オペレーティング・システム 42 － 6
（１９８９．２．２４）

マルチプロセッサシステムの評価技法と評価システム

松岡　浩司、堀川　隆、難波　信治
日本電気（株）　Ｃ＆Ｃシステム研究所

要求仕様に最も適したマルチプロセッサシステムのアーキテクチャやシステム構成を決めるための性能評価技法の研究と、その評価技法にもとづいた評価システムの開発を進めている。この評価技法の特徴は、実計算機のプログラム実行履歴が記録されたトレースデータを用いてシミュレーションを行い、システムの実効性能を評価する点にある。

このような評価技法を導入する、本評価システムは、設計の初期段階でシステムの実効性能の評価を行い、さらに、評価に要する時間を短縮することにより、最も適したシステム構成を選択するための比較検討を効率良く行うことを目指している。

An Evaluation Method and an Evaluation System for Multi-processor Systems

Hiroshi Matsuoka, Takashi Horikawa, Shinji Nanba
C&C Systems Reserch Laboratories, NEC Corporation
4-1-1, Miyazaki, Miyamae-ku, Kawasaki, 213, Japan

We propose a performance evaluation method for multi-processor systems and a performance evaluation system using the method. The method is applied to architecture and system configuration designs of multi-processor systems in early stage. It is characterized by using the trace data in which user program and system program behaviors are recorded. By using the method, the evaluation system evaluates the effective performance. A model for simulation is described using a simplified method to decreases the time needed for an evaluation. It is used to increase the efficiency in system designs to select the best architecture and system configuration form proposed ones.

1．はじめに

コンピュータシステムの性能が向上すると、より大規模な問題が扱えるようになるため、さらに高い性能が要求されるようになる。この要求に対し、最新デバイス技術の導入によるシステムの高速化が図られている。ところが、プロセッサに見合った動作速度を持つメモリの実現が難しいといった問題点があるため、システムのマルチプロセッサ化が検討されている。

マルチプロセッサシステムの設計では、初期の段階で設計するシステムに要求される性能に最も適したアーキテクチャやシステム構成を決める。このために、検討の対象となっている各システムの性能の概略を把握することが特に重要である。ところが、マルチプロセッサシステムでは複数のプロセッサを並列に動作させるために複雑な制御が行われており、性能を予測評価することが難しいものとなっている。このため、設計の初期段階において性能を予測評価できるツールを要求する声が大きい。

著者らは、要求仕様に最も適したマルチプロセッサシステムのアーキテクチャやシステム構成を決めるための性能評価技法の研究と、その評価技法にもとづいた評価システムの開発を進めている[1]。この評価技法の特徴は、実計算機のプログラム実行履歴が記録されたトレースデータを用いて、システムの実効性能を評価する点にある。

本稿では、マルチプロセッサシステムの性能を評価するための課題と技法を論じ、さらに、その評価技法を適用した性能評価システムについて述べる。

2．システム設計と性能評価

本評価システムは、共有バスに接続された複数のマイクロプロセッサが主記憶を共有する密結合マルチプロセッサシステムを評価対象としている。ここでは、このようなシステムにおける現状の設計方法を分析し、設計の初期段階で行われる性能評価について論じる。

2．1　現状のシステム設計とその問題点

システムの設計は、大きく、アーキテクチャ設計、機能設計、詳細論理設計の３つのステップに分けることができる。まず、アーキテクチャ設計では、要求仕様を検討し要求仕様に最も適したアーキテクチャを決め、さらに、機能の分割を行いシステム構成を決める。次に、機能設計では、分割された個々の機能についての検討を行う。さらに、詳細論理設計では、機能設計で検討された個々の機能をハードウェアとして実現するために、回路の論理設計を行う。

このように設計し作製したシステムが設計段階で期待した性能を発揮しないといった性能問題を起こさないようにするため、詳細論理設計の結果として得られたＲＴレベルの回路記述を組み合わせて、論理設計の検証と同時に性能の予測評価が行われている。ところが、この段階で性能問題が明らかになっても、機能レベル、悪くすれば、アーキテクチャレベルの変更が必要となる。この変更に伴い、詳細論理設計をやり直す必要が生じ、システムの設計に要する時間が長くなってしまうという問題が生じる。

2．2　設計初期段階における性能予測評価

上記の問題を回避するためには、採用したアーキテクチャやシステム構成が適切であることを、アーキテクチャ設計や機能設計の段階で検証する必要がある。この検証は、従来、表２－１に示すような性能予測評価によって行われていた。
表２－１に示すように、ハードウェアを強く意識するシステム設計では、シミュレーションによる性能予測評価が適しているといえる。

このようなシミュレーションによる性能予測評価を行うためのツールは、大きく、次の２つに分類できる。

（１）回路設計支援ツール
（２）アーキテクチャシミュレータ

回路設計支援ツールは、基本的には、ハードウェアを作製するためのＣＡＤツールである。しかし、米国国防省の超高速ＩＣプロジェクトで開発されたハードウェア記述言語VHDL[2]を使うツールなどでは、動作時間を設定してシステムの動作を記述することにより、性能の評価を行うことを目

性能評価方法	特徴	問題点
①机上検討による性能予測評価	・簡便	・熟練と経験を要する。 ・長い時間の動作を検討できない。
②ｼﾐｭﾚｰｼｮﾝによる性能予測評価	・長い時間の動作を検討できる。 ・ﾊｰﾄﾞｳｪｱを記述する。	・ｿﾌﾄｳｪｱの動きを評価に反映させることが難しい。
③解析的評価（待ち行列など）	・ｼｽﾃﾑ性能の評価に適している。	・ﾊｰﾄﾞｳｪｱの違いを評価に反映させることが難しい。

表２－１　機能設計段階における性能予測評価方法

指している。

アーキテクチャシミュレータは、評価対象のアーキテクチャの有効性を簡便に評価するためのツールである。例えば、スタンフォード大のCARE SYSTEM[3]では、システムの動作を目で見ながらシミュレーションを行うことができるなど、ハードウェアの作成を意識しない分評価のための機能が充実している。

しかし、これらの性能評価ツールでマルチプロセッサシステムの性能を評価する場合には、ＯＳによるプロセス管理や排他制御などを考慮しないと、性能を高く見積りすぎてしまうという問題が生じる。例えば、マルチプロセッサシステムでは、リソースの排他制御のために、必ずしもすべてのプロセッサがいつも動作しているとは限らないからである。

本評価システムでは、このようなＯＳによるプロセス管理や排他制御などを反映したシミュレーションを行うことにより、システムの実行性能を評価すること目指している。図２－１に各評価ツールの適用範囲を示す。

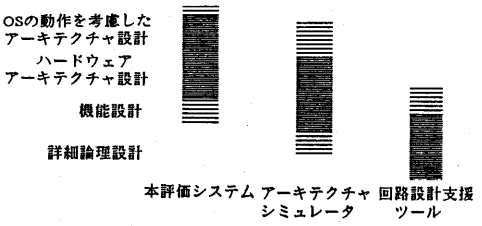


図２－１　性能評価ツールの適用範囲

３．性能評価システムの課題と性能評価技法

アーキテクチャやシステム構成を決定するための性能評価には次の３つの課題がある。

１）システムのモデリング
２）システムの実効性能の評価
３）評価時間の短縮

３．１　システムのモデリング

システムの性能を評価する場合には、まず、評価対象システムのハードウェアを計算機上で扱うことができるように、評価の目的にあった粒度でモデリングする必要がある。ここでは、まず、モデリングの粒度を論じ、次に、本評価システムで採用したシステムのモデルについて説明する。

３．３．１　モデリングの粒度

システムをモデリングする場合には、システムを構成単位に分割していき、構成単位のモデルを組み合わせることによってシステム全体のモデリングを行う。ここでは、この最小の構成単位をモデリングの粒度と呼んでいる。システムをモデリングする場合には、表３－１に示すような粒度のモデリングが考えられる。

表３－１で示すように、アーキテクチャやシステム構成を決めるためのシミュレーションには、ユニットレベルのモデリングが適していると考えている。

３．１．２　システムのモデル

モデリングの粒度をユニットとすると、システムをユニットに分割していき、ユニットのモデルを組み合わせることによってシステム全体のモデリングを行う。

本評価システムでは、システムの構造モデルはユニットとユニットを結ぶリンクから構成されている。ユニットにはポートが定義されていて、リンクはユニットが持つポートとポートを結ぶ。

また、ユニットは内部の詳細な構造や動作を考えないで、ブラックボックスとして扱われ、その機能的動作とその動作に要する時間によって特徴づけられている。例えば、メモリユニットのモデルは、メモリアクセスリクエストによってデータの読み出し動作を開始し、アクセスタイムだけ時間が経過した後、アクセスされたデータを出力するというようにモデル化されている。

プロセッサがメモリに直接接続された最小構成のシステムでは、次のようにシステムの動作がモデル化されている。まず、プロセッサユニットはメモリアクセスを行いメモリアクセスの完了を待つ。メモリユニットはメモリアクセスが行われると、動作を開始し、アクセスタイムだけ時間が経過するとメモリアクセスの完了を通知する。メモリアクセスの完了が通知されると、プロセッサユ

モデルの粒度	構成単位の例	特徴	問題点
①サブシステム	ﾌﾟﾛｾｯｻｻﾌﾞｼｽﾃﾑ I/Oｻﾌﾞｼｽﾃﾑ	・ｼｽﾃﾑ性能の評価に適している。	・ﾊｰﾄﾞｳｪｱの評価が難しい
②ユニット	ﾌﾟﾛｾｯｻﾕﾆｯﾄ ﾒﾓﾘﾕﾆｯﾄ	・ﾊｰﾄﾞｳｪｱとの対応関係が明確(ﾒﾓﾘﾎﾞｰﾄﾞ/ﾒﾓﾘﾕﾆｯﾄ)	・ユニット内部の詳細な動作を評価できない。
③レジスタ	レジスタ 組合せ回路	・詳細な動作を評価できる	・詳細設計まで進まないとモデルが得られない。

表３－１　モデリングの粒度

ニットは次のメモリアクセスを行う。このように1つ1つメモリアクセスを処理しながら、システムの動作が進んでいく。

3．2　実効システム性能の評価

本評価システムで採用した評価技法の特徴は、実計算機のプログラム実行履歴であるトレースデータを使用する点にある。

ここでは、まず、システムの実効性能を評価するための課題をまとめ、次に、トレースデータを分割し組み合わせて評価対象システムにおけるソフトウェア実行をシミュレーションする方式について説明する。さらに、分割したトレースデータを組合せために行うプロセス管理と排他制御のシミュレーションについて説明する。

3．2．1　実効システム性能評価の課題

マルチプロセッサシステムでは、複数のプロセッサを並列に動作させるために複雑な制御が行われている。ソフトウェアの実行には次のような要因が影響を与える。

①ワークロード
②プロセスディスパッチング方式
③例外
④割り込み
⑤ＯＳの排他制御方式
⑥システムプロセスの動き

システムの実効性能を評価するためには、これらの要因の影響を反映したシミュレーションを行わなければならない。

3．2．2　トレース駆動シミュレーション

トレースデータには、アプリケーションだけではなくＯＳの動作も含まれている。このようなトレースデータを分割し、上記の要因に注目しながらこれらを組み合わせて、評価対象システムにおけるソフトウェア実行をシミュレーションする方式を、我々はTrace Driven Simulation (TDsiml) 方式と呼んでいる。

ここでは、まず、シミュレーションに使用するトレースデータの概要を説明し、次に、TDSiml方式について説明する。

3．2．2．1　トレースデータ

トレースデータはソフトウェアを実行するためにプロセッサが行ったメモリアクセスが記録されたメモリアクセストレースである。また、ロジックアナライザを改造したハードウェアトレーサを用いて、実時間で採取されるため[4]、採取時に実行されていた次のようなソフトウェアの実行履歴が記録されている。

①アプリケーション
②システムコール
③例外処理
④割り込み処理
⑤システムプロセス

3．2．2．2　TDSiml方式

TDSiml方式では、3．1．2で述べたシステムのモデルにおけるプロセッサユニットが行うメモリアクセスとして、トレースデータに記録されたメモリアクセス使いシミュレーションを実行する。マルチプロセッサシステムでは、プロセッサ数と同数のソフトウェアを同時に実行できるため、複数の同時に処理できるトレースデータを用意する必要がある。また、トレースデータを採取したシステムと評価対象システムではシステム構成や性能が異なるため、ソフトウェアの実行順序が変化する。例えば、プロセスの実行と非同期割り込みの処理の時間関係が前後する。このため、実行するソフトウェアに対応して使用するトレースデータを切り替える必要がある。

TDSiml方式の概略を図3－1に示す。TDSiml方式では、同時に処理できるトレースデータとして、プロセスのトレースデータを用意する。これは、ＯＳがプロセスを単位として実行管理を行っているためである。また、実行するソフトウェアに対応してトレースデータを切り替えるために、割り込みや例外といったイベントの処理が記録された部分を抜き出し、イベントのトレースデータを用意する。評価対象システムにおけるソフトウェアの実行順序を決めながら、これらのトレースデータを組み合わせてシミュレーションを実行する。

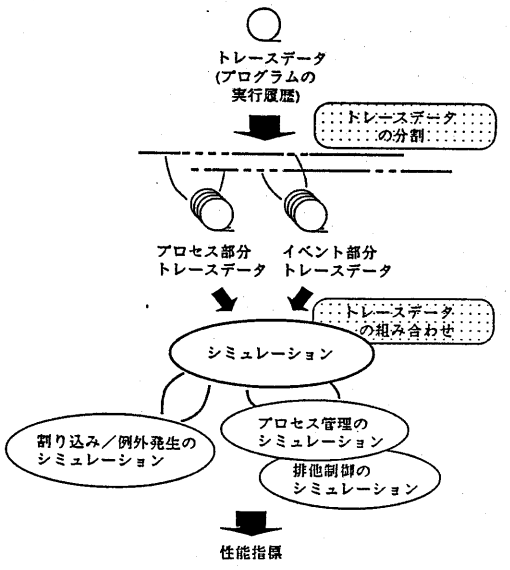


図3－1　TDSiml方式

• トレースデータの分割

トレースデータを分割するためにトレースデータを、まず、MAS(Memory Access Stream)に分割する。イベントに着目して、そのイベントを処理するためにプロセッサが行ったメモリアクセスが記録された部分をイベントMASとして分離する。残りのMASはプロセスを実行するためにプロセッサが行ったメモリアクセスが記録されたプロセスMASである。

次に、これらのMASを識別分類し、プロセスMASをプロセス番号別にプロセス部分トレースデータにまとめ、イベントMASをイベント別にイベント部分トレースデータにまとめる。

• 部分トレースデータによるシミュレーション

システムの実効性能を求めるために、TDSim1方式では、次のようにソフトウェア実行のシミュレーションを行う。

プロセッサユニットは通常プロセスを実行していて、イベントが発生するとイベントの処理に移行する。イベントの処理が完了するとプロセスの処理に復帰する。ここで、まず、いつイベントが発生するかを決めるためのシミュレーションを行う。イベントが発生したと決まった時点で、イベントに対応するMASをイベント部分トレースデータから持ってきて埋め込んだり、プロセス部分トレースデータの切り替えを行う。

3．2．3　プロセス管理のシミュレーション

プロセスディスパッチングによるプロセス実行順序の変化をシミュレーションするために、ＯＳによるプロセス管理のシミュレーションを行う。プロセスがsleepする、あるいは、強制的なディスパッチングが行われるといったイベントの発生をシミュレーションし、ready queueやwait queueなどの管理を行い、次に実行するプロセスを決める。また、プロセスをwake upするイベントの発生をシミュレーションし、プロセスの状態を変更する。

3．2．4　排他制御のシミュレーション

ＯＳの排他制御によるソフトウェア実行の変化をシミュレーションするために、次のようなシミュレーションを行う。カーネルや割り込み処理への移行とカーネルや割り込み処理からの復帰といったイベントの発生をシミュレートし、マルチプロセッサシステムの中でカーネルあるいは割り込み処理を行っているプロセッサが１以下となるように、部分トレースデータの実行を制御する。このような排他制御方式を、我々は、排他カーネルと呼んでいる。

3．3　評価時間の短縮

要求仕様に最も適したアーキテクチャやシステム構成を選択するためには、構成を少しづつ換えながら、いくつかのシステムの評価を繰り返す。このため、１つのシステム構成を評価するために要する時間はできるだけ短いものでなければならない。

性能評価では、まず、評価対象のモデルを記述し、次に、シミュレーションを実行し性能を評価するための指標を求める。したがって、評価に要する時間を短縮するためには、まず、モデルの記述を単純化し、記述に要する時間を短くしなければならない。また、シミュレーション自体を高速化し、シミュレーションに要する時間を短くすることも重要である。

本評価システムでは、シミュレーションを高速化するために、イベントドリブン方式を採用している。ここでは、特に、システム記述を簡素化するために採用したブロック図ベースのシステム記述方式について説明する。

3．3．1　ブロック図ベースシステム記述方式

本評価システムで採用したブロック図ベースのシステム記述であるBlock Function and Timing Diagram (BFTD) は、設計初期の段階で一般的に用いられるシステムの構成を示したブロック図の持つ構造情報に、構成要素としてのユニットの動作に関連する情報を付加したシステム記述である。ユニットの動作に関連する主な情報として、メモリのアクセスタイムなどような時間情報が付加される。また、このBFTD方式では、ユニットに関する設定を単純化するために、標準ユニットを導入している。

ここでは、まず、標準ユニットについて説明し、次に、例を用いてBFTDによるシステム記述について説明する。

3．3．1．1　標準ユニット

個別ユニットに関する設定を単純化するため、まず、マルチプロセッサシステムで使用されると考えられる標準的なユニットを標準ユニットとしてあらかじめ用意する。この標準ユニットの集合を標準ユニットライブラリと呼んでいる。BFTD方式では、個別ユニットの機能に関する設定を行うために、標準ユニットライブラリから個別ユニットと同じ機能を持つ標準ユニットを１つ選んで指定する。例えば、ライトスルー方式のキャッシュとコピーバック方式のキャッシュは異なる標準ユニットとして用意されていて、評価対象システムのキャッシュユニットの制御方式によって１つを選択する。また、動作に関する設定は標準ユニットに定義された次の２種類のユニットパラメータ

を使って行う。

①時間パラメータ

ユニット内部の動作時間を表すパラメータ

例えば、メモリのアクセスタイム

②構成パラメータ

ユニット内部の構成の特徴を表すパラメータ

例えば、キャッシュユニットのセット数

このユニットパラメータを設定することによって、動作時間などが異なるユニットを１つの標準ユニットを使ってモデル化することができる。

最近の回路設計支援ＣＡＤツールでは、記述に要する時間を短縮するために、市販されているＩＣのセルライブラリなどを利用しているが、標準ユニットはこのような記述の簡素化の方向に合致するだけでなく、ユニットパラメータを使うことによって動作時間などの設定を比較的自由に行うことができるという特徴を持っている。

３．３．１．２　BFTDよるシステムの記述例

BFTD は次２つの記述要素から構成されている。

①個別ユニットとユニットリスト

②リンクとリンクリスト

ユニットリストには、システムを構成する個々のユニットの機能と動作時間などが設定される。また、リンクリストには、システムを構成する個別ユニット間の接続関係が記述される。

BFTDによるマルチプロセッサシステムの記述例を図３－２に示す。（ａ）に示した図はBFTDの図的な表現で、実際には（ｂ）のリストのように記述される。

ユニットリストには、次のような書式でマルチプロセッサシステムを構成する個別ユニットを定義する。

個別ﾕﾆｯﾄ名：標準ﾕﾆｯﾄ名(ﾊﾟﾗﾒｰﾀﾘｽﾄ)

この記述例では、６つの個別ユニットが定義されている。これらの個別ユニットを定義するために、４つの標準ユニットが使用されている。例えば、プロセッサの標準ユニットSTDPROCOは最も単純なもので、内部動作時間t_exeのみが定義されている。この標準ユニットを用いたproc0、proc1はメモリアクセスを行い、メモリアクセスの完了が通知されると、t_exeだけ時間経過後次のメモリアクセスを行う。この動作時間はパラメータリストに設定される。例では、内部動作時間は１Ｔに設定されている。また、キャッシュユニットには次のようなパラメータが定義されている。

```
t_acc_hit    ：ヒット時のアクセスタイム
t_dealy_miss：ミスヒット時に、プロセッサからアクセスが行われ
              てからバスアクセスが開始されるまでの遅延時間
t_delay_ret ：ブロックデータが返ってきて、プロセッサにファ
              ーストデータが返されるまでの遅延時間
n_way        ：セットアソシアティブ方式のｗａｙ数
n_set        ：セット数
s_sector     ：セクターサイズ[Byte]
```

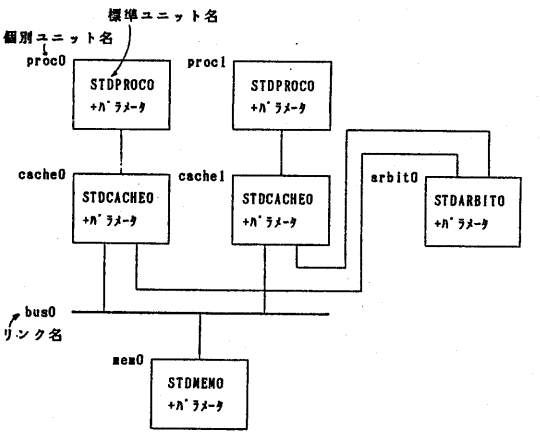
リンクリストには、次のような書式で個別ユニット間の接続関係を記述する。

リンク名：個別ユニットのポートの並び

個別ユニットのポートは次のように指定する。

個別ユニット名．ポート名

この記述例では、バスにbus0という名前がつけられていて、bus0は２つのキャッシュユニットとメモリユニットのポートを結んでいる。なお、リンクの情報が評価に必要でない場合にはリンク名は省略することができる。



（ａ）

```
unit_list
proc0  : STDPROCO ( t_exe=1 )
proc1  : STDPROCO ( t_exe=1 )
cache0 : STDCACHEO( t_acc_hit=2,t_dealy_miss=1,t_delay_ret=1,
                    n_way=4,n_set=128,s_sector=16 )
cache1 : STDCACHEO( t_acc_hit=2,t_dealy_miss=1,t_delay_ret=1,
                    n_way=4,n_set=128,s_sector=16 )
arbit0 : STDARBITO( t_arbit = 1 , t_recov = 0 )
mem0   : STDMEMO  ( t_acc_store = 3 , t_acc_bload = 2 )
link_list
       : proc0.port0 ,  cache0.port0
       : proc1.port0 ,  cache1.port0
       : caceh0.port1 , arbit0.port00
       : caceh1.port1 , arbit0.port01
bus0   : caceh0.port2 , cache1.port2 , mem0.port0
```

（ｂ）

図３－２　BFTDによるマルチプロセッサシステムの記述例

4．マルチプロセッサ評価システム

本評価システムはアーキテクチャ設計や機能設計段階で、システムの実効性能を評価するための性能評価ツールである。システムの実効性能を評価するために、本評価システムは、３節で提案したマルチプロセッサシステム評価技法を導入している。

4．1　本評価システムの特徴

本評価システムは次の３つの特徴を持っている。

（１）トレースデータを用いる。
（２）システムの実効性能を評価する。
（３）システムの記述に要する時間が短い。

本評価システムは、３．２．３や３．２．４で述べたようにＯＳのプロセス管理や排他制御管理のシミュレーションを行うことによって、システムの実効性能を評価する。また、３．２．２．２で述べたようにトレースデータを分割し組み合わせるTDSiml方式のシミュレーションを行うことによって、システムとして動作するために処理した割り込みなどの影響を反映した評価を行う。さらに、３．３．１で述べたように標準ユニットを導入したBFTD方式のシステム記述を採用することにより、システムの記述に要する時間を短くしている。

4．2　本評価システムの構成

本評価システムは、図４－１に示すような構成をしている。評価対象システムのシステム記述と、トレースデータを入力として、次のような性能を評価するための性能指標[5]を出力する。

（１）与えられた複数プロセスの実行を完了するまでの時間
（２）実効ＭＩＰＳ値

また、本システムは、図に示すように、次の３つの部分から構成されている。

（１）トレースデータ分割系
（２）シミュレーション実行系
（３）性能指標導出系

トレースデータ分割系は、３．２．２．１で述べた方式により、入力のトレースデータを分割し、部分トレースデータを出力する。

シミュレーション実行系は、システム記述と、トレースデータ分割系によって用意された部分トレースデータと、シミュレーション条件設定リストを入力として、３．２．２で述べたTDSiml方式によるシミュレーションを実行し、シミュレーション結果を出力する。入力のシミュレーション条件設定リストには、ワークロードが部分トレースデータの名前のリストとして設定される。また、シミュレーション結果には、シミュレーション条件とプロセッサが行ったメモリアクセスの回数などの計数値が記録される。

性能指標導出系は、シミュレーション結果を入力として、性能指標指定リストで指定された性能指標を求める。入力の性能指標指定リストには、ユーザが得たい性能指標をユニットパラメータと計数値を要素とする導出式を用いて指定する。また、性能指標の出力フォーマットを指定する。

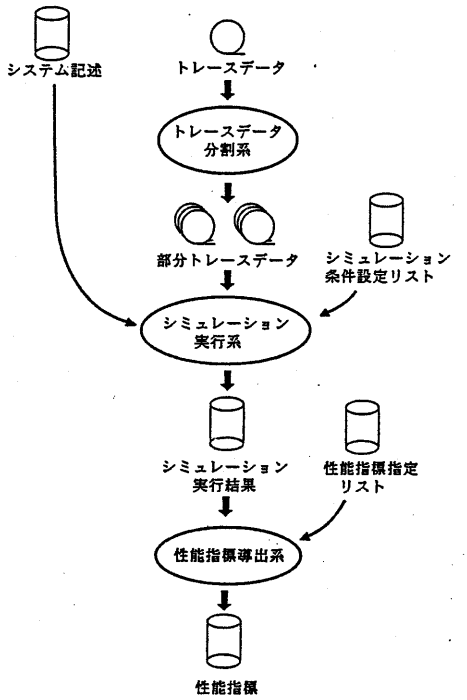


図４－１　本評価システムの構成

4．3　プロトタイプ

本評価システムによる評価の有効性を示すために、プロトタイプを作成し次の評価を行った。

（１）ワークロードによる性能差
（２）バスボトルネックの解析

4．3．1　ワークロードによる性能差

システム性能はワークロードによって変化する。異なるワークロードを使った場合のプロセッサ台数とシステム性能の関係を図４－２に示す。図４－２にはワークロードとしてｃコンパイラ、ｒｅエディタを用いた時のシミュレーションの結果が、それぞれ、実線と一点鎖線で示されている。

プロトタイプでは３．２．２．１で述べたトレースデータの分割は行っていないが、シミュレーションに使用するトレースデータを選択すること

によってワークロードによるシステム性能の違いを評価できる見込みを得た。

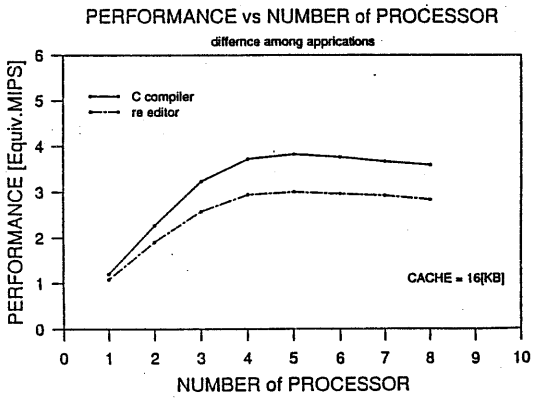


図４－２　ワークロードによる性能差

４．２．２　バスボトルネックの解析

性能要因解析の例として、マルチプロセッサシステムにおける共有バスの占有状況を分析した。プロセッサ台数とバスの占有率（時間比）の関係を図４－３に示す。図４－３にはライトアクセスよるバス占有とリードアクセスによるバス占有とこれらの合計が、それぞれ実線と破線と二点鎖線で示されている。

この評価対象システムではライトスルー方式のキャッシュを採用している。ライトアクセスがそのままバスに出るため、図４－３で示すようにライトアクセスの占有比率が大きい。さらに高い性能が求められる場合には、ライトアクセスのローカリティを利用するコピーバック方式のスヌーピングキャッシュを採用する必要性があることが結論できる。

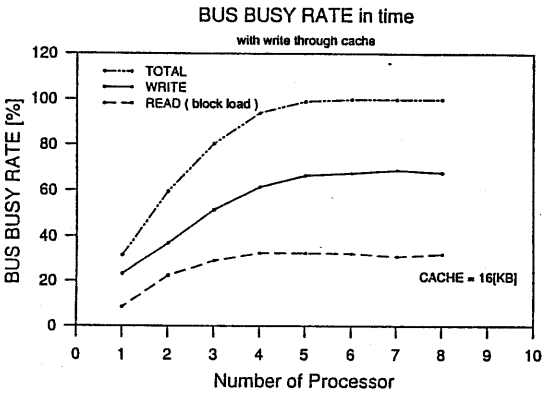


図４－３　共有バスの占有状況

５．まとめ

本評価システムはアーキテクチャ設計や機能設計段階で、システムの実効性能を評価するための評価ツールである。３節で提案したマルチプロセッサシステム評価技法を導入することにより、最も適したシステム構成を選択するための比較検討を効率よく行うことを目指している。現在、著者らはこの評価システムの開発を進めている。本評価システムでは、特に、実計算機のプログラム実行履歴が記録されたトレースデータを用いてシステムの実効性能を求める点と、簡素化されたシステム記述を採用している点が大きな特徴となっている。

アプリケーションにはＩ／Ｏインテンシブなものがあるため、システムの性能を評価するためにはＩ／Ｏの特性を反映した評価を行う必要がある。しかし、このような評価を行うために必要とされる実時間で数十秒といった長い時間の動作シミュレーションには、本評価システムで採用したモデルの粒度は小さすぎる。従来、このようなＩ／Ｏ系を含めたシステム性能の評価には解析的評価、例えば、待ち行列理論を用いた評価が行われている[5]。ところが、ハードウェアの変更に伴うパラメータの設定が難しいという問題点がある。このため、我々は、TDSim1方式による実時間で数秒の動作シミュレーションの結果として得られる平均入出力発生間隔などの性能指標をパラメータとして与え、解析的評価を行う複合シミュレーションの可能性を検討している。

参考文献

[1]松岡浩司,堀川隆,大野直哉:トレースデータを使ったマルチプロセッサ評価システム,情報処理学会,第３６回全国大会,5C-8,(1988).

[2]今井正治,杉山尚志:超ＬＳＩ設計シリコンコンパイレーション,サイエンスフォーラム,(1988).

[3]Delagi,B.,Saraiya,N.,Nishimura,S.,Byrd,G.,:An Instrumemted Architectural Simulation System,Tech.Rept.KSL-86-36,Knowledge Systems Laboratory,Computer Science Department,Stanford Univ.,Jan(1987).

[4]堀川隆,大鷹正之,大野直哉,加藤哲:ハードウェア・トレーサを用いた計算機アーキテクチャ評価システム,情報処理学会,ＯＳ研究会,Feb(1987).

[5]McKerrow,P.:Performance Measurement of Computer Systems,ADDISON-WESLEY PUBLISHING COMPANY,(1988).

[6]紀一誠,納富研造:待ち行列網モデルによる計算機システムの性能評価用ソフトウェアパッケージＱＭ－Ｘ,情報処理学会誌,vol.25,No.4,pp.570-578,(1984).