)L(1-R)(1-V)(1-W)(1-Y) \\ D(1-H)L(1-R)(1-V)W(1-Y) \\ (1-W)(D(1-H)L(1-V)X(1-Y)+YZ) \\ W(D(1-H)L(1-V)X(1-Y)+YZ) \\ (1-H)(1-L)(1-V)(1-W)(1-Y) \\ (1-H)(1-L)(1-V)W(1-Y) \\ V(1-W)(1-Y) \\ VW(1-Y) \end{pmatrix}$$

$$\frac{}{2-w+(1-V)(1-V)[CH(1-R)-HW+(1-H)L\{1-R+D(R+X)\}]} \quad (3)$$

で求められる。また、各状態からの脱出確率は式2で求められる。

3.2.2 Pモデル中の変数定義

このようにして定められたモデルを評価するには、モデルに対する適当なパラメータが必要である。Pモデルでは静的なパラメータとして、PCHのヒット率H、データ・リクエストに対するリードの割合R、ローカル・メモリに対するリクエストの割合L、全てのデータに対するリード／ライト・データの割合RW_ratio[4]を考えることとする。また、動的なパラメータとしては、PC外部にコヒーレント制御命令を要求する確率V、他のPCからコヒーレント制御要求が来る確率Y、その時の要求がライトバックである確率Zがあるが、これらは適当な初期値を与え、以後Gモデルとの相互作用で収束させる。Pモデル内部だけで使う動的なパラメータとしては、プロセッサのリクエスト率φがある。また、PモデルでのSMPの状態iの定常分布をP_i、脱出確率をλ_i、平均滞在時間をη_iと表すことにする。

まず、先に上げたパラメータとPC内のプロセッサ数Pを使って、変数D, C, X[5], W, V, WTを表す。これらは、[14]より、

$$D = (P-1)H(1-R)(1-H(1-R))^{P-2} \quad (4)$$

$$C = 1-(1-HRRW\_ratio)^{P-1} \quad (5)$$

$$x = (1-(HC(1-R)\varphi+(1-H)(1-D)(1-R)\varphi))^{P-1} \quad (6)$$

LMおよびGMへのリクエストを出す状態の集合をQ_{LM}, Q_{GM}とすると、

$$Q_{LM} = \{pWcIV, pRc, pRd, pRdWB, pWc, pWcIV, pWd, pWdWB, pRp\} \quad (7)$$

$$Q_{GM} = \{pRW, pCO\} \quad (8)$$

あるプロセッサの状態が$i \in Q_{LM} \cup Q_{GM}$の時、次のサイクルでも引続き状態を変えない確率は$P_i - \lambda_i$であるから、バスが空いていない確率を$BUSY$と

[3] Pモデルでのリクエストはプライオリティを仮定しない。すなわち、各待ち状態の平均滞在時間は同じと仮定する。

[4] データをリード・オンリー、ライト・オンリー、リード／ライトと分けた場合、キャッシュ・コヒーレンス制御に関係するのは同期変数等のリード／ライト・データである。

[5] 本モデルではキャッシュを使い切った状態を考えているが、キャッシュを使い切っていない状態は$X=0$とすることにより容易に記述出来る。

すると、

$$BUSY = (P-1)\{L\sum_{i\in Q_{LM}}(P_i-\lambda_i)+\frac{(1-L)\sum_{i\in Q_{LM}}(P_i-\lambda_i)}{MK}\} \quad (9)$$

他のプロセッサからのリクエストとの競合に負ける確率は、

$$1-\frac{1-(1-\varphi)^P}{P\varphi} \quad (10)$$

よって、待ち状態に陥る確率Wは

$$W = BUSY + (1-BUSY)(1-\frac{1-(1-\varphi)^P}{P\varphi}) \quad (11)$$

サイクル毎のプロセッサのリクエスト率φは次のようにして求める。ある状態から抜け出した時にリクエストを発行する状態は、$pCOM$および、待ち状態である。前者はキャッシュ・ミスの場合にリクエストが発行される。よって、式2を使って次のようにして求まる。

$$\begin{aligned}\varphi &= (1-H)\lambda_{pCOM} + L(\lambda_{\overline{pWcIV}} + \lambda_{\overline{pRc}} \\ &+ \lambda_{\overline{pRd}} + \lambda_{\overline{pRdWB}} + \lambda_{\overline{pWc}} + \lambda_{\overline{pWcIV}} \\ &+ \lambda_{\overline{pWd}} + \lambda_{\overline{pWdWB}} + \lambda_{\overline{pRp}}) \\ &+ (1-L)(\lambda_{\overline{pRW}} + \lambda_{\overline{pCO}}) \end{aligned} \quad (12)$$

WTは待ち状態にいる時、バスが次に空くまでの平均時間であるので、

$$WT = \sum_{i\in Q_{LM}\cup Q_{GM}} P_i\eta_i \quad (13)$$

によって求まる。

3.3 Gモデル

Pモデルではプロセッサの状態推移をPCHの動きに注目してモデル化を行なったが、Gモデルではプロセッサの代わりにNIFの状態推移をGCHの動きに注目してモデル化を行なうこととする。従って、Pモデルでの計算状態の代わりに、Gモデルではリクエストがない状態、アイドル状態、を状態推移の中心に考える。

3.3.1 状態の定義

Gモデルのモデル化を行なうために、状態定義を行なう。以下はNIFのGCH、共有メモリ、PC間ネットワークに対する各状態を表している。

$gIDLE$	アイドル状態
gRh	GCHからの読み込み
gWh	GCHの書き込み
$\overline{gWhIV}$	gWhにより発生するインバリデーション
$\overline{gWhIV}$	$gWhIV$への待ち状態
gRc	そのライン属性が$Dirty$ではないグローバル・データのGCHミス・ヒットによる読み込み
$\overline{gRc}$	gRcへの待ち状態
gRd	そのライン属性が$Dirty$であるグローバル・データのGCHミス・ヒットによる読み込み
$\overline{gRd}$	gRdへの待ち状態
gWc	そのライン属性が$Dirty$ではないグローバル・データのGCHミス・ヒットによる書き込み
$\overline{gWc}$	gWcへの待ち状態
$gWcIV$	gWcにより発生するインバリデーション、もしくはMLI保証のため、PCから要求されるインバリデーション
$\overline{gWcIV}$	$gWcIV$への待ち状態
gWd	そのライン属性が$Dirty$であるグローバル・データのGCHミス・ヒットによる書き込み
$\overline{gWd}$	gWdへの待ち状態
gWB	ダーティ・ラインへの読み書きにより発生するライトバック
$\overline{gWB}$	gWBへの待ち状態
gRP	ラインのリプレースにより発生するライトバック
$\overline{gRP}$	gRPへの待ち状態

図2は定義された状態間の遷移を示す。図2において、hはGCHのヒット率、rは通常のリクエストに対するリード・リクエストの割合、dはリクエスト要求のあるデータを含むラインが$Dirty$である確率、cはリクエスト要求のあるデータを含むラインが自分以外の少なくとも一つのキャッシュで$Valid$である確率、wは（PC間ネットワークに対して）リクエストを出した時に待ち状態に陥る確率、xはキャッシュが既に一杯である確率、uはPCからコヒーレント制御命令が要求される確率を意味する。この時、各サイクル毎のNIFに対するリクエスト率ψが必要となる。このリクエストはPモデルでの状態pRWから発行される。あるプロセッサからNIFに対して発行されるリクエストは、[13]で考察したように、リクエスト先がPCGM、CGGM、RGMであるので、それらの確率をそれぞれS_{PCGM}、S_{CGGM}、S_{RGM}とする。これら以外のリクエストは、LMに対してなされるため、次式が成り立つ。

$$L + S_{PCGM} + S_{CGGM} + S_{RGM} = 1 \quad (14)$$

図2からも明らかなように、このSMPをEMCと見るとこれはエルゴード的であるので、式1を使って定常分布を求めることが出来る。また、その推移確率行列は図2のように表される。

Gモデルでの平均滞在時間はPモデルと違って一意に定まらないものが多い。これは、Gモデルの場合はリクエスト先が同一場所ではないため、アクセス時間が異なるからである。

まず、滞在時間が一意に定まるものとしては、ネットワークを使わないグローバル・キャッシュのアクセスがある。η_{gRh}, η_{gWh}の値としては、ローカル・メモリのアクセス時間と同じ（121サイクル）とする。次に、ネットワークを使うリクエストが発行された場合、平均滞在時間はメモリのアクセス時間をΓサイクル、PC間ネットワークの遅延時間をΔ_iサイクル（$i\in\{PCGM, CGGM, RGM\}$）とすると、

$$\frac{\sum_{i\in\{PCGM,CGGM,RGM\}} S_i\Delta_i}{1-L} + \Gamma \quad (15)$$

となる。例えば$\Delta_{PCGM} = 5, \Delta_{CGGM} = 10, \Delta_{RGM} = 50$、メモリ・アクセスの場合、$\Lambda = 256$、インバリデーションの場合、$\Lambda = 20$、ライトバックの場合、$\Lambda = 20$、などにする。これらのリクエスト状態に対する待ち時間はPモデル同様、適当な初期値を与えたwtに対して繰り返し計算することにより収束させる。最後に、アイドル状態

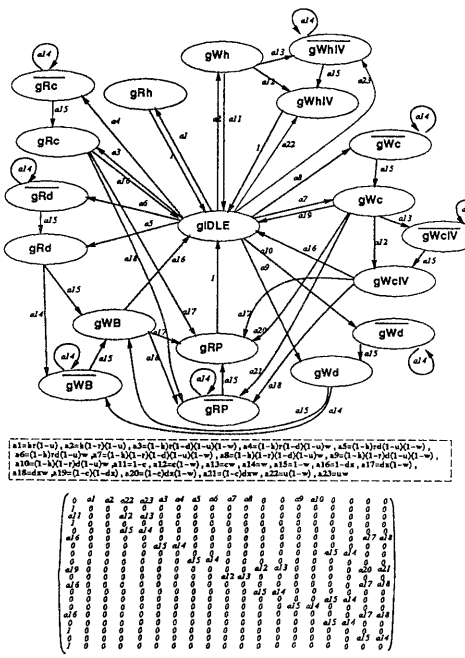


図 2: Gモデルの状態遷移図とそのEMCの遷移確率行列

の平均滞在時間η_{gIDLE}は、Pモデルでの計算状態の平均滞在時間に対応するものであるが、これはPモデルのP_{RW}によって変化する。

SMPの定常分布は、EMCの定常分布$\{\pi_i\}$と先に定義した滞在時間$\{\eta_i\}$を使って式1から求められる。EMCの定常分布$\{\pi_i\}$は次でもとめられる。また、各状態からの脱出確率は式 2で求められる。

$$\frac{\begin{pmatrix} \frac{1-w}{hr(1-w)(1-w)} \\ h(1-r)(1-w)(1-w) \\ (ch(1-r)(1-w)+w)(1-w) \\ \{ch(1-r)(1-w)+w\}w \\ (1-d)(1-h)r(1-w)(1-w) \\ \{(1-d)(1-h)r(1-w)w \\ d(1-h)r(1-w)(1-w) \\ d(1-h)r(1-w)w \\ (1-d)(1-h)(1-r)(1-w)(1-w) \\ (1-d)(1-h)(1-r)(1-w)w \\ c(1-d)(1-h)(1-r)(1-w)(1-w) \\ c(1-d)(1-h)(1-r)(1-w)w \\ d(1-h)(1-r)(1-w)(1-w) \\ d(1-h)(1-r)(1-w)w \\ d(1-h)(1-w)(1-w) \\ d(1-h)(1-w)w \\ d(1-h)(1-w)(1-w)x \\ d(1-h)(1-w)wx \end{pmatrix}}{2+w+(1-w)\{c(1-d+dh)(1-r)+d(1-h)(1+x)-hw\}} \quad (16)$$

3.3.2 Gモデル中の変数定義

Pモデル同様、Gモデルでは静的なパラメータとして、GCHのヒット率h、データ・リクエストに対するリードの割合r、メモリ階層に対応するリクエストの割合$S_{PCGM}, S_{CGGM}, S_{RGM}$、を考えることとする。動的なパラメータとしてはPモデル同様d, c, x, w, wt、下位のPCから共有データへのリクエストが来る確率ψがある。また、Gモデルでの SMPの状態iの定常分布をP_i、脱出確率をλ_i、平均滞在時間をη_iと表すことにする。

先に上げたパラメータとを使って、変数d, c, x, w, wtを表す。これはPモデルの時と同様にして、

$$d = (MK-1)h(1-r)(1-h(1-r))^{MK-1} \quad (17)$$

$$c = 1-(1-hRRW\_ratio)^{MK-2} \quad (18)$$

$$x = (1-(\psi h(1-r)(1-(1-c)^{MK-1})))^{MK-1} \quad (19)$$

PC間ネットワーク対してリクエストを出した時、待ち状態に陥る確率はPモデルと同様、ネットワークの状態とアービトレーションによって定義される。まず、ネットワークにリクエストを出す状態の集合をΩとする。

$$\Omega = \{gWcIV, gRc, gRd, gRdWB, gWc, gWcIV, gWd, gWdWB, gRp\} \quad (20)$$

あるプロセッサの状態が$i \in \Omega$の時、リクエスト先がPCGMの場合は待ち状態に陥ることはない。CGGMの場合、リング・ネットワークが空いていない確率をB_Cとすると、

$$B_C = \frac{1}{M}\sum_{i\in\Omega}(P_i - \lambda_i) \quad (21)$$

ネットワーク内での競合に負ける確率$Lost_C$は、

$$1-\frac{1-(1-\psi)^M}{M\psi} \quad (22)$$

リクエスト先がRGMの場合、クロスバ・ネットワークが空いていない確率をB_Rとすると、

$$B_R = \frac{K-1}{K}\sum_{i\in\Omega}(P_i - \lambda_i) \quad (23)$$

ネットワーク内での競合に負ける確率$Lost_R$は、

$$1-\frac{1-(1-\frac{\psi}{K})^K}{K\psi} \quad (24)$$

で求まる。よって、待ち状態に陥る確率wは

$$\begin{aligned} w = & \frac{S_{CGGM}}{1-L}(B_C + (1-B_C)Lost_C) \\ & + \frac{S_{RGM}}{1-L}(B_C + (1-B_C)B_R \\ & + (1-B_C)(1-B_R)Lost_R) \end{aligned} \quad (25)$$

PCからのリクエスト率ψは、次のようにして求める。まず、待ち状態を考えない場合の初期値としてのψは、$(1-H)(1-L)\varphi$の確率で各PCから発行されるリクエストであるから、

$$1-\{1-(1-H)(1-L)\varphi\}^P \quad (26)$$

となる。以後はPモデルと同様、待ち状態を考慮したリクエスト率ψの修正を次のようにおこなう。

$$\begin{aligned} \overline{\Omega} = & \{\overline{gWcIV}, \overline{gRc}, \overline{gRd}, \overline{gRdWB}, \\ & \overline{gWc}, \ \overline{gWcIV}, \overline{gWd}, \overline{gWdWB}, \overline{gRp}\} \end{aligned} \quad (27)$$

$$\psi = (1-h)\lambda_{gIDLE} + \sum_{j\in\overline{\Omega}}\lambda_j \quad (28)$$

wtは待ち状態にいる時、ネットワークが次に空くまでの平均時間であるので、

$$wt = \sum_{i\in\Omega} P_i\eta_i \quad (29)$$

によって求まる。

3.4 PモデルとGモデルのI／F

MLI属性の保証をするためのPモデルとGモデルとの入出力に関わる変数V, Y, Z, u および、

その相互作用に影響をうける滞在時間η_{pRW}, η_{pCO}, η_{gIDLE}の定義を行なう。

まず、PモデルからGモデルへの出力変数を定義する。

他のPCにコヒーレント制御のためのリクエストを要求する確率Vは以下のようにして求める。PCHにおいて、ライト・ヒットもしくはライト・ミスが起こった場合、そのラインを含むGCHのラインが*Valid*である時、他のGCHに対するインバリデーション要求が必要となる。よって、

$$V = (P_{pWcIV} + P_{pWhIV}) * c \quad (30)$$

Gモデルにおいて、下位のPCからコヒーレント制御命令を要求される確率uは、Vを用いて、

$$u = 1 - (1 - V)^P \quad (31)$$

によって表される。

次に、GモデルからPモデルへの出力変数を定義する。Gモデルにおいてインバリデーションやライトバックが発生した場合、そのラインの一部をキャッシングしているPCHがあると、MLI属性を保証するためには、そのPCHにもコヒーレント制御命令を送らなければならない。

GCHで*Dirty*なライン、もしくはGCHで*Valid*なラインでも、そのラインをさらにキャッシングしているPCHが*Dirty*であった場合、インバリデーション要求が起きると、下位のPCHにもインバリデーション要求を起こさなければならない。PCHで*Dirty*の場合は、インバリデーションの変わりに、ライトバック要求がなされる。ライトバック要求に対しても同様である。

GモデルからMLI属性保証のために、あるPCHにインバリデーションを流す確率は、

$$\frac{1}{P}\{(P_{gWhIV} + P_{gWcIV})(c + dC) + P_{gRP}(c + dC)\} \quad (32)$$

同じく、リプレース要求を出す確率は、

$$\frac{1}{P}\{(P_{gWhIV} + P_{gWcIV} + P_{gRP})dD\} \quad (33)$$

よって、他のPCからコヒーレント制御要求が来る確率Yは、式 32と式 33を足し合わせたものになる。また、その時の要求がライトバックである確率Zは、Yに対する式 33の割合である。

最後に、残された滞在時間の定義を行なう。Pモデルでの状態、GRW は、Gモデルでの通常のデータ・アクセスに要する時間であるから、次のように定義できる。

$$\begin{aligned} GRW = {} & hR\eta_{gRh} + \\ & h(1-R)\{\eta_{gWh} + c(\eta_{gWhIV} + w\eta_{\overline{gWhIV}})\} + \\ & hR(1-d)\{\eta_{gRc} + w\eta_{\overline{gRc}} + dx(\eta_{gRP} + w\eta_{\overline{gRP}})\} + \\ & hRd\{\eta_{gRd} + w\eta_{\overline{gRd}} + \eta_{gWB} + w\eta_{\overline{gWB}} + \\ & \quad dx(\eta_{gRP} + w\eta_{\overline{gRP}})\} \\ & h(1-R)(1-d)\{\eta_{gWc} + w\eta_{\overline{gWc}} + \\ & \quad c(\eta_{gWcIV} + w\eta_{\overline{gWcIV}}) + dx(\eta_{gRP} + w\eta_{\overline{gRP}})\} \\ & h(1-R)d\{\eta_{gWd} + w\eta_{\overline{gWd}} + \eta_{gWB} + w\eta_{\overline{gWB}} + \\ & \quad dx(\eta_{gRP} + w\eta_{\overline{gRP}})\} \end{aligned} \quad (34)$$

同様にして、COは次式で求められる。

$$CO = \eta_{gWhIV} + w\eta_{\overline{gWhIV}} \quad (35)$$

Gモデルでの$gIDLE$の滞在時間は、リクエスト率からも求まるであろうが、ここでは次のようにして求めた。

$$\eta_{gIDLE} = \frac{P_{gIDLE}\sum \pi_i\eta_i}{(1 - P_{gIDLE})\pi_{gIDLE}} \quad (36)$$

3.5 モデルの計算手順

このようにして定義したモデルを実際に計算する手順は以下の通りである。

1. パラメータとして、PCH、GCHのヒット率、リード・リクエストの割合を与える。
2. サイクル毎のプロセッサのリクエスト率とPモデルおよびGモデルでの平均待ち時間に適当な初期値を与える。
3. PモデルのEMCの定常分布$\{\pi_i\}$を求める
4. Pモデルの平均滞在時間$\{\eta_i\}$を求める
5. 各状態からの脱出確率$\{\lambda_i\}$を求める
6. 動的な変数（W,X,C,D等）のアップデート。
7. 前回のリクエスト率と新しく求めたリクエスト率の差が適当な値よりも大きければ3に戻る
8. GモデルのEMCの定常分布$\{\pi_i\}$を求める
9. Gモデルの平均滞在時間$\{\eta_i\}$を求める
10. 各状態からの脱出確率$\{\lambda_i\}$を求める
11. 動的な変数（w,x,c,d等）のアップデート。
12. GモデルからPモデルへの入力パラメータ（Y,Z,V）の計算。
13. 前回のリクエスト率と新しく求めたリクエスト率の差が適当な値よりも大きければ8に戻る
14. 以上の操作を適当な回数繰り返す

4 評価

SMPを用いたASURAのモデルに対する評価対象として、プロセッサ利用率、通常及びコヒーレント制御のリクエスト時間の割合、それらの待ち時間の割合等を考える。これらの計算手順については文献[14]を参照されたい。

ASURAの評価に対するパラメータとして、$r = 0.7, 0.7 < h < 1.0$としたものの結果をグラフ 3、4に示す。ただし、他のパラメータは、$L = 0.4$、$S_{PCGM} = 0.3$、$S_{CGGM} = 0.2$、$S_{RGM} = 0.1$、$RW\_Ratio = 0.1$ である。また、PCHとGCHのヒット率、リードの割合は同じ値を使用した。

図 3では、各プロセッサが、PCにおいてキャッシュ・ヒット率とどのような関係を持つかを示している。プロセッサの利用率からすると、キャッシュ・ヒット率が90%以上なければ、さほど性能が発揮されず、主にメモリ・リクエストとコヒーレンス制御に時間を費やされることがわかる。

図 4では、各PCが、ASURA全体におけるキャッシュ・ヒット率とどのような関係を持つかを示している。PCは絶対数が多いため、キャッシュ・ヒット率が高くなってもコヒーレンス制御命令が多発するため、アイドル時間がほとんど増えないことがわかる。

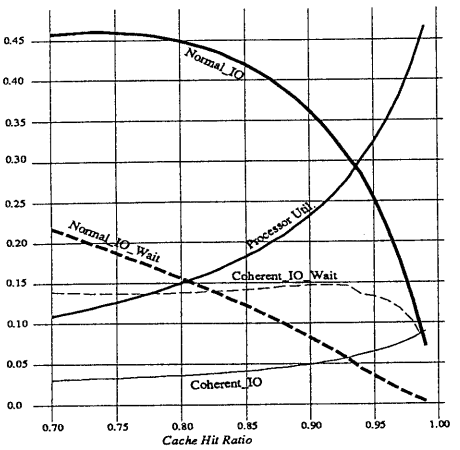


図 3: P-Model Performance under $R = 0.7, 0.7 < H < 1.0$

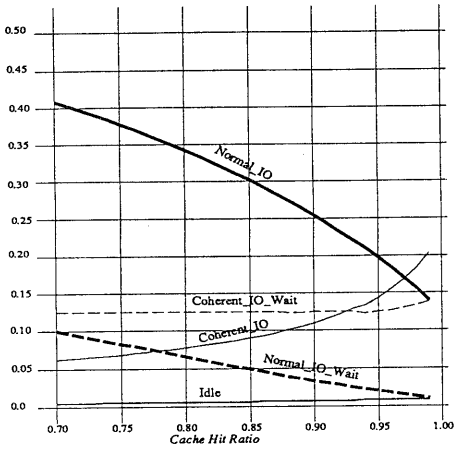


図 4: G-Model Performance under $R = 0.7, 0.7 < H < 1.0$

5 結論

分散共有メモリ型並列計算機ＡＳＵＲＡに対する解析モデルを提案し、それを実際に計算した結果、クラスタ内ではクラスタ内ネットワークの負荷を軽くするために高キャッシュ・ヒット率が、クラスタ間ではコヒーレンス制御命令を減らすことが、全体の性能向上に必要であることが判明した。

謝辞

日頃ご討論頂く京都大学工学部富田教授並びに同研究室の諸氏に感謝致します。また、研究の機会を与えて頂いた（株）クボタ山口部長並びに名古屋大学工学部阿草教授に感謝いたします。最後に、ここまでプロジェクトを推進してこれたのはＡＳＵＲＡプロジェクト・チーム諸氏の努力の賜であり、諸氏に敬意を表します。

参考文献

[1] A.Agarwal, Richard Simoni, John Hennessy, and Mark Horowitz. An evaluation of directory schemes for cache coherence. In *Proceedings of the International Symposium on Computer Architecture*, pages 280–289, 1988.

[2] Santosh G. Abraham and Edward S. Davidson. A communication model for optimizing hierarchical multiprocessor systems. In *Proceedings of the 1986 International Conference on Parallel Processing*, pages 467–474, 1986.

[3] Dharma P. Agrawal and Imadeldin O. Mahgoub. Performance analysis of cluster-based supersystems. In *Proceedings of the International Conference on Supercomputer System*, pages 593–602, 1985.

[4] Laxmi N. Bhuyan. A combinatorial analysis of multibus multiprocessors. In *Proceedings of the 1984 International Conference on Parallel Processing*, pages 225–227, 1984.

[5] David R. Cheriton and Hendrik A.Goosen. Paradigm: A highly scalable shared-memory multicomputer architecture. *IEEE Computer*, pages 33–46, 1991.

[6] Chita R. Das and Laxmi N. Bhuyan. Bandwidth availability of multiple-bus multiprocessors. *IEEE Transactions on Computers*, 34(10):918–926, 1985.

[7] Daniel E.Lenoski. The design and analysis of dash: A scalable directory-based multiprocessor. Technical Report CSL-TR-92-507, Stanford University, CSL, 1992.

[8] David J. Kuck, Edward S. Davidson, Duncan H. Lawrie, and Ahmed H. Sameh. Parallel supercomputing today and the cedar approach. *Science*, 231(2):967–974, 1986.

[9] Imadeldin O. Mahgoub and A. K. Elmagarmid. Performance analysis of a generalized class of m-level hierarchical multiprocessor systems. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):129–138, 1992.

[10] T. N. Mudge, J. P. Hayes, and D. C. Winsor. Multiple bus architectures. *IEEE Computer Magagine*, 20(6):42–48, 1987.

[11] S.M. Ross. *Applied Probability Models with OptimizationApplications.* Holden-Day, 1970.

[12] 内藤 潤, 城 和貴, 松野 宏昭, and 新田 博之. ＡＳＵＲＡクラスタの性能評価. Technical Report 92-ARC-97-10, 情報処理学会計算機アーキテクチャ研究会, 1992.

[13] 城 和貴, 柳原 守, David Fraser, 田中 高士, 新田 博之, 森 眞一郎, 齋藤 秀樹, and 富田 眞治. 分散共有メモリ型マルチプロセッサ「ＡＳＵＲＡ」の階層性とその評価. Technical Report 92-ARC-95-1, 情報処理学会, 1992.

[14] 城 和貴 and 内藤 潤. セミ・マルコフ過程を用いたＡＳＵＲＡクラスタのモデル化. Technical Report 92-ARC-97-9, 情報処理学会, 1992.

[15] 森 眞一郎, 齋藤 秀樹, 五島 正裕, 富田 眞治, 田中 高士, David Fraser, 城 和貴, and 新田 博之. 分散共有メモリ型マルチプロセッサ「阿修羅」の概要. Technical Report 92-ARC-94-6, 情報処理学会, 1992.