スーパデータベースコンピュータ (SDC) における性能評価支援ツールの構築とそれによる評価

喜連川 優† 鈴木和宏† 原田昌信†
平野 聡† 高木幹雄†

並列コンピュータの開発においてはマシン自体の構築と同時にその性能評価支援ツールの開発が極めて重要である．現在，関係データベース処理に関しメインフレームをはるかに凌ぐ高い性能をマイクロプロセッサ複合体と高機能ディスクにより実現することを目的とし，スーパデータベースコンピュータ (SDC-I) を開発している．今回 SDC 試作機における2つのバスのバストラヒック，5つのプロセッサの稼働率，2台のディスクの稼働率，ステージングメモリの利用効率等を測定するためのハードウェアモニタならびにソフトウェアモニタを試作するとともに，測定データを評価するための可視化ツールとして SDCView, SDCTacho を試作し，さらにこれらを統合することにより SDC 性能評価支援システムを構築した．実際に SDC 上でウィスコンシンベンチマークを走行させ，本ツールにより解析することにより極めて効率よく動作していることが確認できた．本論文では，性能評価支援システムの各ツールの構成ならびに測定手法，測定結果について述べる．

Performance Evaluation Tool for Super Database Computer (SDC) and Its Evaluation

Masaru Kitsuregawa,† Kazuhiro Suzuki,† Masanobu Harada,†
Satoshi Hirano† and Mikio Takagi†

This paper describes the performance evaluation tool for the super database computer (SDC), which is being developed to attain much higher performance for relational database operations than the current high end mainframes. The evaluation tool combines the hardware and software monitors, with which the bus traffic, the utilization of the memory space, disk activity etc. are measured in detail. Visualization tool (SDC View and SDC Tacho) for measured data is also developed. Using this tool, we could examine that SDC runs the relational query very efficiently.

1. はじめに

単一プロセッサの性能限界が顕在化する中で，近年，並列コンピュータの研究・開発が活発に行われている．並列コンピュータでは複数の処理要素が同時走行するため単一プロセッサに比べ一層その動作把握が困難である．一方，より望ましいマシンアーキテクチャを確立するためには，試作機（あるいは現行マシン）の詳細な解析・評価が不可欠である．またより効率良く並列動作させるためにはプログラマに対しシステムの動作状況を適切に表示することが望まれる．このような背景から並列処理システムに対する性能評価支援ツールに関する研究が活発化しつつある[1)~3)]．

† 東京大学生産技術研究所
Institute of Industrial Science, University of Tokyo

しかしながら現時点では並列マシンに適した汎用の性能評価ツールは存在しないため，対象システムごとに開発せざるを得ない．

我々はマイクロプロセッサ複合体によりメインフレームを大幅にしのぐ性能を達成すべく高並列 SQL サーバ，SDC (Super Database Computer) を開発している．データベース処理は入出力負荷が重く，二次記憶に対する入出力動作と CPU による関係データ処理，さらに相互結合網を通した通信を適切にバランスさせることが重要な課題となる．すなわち主記憶上でのみ走行し，2次記憶へのアクセスを行わない通常のプログラムの性能解析に見られるような CPU のみに着目した評価では不十分である．このことから我々は，システム資源の利用効率を測定することにより，詳細な動作解析を行いアーキテクチャの評価を行うことを目的として SDC 用の性能評価支援システムを構

築したので報告する.

筆者の知る限り，データベースマシンに対する本論文で示すような高精度な性能評価ツールは報告された例はなく，今後のデータベースプロセッサに関する研究・開発にも有益と考える.

第2章では，システムの評価に関する基本的なアプローチについて述べ，第3章では，SDC の概要についてハードウェアとソフトウェアの両面から説明する．第4章において実際に試作した性能データ測定ツール，第5章では性能可視化ツールについて述べる．また，第6章で性能評価支援システムによって得られた測定結果について説明する.

2. 並列計算機システムの性能評価支援

システムの性能評価を支援することは，まずシステムの諸パラメータを測定し，次にその収集されたデータを可視化するという2つのフェーズからなる．以下に，各フェーズにおける基本的なアプローチについて述べる.

2.1 システムの測定：ハードウェアモニタとソフトウェアモニタ

第一のフェーズであるシステムの測定とは，性能を評価・解析するためのデータ（以下，性能データ）を収集することに相当する．性能データを収集するためのアプローチはハードウェアモニタおよび，ソフトウェアモニタに分類することができる[4].

ハードウェアモニタは，被測定システムの外から専用のハードウェアによって測定するため，被測定システムに与える影響が小さいことが特徴である.

それに対し，ソフトウェアモニタは被測定システム内で実行環境を共有するため，オーバヘッドを伴うが，より多様な性能データを柔軟に計測することが可能であるという特徴を有する．ソフトウェアモニタでは，被測定プロセス中にモニタルーチンを埋め込んで性能データを測定する．そのためには，システムに用意されたライブラリを明示的に付加する方法，コンパイラに測定機能を持たせ自動的にモニタルーチンを埋め込む方法等がある[3),5].

我々は第4章で詳述するようにバストラフィックをハードウェアモニタにより，また CPU の稼働率ならびに各種バッファの使用量はソフトウェアモニタにより取得するという，両者の長所を融合したシステムを構築することとした.

2.2 性能データの可視化

第一の測定フェーズで収集されたデータは一般に膨大な量となることから，何らかの手法によりこれを解析者に対して理解容易な形で提供することが望まれる．最も基本的な手法は各種測定データを時間経過に対して示す対時間グラフである．一般に各種測定データは異なる測定系から得られるため，それらを時間的に整合させて表示する必要があり，専用の可視化ツールが不可欠である．これは一般に測定後に詳細な解析を行う際に有効であるが，測定時にオンラインでシステムステータスをモニタできることもシステムの動作把握に大変有効である.

第5章において詳述するように，我々は SDCView なる各種収集データを統合的に可視化するルーチンならびに，SDCTacho なる SDC の動作状況をオンラインで表示するルーチンの2つを開発した.

3. スーパデータベースコンピュータ SDC の概要

3.1 ハードウェアアーキテクチャ

スーパデータベースコンピュータ（SDC）は SQL を高速に実行することを目的とした高並列リレーショナルデータベースサーバである．**図1**に SDC のアーキテクチャを示す．SDC はプロセッシングモジュール複数台をデータネットワークならびにコントロールネットワークにより結合した構成をとっており（図1(a)），各モジュールは我々が過去研究・試作により有効性を確認している機能ディスクシステム[6)~9)]に相当し，図1(b)に示される構成となっている．またモジュールの一部はホストや通信網との接続にも利用される.

SDC の1つのモジュールはバス結合型共有メモリマルチプロセッサアーキテクチャを採り，2台のソフトウェアストライピングディスクと5台のマイクロプロセッサを2本のバスで結合している．HBus はデータベース内のレコードを DMem 上に高速に転送するためのデータ転送専用バスであり，一方 CBus は CMem 上の制御情報に対する排他的アクセスのために用いられる．次章で述べるように，我々はこれらの2本のバス上のトラヒックを測定するためのハードウェアモニタを開発した.

5台のプロセッサのうち1台はコントロールプロセッサ CP と呼ばれ，モジュール全体の制御を司るとともに2台のディスク入出力を実時間駆動する．1モジ

ュール内の評価対象とする各資源を以下に示す.

CP: モジュール全体と I/O を制御するプロセッサ MC 68020 (20 MHz)

P 0–P 3: データベース処理を行うプロセッサ群 MC 68020 (20 MHz)

DMem: ステージング用メモリ 8 MByte

CMem: コントロール情報, ロック用のメモリ

DNA: データ転送用オメガネットワークインタフェース

CNA: コントロール情報転送用ネットワークインタフェース

DC 0, DC 1: ディスクコントローラ

Disk 0, Disk 1: 磁気ディスク

HBus: データ用高速バス (40 MB/sec)

CBus: コントロール用バス

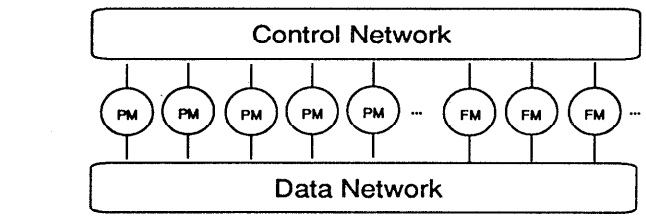
SDC ではこれらのモジュールをバケット平坦化機能を有するオメガネットワーク[10]により結合している. 本ネットワークにより処理対象となるバケットデータをモジュール群に対しプロセッサの助けなしに均等に分配し, これによりモジュール間の負荷の均等化を図っている. 以上のように SDC は密結合マルチプロセッサをオメガネットワークにより疎に結合したハイブリッドアーキテクチャを有している.

3.2 SDC におけるコンテナモデルとシステムソフトウェアの構成

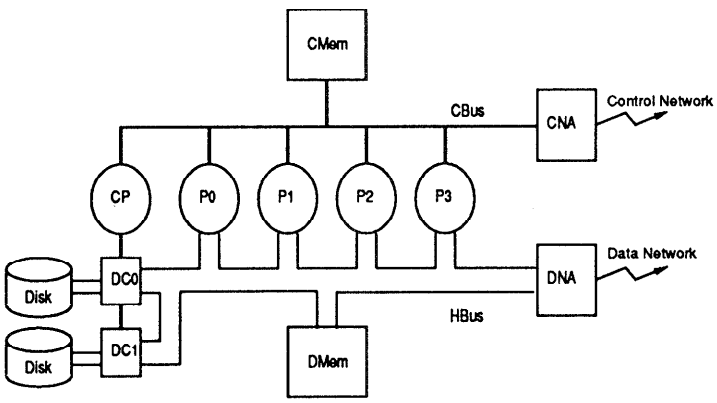
3.2.1 コンテナモデル

前節で述べた SDC のアーキテクチャ上で我々の提案するバケット平坦化 GRACE ハッシュジョインアルゴリズム[11]を高速に実行すべく, 「コンテナモデル」なるシステムソフトウェア構築のためのフレームワークを採用している[12]. DBMS は OS 配下の 1 つのアプリケーションとみなされ, その入出力は OS を介して行われることになる. OS と DBMS とでは入出力に対するバッファ管理手法やデータ管理手法が必ずしも一致せず, この不整合性から大きな性能低下が生ずると指摘されてきた[13]. これに対し, コンテナモデルではコンテナと呼ぶ複数個のタプルの入った器をタスクの単位としてシステム最下層ソフトウェアのレベルでサポートすることにより高効率化を図っている. コンテナモデルの詳細は文献 12) にゆずり, ここでは本論文での測定対象を明らかにするためコンテナの流れの概略を**図 2** に示す.

ディスクから連続的に読み出されるデータおよび, ネットワークを転送されてきたデータは, それぞれ数個のタプルを含むタスクとしてコンテナに詰められる. コンテナは, 仮想的なパイプを通してプロセッサに渡される. このパイプは 2 台のディスクとネットワークを同一とみなすことを可能にすると共に, プロセッサ群の処理能力の変動を吸収するためのバッファの役割を果たす. バッファは複数のコンテナをリストとして管理し, 空のコンテナがフリーリストを, またデータが詰められたコンテナがフルリストを構成している. フリーリストから取得した空のコンテナにディスクおよびネットワークからのデータが満たされるとフルリストの最後尾に接続され, プロセッサによる処



(a) SDC の全体像
(a) SDC overview



(b) モジュールの構成
(b) Hardware architecture of a PM

図 1 SDC のアーキテクチャ
Fig. 1 Architecture of The SDC.

理を待つ．データ処理プロセッサはフルリストの先端を監視し，コンテナが接続されると競争して取得し処理を行い，モジュール内のプロセッサ間での負荷分散が実現される．処理済みのデータは同様にディスクに対するタスクとしてコンテナに詰め込まれ，後段のパイプへ送られる．処理が終了し，空となったコンテナは再びフリーリストに追加される．ディスクの出力バッファは，ディスクの入出力モードを切り替えるタイミングを決定するために中断点（Suspend Point）という閾値を備える．ディスク出力バッファ中のコンテナ数が中断点を上回るとディスクの入力動作を中断し，出力モードに切り替えて空きコンテナ数の初期値の4分の3程度を回収するようにディスク出力バッファの内容をディスクへ書き出す．ディスクの出力を行っている間に，使用されていたバッファのコンテナは順次解放されてゆくため，プロセッサ群はネットワークからのデータに対する処理を継続することが可能である．

結合演算などの処理すべきデータ量が主記憶の容量を越える場合には，中間リレーションを作る必要がある．この時ディスクの出力バッファとして，バケットバッファと呼ぶバッファをバケットごとに用意する．各バケットバッファは，それぞれ固定量の空きコンテナを有するのではなく1つのフリーリストを共有しており，そこから空きコンテナを取得する．

ネットワークを介したデッドロック回避のためネットワークの出力バッファについても同様に管理している．コンテナ数が中断点を上回ると，ディスクの入力を中断し，バケットバッファの書き出しを行う．

後述するソフトウェアモニタを用い各バッファでのコンテナ消費量を測定することによりシステム内でコンテナが淀むことなく処理されていることを確認する．

3.2.2 プロセス構成

上述のコンテナモデルを実現すべく SDC では図3に示されるプロセス構成を有している．1つのモジュールは PMC と呼ばれるモジュール制御プロセスによって管理され，1つの関係演算は Executer により複数の PMC の協調によって実行される．またモジュール内ではデータベースレコードを処理するデータ処理プロセス（DPP）が各 Pn 上で走行し，ネットワークならびにディスク管理プロセスが CP 上で走行する．

4. 性能データ測定ツール

SDC における性能評価支援システムは，性能測定ツールおよび，性能可視化ツールの2つのツールによって構成される．さらに，性能測定ツールはハードウェアモニタである「バスモニタ」と，ソフトウェアモ

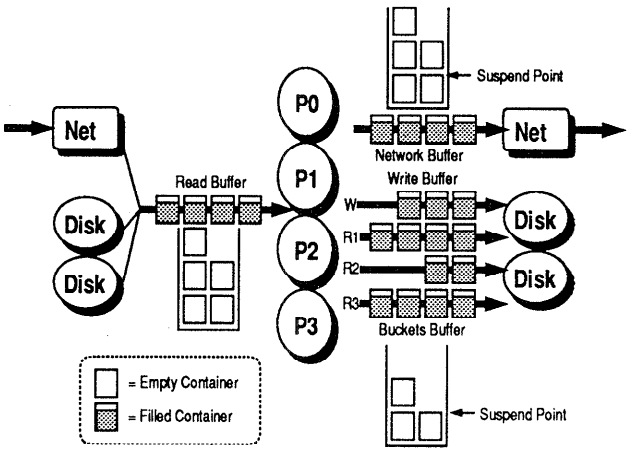
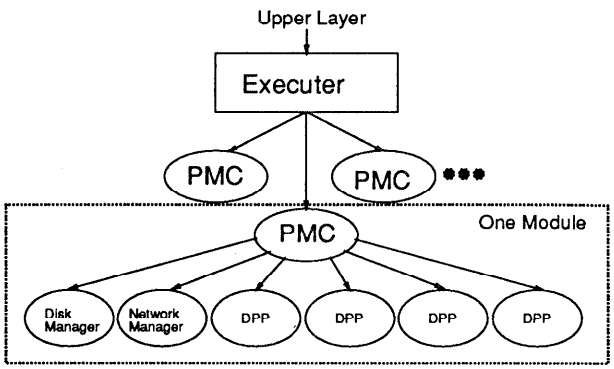


図 2 コンテナモデルによる関係データベース処理
Fig. 2 Relational database process in container model.



PMC: Processing Module Control Process; DPP: Data Processing Process

図 3 SDC におけるシステムソフトウェアの構成
Fig. 3 System software architecture of The SDC.

ニタである「パフォーマンスモニタ」の2つのツールによって構成される[14),15)]．すなわち本ツールは，ハードウェアモニタとソフトウェアモニタのそれぞれの利点を生かすために，ハイブリッドモニタの構成をとる．

本章では，性能測定ツールであるバスモニタおよび，パフォーマンスモニタのそれぞれについて，その構成や動作原理等を述べる．

4.1 バスモニタ

バスモニタは，処理モジュール内の共有バスの使用率を測定するためのハードウェアモニタである．したがって本ツールは，ハードウェアモニタの最大の特徴を生かして，被測定システムである SDC に対してほとんど影響を与えることなく，性能データの測定を行うことが可能である．ハードウェアモニタの駆動時に，CP 上でそのためのわずかなソフトウェアを実行することになるが，それ以外には被測定系に対し影響を与えることはない．

SDC における処理モジュール内には，前述のごとく CBus，HBus の2つの共有バスが存在する．CBus を使用する資源として，1台の CP ならびに4台の Pn があり，バス上には CBus バスアービタからそれぞれのプロセッサへ，バス使用許可信号が1本ずつ計5本出ている．

一方，HBus を使用する資源としては，5台のプロセッサ，2台のディスク，およびネットワークインタフェースがある．バス上には，CBus と同様に HBus バスアービタからそれぞれの資源へバス使用許可信号が1本ずつ計8本出ている．

バスモニタは，これら計 13 本のバス使用許可信号から4本を選択しバス使用許可信号が，一定時間内でアクティブになっている時間を測定し，バスの使用率を求める．

4.1.1 バスモニタの構成

バスモニタのブロック図を図4に示す．構成要素は，以下のとおりである．

マルチプレクサ：コントローラからの指示により複数のバス使用許可信号の中から測定の対象となる4つを選び出す．この時，マルチプレクサは『すべての処理用プロセッサの HBus 使用率』のように複数のバス使用許可信号を加算する機能も備えている．

カウンタ：マルチプレクサによって選択された信号が，アクティブになっている時間を計測する．カウンタのビット数は8ビットである．1ビット当たりの時間分解能は，コントローラからの指示により 50 ns もしくは 100 ns が選択される．これは，SDC におけるシステムクロックが 50 ns であり，CBus は非同期バスを，HBus はバスサイクル 100 ns の同期バスを採用しているためであり，どちらのバスを詳しく測定するかによって，1ビット当たりの時間分解能を変更可能となっている．

メモリ：カウンタの計測値を記録する．4バンクあり，容量はそれぞれ 2 MByte である．

メモリコントローラ：タイマからの書き込み要求や，コントローラからの読み出し要求により，メモリに与えるアドレスの管理，および読み／書きの制御を行う．

タイマ：一定時間ごとにメモリコントローラに対し書き込み要求を出すと同時に，カウンタをリセットする．

コントローラ：バスモニタ全体の制御，および CBus とのインタフェースを司る．

4.1.2 測定時間

バスモニタによって採取されるデータは上記のメモリバンクに順次格納されるが，メモリ容量と測定精度によりその測定可能な時間が決まることになる．

バスモニタはタイマの値が上限設定値になるたびに，カウンタの内容をメモリへ書き込む．従って，最大測定可能時間はすべてのメモリ空間に測定結果が書き込まれるまでの時間となる．

タイマのビット数は8ビット，9ビット，10ビットのうち1つが選択される．また，1ビット当たりの時間分解能は，カウンタと同一で 50 ns もしくは 100 ns

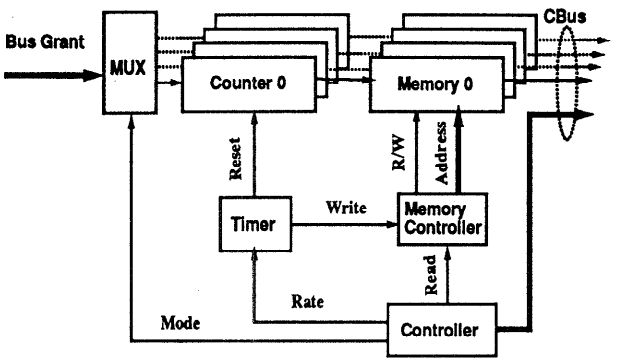


図 4 バスモニタのブロック図
Fig. 4 Block diagram of The Bus Monitor.

である.

いまタイマを8ビットとし，1ビット当たりの時間分解能を 100 ns にした時，カウンタの値がメモリに書き込まれる時間間隔は，

$100\,\text{ns} \times 2^8 \simeq 25.6\,\mu\text{s}$

である.

よって，測定可能時間はメモリの容量が 2 MByte であるため，

$25.6\,\mu\text{s} \times 2.2^{20} \simeq 54$ 秒

となる.

この時，カウンタのビット数はタイマと同じく8ビットであるため，測定され得るバス使用率の上限は100% となる.

また，タイマを9ビットとすると，カウンタのビット数は固定されているためバスの使用率の上限は 50 % となり，バス負荷の重い測定はできなくなるが，測定時間は2倍，すなわち約108秒とすることが可能となる. 同様に10ビットと設定することも可能であり，この際バス使用率の上限は 25%，測定可能時間は約216秒となる.

一方，1ビット当たりの時間分解能を 50 ns にした時の，測定可能時間は，タイマのビット数により約27秒，54秒，108秒となる.

以上を**表1**にまとめる.

4.2 パフォーマンスモニタの機能と構成

性能測定ツールを構成する，もう1つのツールがパフォーマンスモニタである. パフォーマンスモニタは，モジュール内のプロセッサの稼働率および，共有メモリ（DMem）上の各種バッファの使用率を測定するための，純粋なソフトウェアモニタである.

4.2.1 性能データの収集

パフォーマンスモニタは SDC 上のプロセスの中に各種のモニタルーチンを組み込み，性能データの収集を行う. 以下にそれぞれの性能データの測定方式について述べる.

● データ処理プロセッサ Pn の稼働率

データ処理プロセッサ群 Pn は共有メモリ上に用意された仮想的なパイプの出口からディスクからのデータをタスクとして取得し関係演算を施す. パイプの出口にタスクが到着していない時はタスクの到着を待つ. そこで，データ処理プロセス DPP 内にカウンタを用意し，タスクの到着を待っている間にこれをインクリメントするようなルーチンを付加する.

これらのカウンタの値を一定時間ごとに読み出し，最大値と比較することによって Pn の稼働率を求めることができる.

● コントロールプロセッサ CP の稼働率

データ処理中に CP 上で行われるのは，ディスクやネットワーク等の I/O 処理である. これらの処理はすべて割り込みプロセスとして実行される. 従って，全体の時間に対して割り込み以外の時間が CP の空き時間となる. 空き時間を計測するため，非割り込み時に CP 上でカウンタをインクリメントするプロセスを走らせる. 割り込み処理プロセスは空き時間測定プロセスよりも優先順位が高く，かつプリエンプトされないため，あらかじめ測定しておいた無負荷時のカウンタの値と測定したカウント値を比較することにより割り込み処理での CP の稼働率を算出することができる.

● 各種バッファの使用率

データメモリ（DMem）は，ディスクおよびネット

表 1 バスモニタにおけるモード設定と最大測定時間
Table 1 Monitor modes and maximum measurable time of The Bus Monitor.

タイマへの設定ビット数	時間分解能		測定可能な最大負荷
	50 ns	100 ns	
8 ビット	27 秒	54 秒	100%
9 ビット	54 秒	108 秒	50%
10 ビット	108 秒	216 秒	25%

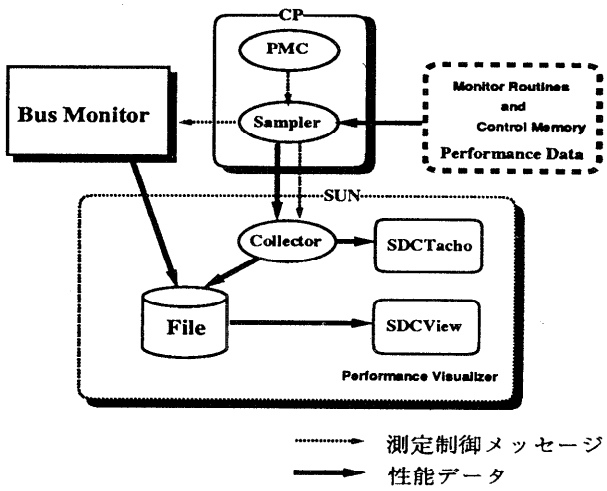


図 5 パフォーマンスモニタの構成
Fig. 5 Overall view of The Performance Monitor.

ワーク入力用のリードバッファ，結果タプル出力用のライトバッファ，中間ファイル出力用のバケットバッファ，および，ネットワーク出力用のネットワークバッファのそれぞれのバッファとして用いられる．各バッファは図2に示したように複数のコンテナをリスト構造によって管理している．コンテナのヘッダ部分はコントロールメモリ (CMem) 上に格納され，データメモリ上にはデータ部分のみが存在する．

このコンテナに関する管理情報，すなわちフルリストに接続されたコンテナの数をコントロールメモリから調べることによってデータメモリの各バッファの使用率を算出することができる．

4.2.2 パフォーマンスモニタの構成

パフォーマンスモニタは，**図5**に示すようにSDCのCP上で動作するSamplerおよびワークステーションSUN上で動作するCollectorによって構成される．

Samplerは，PMCから送られる測定開始要求メッセージを受け性能データの測定を開始する．測定は一定時間ごとに性能データを読み出し，それらをネットワークを通してCollectorに送出することにより行われる．読み出された性能データをその都度転送すると，プロセッサおよびネットワークの負荷が増大し，システムに与える影響が大きくなることが懸念される．そこで本ツールでは10 msecごとに性能データを読み出し，それを30回収集した後送出することでシステムに与える影響を軽減することとした．

またSamplerはバスモニタの制御も行う．SDCの動作と同期してバスモニタによる測定を開始することが可能である．

SamplerはPMCからの測定終了要求メッセージを受けると測定を終了し，Collectorに対し性能データ収集の終了を知らせる．この時バスモニタの終了を待ってバスモニタ上の測定データの前処理を行い，その結果をネットワークを通してSUNへ転送する．

CollectorはSUN上でSamplerから転送される測定データを受けとり，SDCViewが必要とするフォーマットに変換した後，性能データをファイルとして格納すると共に，並行してSDCTachoに対してデータを順次供給する．

5. 性能可視化ツール

SDCにおける性能可視化ツール[16]は「SDCView」と「SDCTacho」の2つのソフトウェアツールからなる．前者は対時間グラフとして静的に性能データを解析するためのツールであり，後者は性能データの変化を動的に観測するためのオンラインツールである．以下にそれぞれのツールについて述べる．

5.1 SDCView

バスモニタおよび，パフォーマンスモニタは専用の可視化ツールであるSDCViewによって性能データ

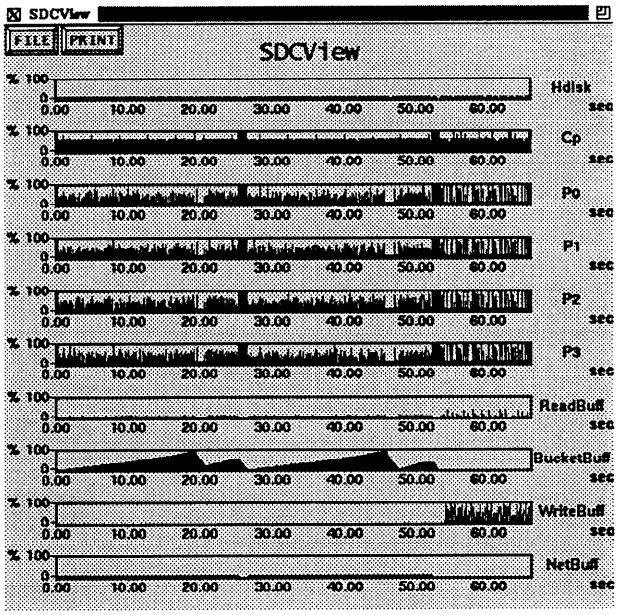


図6 SDCViewの概観
Fig. 6 Overview of The SDCView.

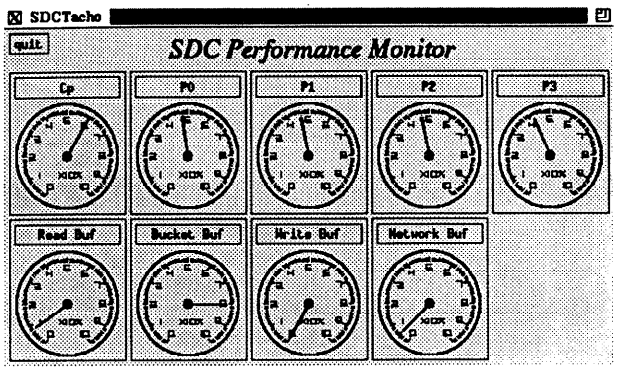


図7 SDCTachoの概観
Fig. 7 Overview of The SDCTacho.

を可視化する．SDCView は性能データが格納されたファイルを読み込むと，**図6**に示されるようなウィンドウを表示する．

ウィンドウ内のグラフは横軸に秒を単位とした時間を，また縦軸にはそれぞれの資源の使用率を百分率で示す．各性能データは縦に並べて表示され，処理の進行にそった定量的な解析を容易にする．ウィンドウシステムとしては，X window system を用いている．

SDCView は次のユーザインタフェースを有する．

グラフの拡大 ウィンドウ内でマウスをドラッグすることによって，新たなウィンドウにグラフを拡大表示することができる．

時間の表示 同様にマウスによって二点間の経過時間がポップアップウィンドウに表示される．

入力ファイルの選択 ウィンドウ内のボタンを操作することによって，所望のファイルを選択し表示する．

ハードコピー ページ記述言語 PostScript によりプリンタデバイスに出力する．またドローイングエディタ idraw の形式で出力することも可能である．

5.2 SDCTacho

パフォーマンスモニタによって得られる性能データを動的に可視化するツールが SDCTacho である．SDCTacho は**図7**に示されるようにウィンドウ内に，Collector から送られる性能データをリアルタイムに表示する．すなわち性能データを SDC の実行と同時に観察することを可能とするツールである．各性能データは最大値を100% とするアナログメータとして表示され，その針の振れによって性能データの変化が表現される．

6. 測定結果

6.1 バスモニタによる HBus の使用率の測定

図8に，バスモニタによって得られた性能データを SDCView を用いて可視化した例を示す．グラフは拡張ウィスコンシンベンチマークテストを2モジュールで行った時に得られた性能データを示したものである．ベンチマークテストの条件は，タプル長 208 バイト，タプル数 100 万件，選択率 10% の結合演算である．

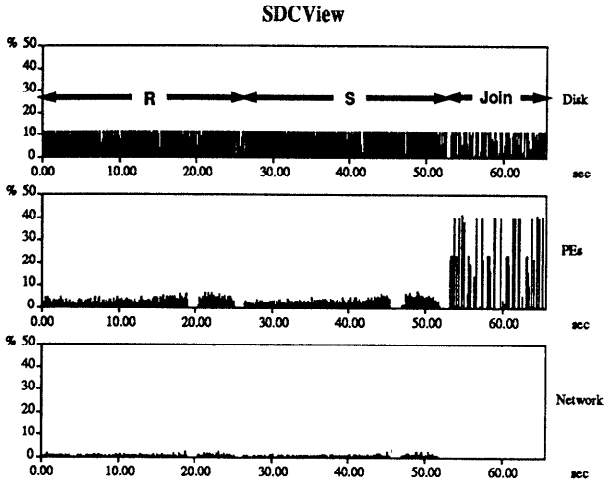


図8 HBus の使用率 (SDCView による表示)
Fig. 8 Bus trafic of HBus (visualized with The SDCView).

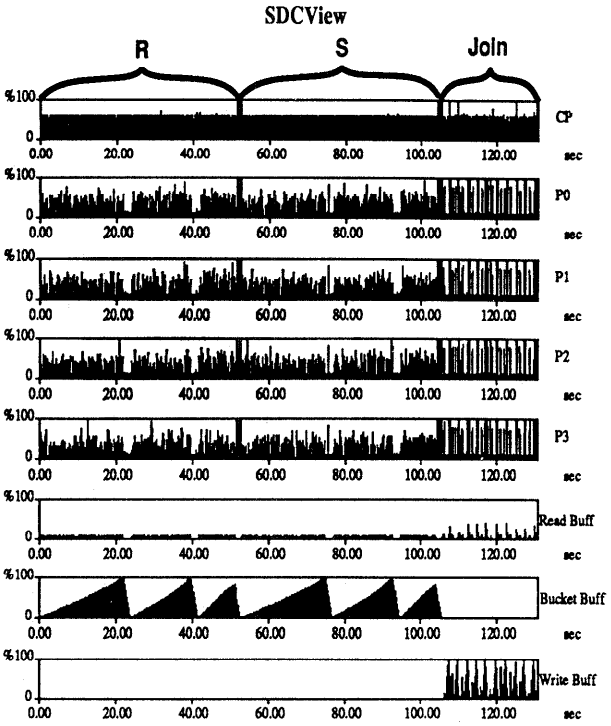


図9 各資源の使用率 (SDCView による表示)
Fig. 9 Utilization ratio of each resources (visualized with The SDCView).

グラフは，HBus の使用率を示しており，各データは上からディスク，処理プロセッサ群，およびネットワークである．R，Sで示される部分が各リレーションのスプリットフェーズであり，Join で示された部分がジョインフェーズを示している．

この結果から，HBus は使用率が最も高いジョインフェーズにおいても，まだ余分を残しており，バスの飽和による性能低下は起きていないことが確認された．

6.2 パフォーマンスモニタによる各資源の使用率の測定

図 9 にパフォーマンスモニタによって得られた性能データを，同様に SDCView によって可視化した例を示す．グラフは，上記のベンチマークテストを1モジュールで行った結果である．各データは上から，CP，P0〜P3 の稼働率，リードバッファ，バケットバッファ，ライトバッファの使用率である．

スプリットフェーズ (S, R) では P0〜P3 のグラフは，ほとんど同じ形をしておりモジュール内での負荷分散が十分実現されていることがわかる．また使用率は 60% 程度の値を示しており，プロセッサ群はまだ余分を残していることがわかる．リードバッファは，ほぼ一定の値で約 10% の使用率である．すなわち，リードバッファが一杯になり，ディスク入力が中断されるような現象は見られず，このことからも，プロセッサ群による処理がI/O の速度に追随していることがわかる．

バケットバッファの使用率は直線的に増加し，100% になると，減少している．これによって，読み込んだリレーションをバケットに分割する際，バケットバッファが一杯になると，読み込みが中断され各バケットが中間ファイルとして書き出されているということが直観的に理解できる．

Join で示されたジョインフェーズを拡大したものが**図 10** である．グラフは1組のバケットの結合演算部分を示しており，Build で示された部分が，一方のリレーションの当該バケットを読み込みハッシュテーブルを生成するビルドフェーズに，Probe で示された部分が他方のリレーションの当該バケットを読ち出しハッシュテーブルを検索（プローブ）し，結果タプルを生成するプローブフェーズに対応している．また Write で示される部分がバケットバッファの内容をディスクに書き出す結果出力部である．

ライトバッファはプローブフェーズにおいて生成されるタプルが格納される．図9におけるバケットバッファの振舞いと同様にバッファが一杯になるとディスクへ出力され，空になると再びプローブ処理が再開される．このことから，バッファの使用率は山形になっていることがわかる．

ビルドフェーズでは，プロセッサの使用率はスプリットフェーズよりも高い値を示しているが100% には至っていない．それはリードバッファの使用率が，スプリットフェーズとほとんど変化がないことからもわかる．

プローブフェーズでは，プロセッサは100% の使用率を示しており，リードバッファが約 40% まで使用されている．これは，プロセッサの処理がI/O に間に合わず，ディスクから読み込まれたデータがバッファリングされるためである．しかしこの時でもリードバッファが溢れる前にライトバッファが一杯になりディ

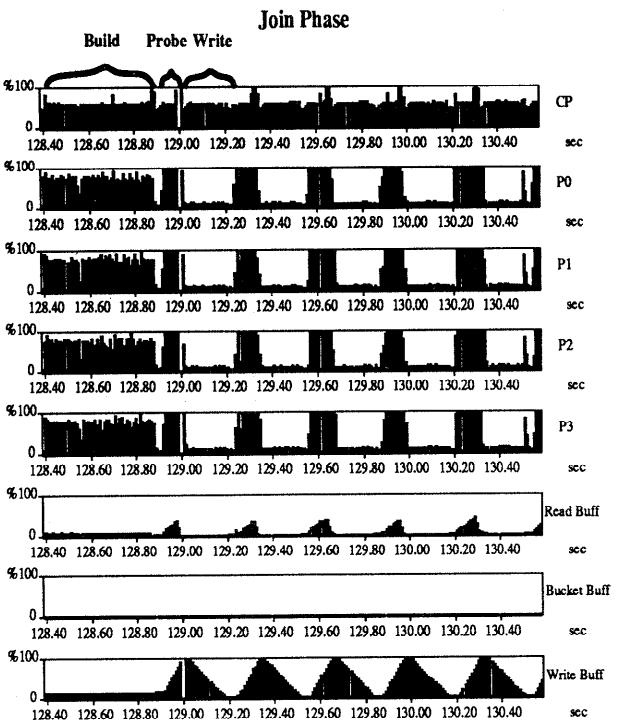


図 10 ジョインフェーズの拡大図
Fig. 10 Magnification of Join phase.

表 2 パフォーマンスモニタによる影響
Table 2 Pertuabation of performance by The Performance Monitor.

	使用時	未使用時	影響(%)
1 module	130.92 sec	130.50 sec	0.32
2 module	65.45 sec	65.36 sec	0.14

スクへの書き込みが駆動され，その間に溜ったデータが処理されるため，ディスクが停止することなく効率良く動作していることがわかる．

6.3 パフォーマンスモニタによる影響

パフォーマンスモニタは純粋なソフトウェアモニタであるため，SDC に対する影響が懸念される．そこで，パフォーマンスモニタを使用した場合と使用しない場合のベンチマークテストの実行時間を**表 2** に示す．

これにより，パフォーマンスモニタの影響は約 0.3 % であり，懸念されたパフォーマンスの低下は極めて小さいものであることが確認された．

7. おわりに

本論文では，スーパデータベースコンピュータ SDC における性能評価支援システムとして性能データ測定ツール（バスモニタならびにパフォーマンスモニタ）および性能データ可視化ツール（SDCView ならびに SDCTacho）についてその構成法を述べるとともに，実際に試作し SDC に適応することによって処理モジュール内のバストラヒック，プロセッサおよびステージングメモリの使用率を測定した結果について述べた．

バスモニタによって，モジュール内の共有バスはまだ余力を残しておき，バスの飽和による性能低下はおきていないことが確認された．またパフォーマンスモニタによる観測から，モジュール内での良好な負荷分散が実現されていることが確認された．SDC はウィスコンシンベンチマークにおける 1M タプルジョインを約 60 秒（2 モジュール時）という極めて高い性能で実行しており，本論文によるモニタによりシステムが設計どおりに効率良く動作し，その高い性能を達成していることをより詳細に把握することができた．

またパフォーマンスモニタが SDC に与える影響は極めて小さく，さらに本ツールを用いたシステムの動作解析の有効性も確認された．

今回は GRACE Hash Join アルゴリズムを用いた結合演算を行った時の測定に留まったので，今後は他の関係演算についてさらに詳しく性能評価を進める予定である．

参考文献

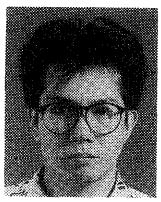
1) Lehr, T., Segall, Z., Vrsalovic, D. F., Caplan, E., Chung, A. L. and Fineman, C. E.: Visualizing Performance Debugging, *IEEE Computer*, Vol. 22, No. 10, pp. 38-51 (1989).
2) Malony, A. D., Reed, D. A. and Rudolph, C. C.: Integrating Performance Data Collection, Analysis, and Visualization, *Parallel Computer Systems: Performance Instrumentation and Visualization*, Simmons, M. and Koskela, R. (eds.), pp. 73-97, ACM (1990).
3) Kenny, K. B. and Lin, K.: Measuring and Analyzing Real-time Performance, *IEEE Software*, Vol. 8, No. 5, pp. 41-49 (1991).
4) Wybranietz, D. and Haban, D.: Monitoring and Measuring Distributed Systems, *Parallel Computer Systems: Performance Instrumentation and Visualization*, Simmons, M. and Koskela, R. (eds.), pp. 27-45, ACM (1990).
5) Malony, A. D.: Jed: Just an Event Display, *Parallel Computer Systems: Performance Instrumentation and Visualization*, Simmons, M. and Koskela, R. (eds.), pp. 99-115, ACM (1990).
6) Kitsuregawa, M., Nakano, M., Harada, L. and Takagi, M.: Functional Disk System for Relational Database, *Proc. of 3rd Int. Conf. on Data Engineering*, pp. 88-95 (1987).
7) Kitsuregawa, M., Nakano, M. and Takagi, M.: Query Execution for Large Relations on Functional Disk System, *Proc. of 5th Int. Conf. on Data Engineering*, pp. 159-167 (1989).
8) Kitsuregawa, M., Nakano, M. and Takagi, M.: Performance Evaluation of Functional Disk System (fds-r2), *Proc. of 7th Int. Conf. on Data Engineering*, pp. 416-425 (1991).
9) 喜連川，中野：機能ディスクシステム：関係データベース処理とその性能評価，電子情報通信学会論文誌，Vol. J 74-D-I, No. 8, pp. 496-507 (1991).
10) 喜連川，小川：バケット平坦化機能を有するオメガネットワーク，情報処理学会論文誌，Vol. 30, No. 11, pp. 1494-1503 (1989).
11) Kitsuregawa, M. and Ogawa, Y.: Bucket Spreading Parallel Hash: A New, Robust, Parallel Hash Join Method for Data Skew in the Super Database Computer (sdc), *Proc. of 16th Int. Conf. on VLDB.*, pp. 210-221 (1990).
12) Kitsuregawa, M., Hirano, S., Harada, M., Nakamura, M. and Takagi, M.: The Super Data-

base Computer (sdc): Architecture, Algorithm and Preliminary Evaluation, *Proc. of HICCS-25, 25th Hawaii Int. Conf. on Computer Sciences*, pp. 308-319 (1992).
13) Stonebraker, M.: Operating System Support for Database Management, *CACM*, Vol. 24, No. 7, pp. 412-418 (1979).
14) 原田，鈴木，平野，中村，喜連川，高木：スーパーデータベースコンピュータ (sdc) のバスモニタ，第 42 回情報処理学会全国大会論文集，3L-9 (1991).
15) 鈴木，平野，喜連川，高木：スーパーデータベースコンピュータ (sdc) における性能評価ツール，第 43 回情報処理学会全国大会論文集，6N-7 (1991).
16) 鈴木，平野，喜連川，高木：スーパーデータベースコンピュータ (sdc) における性能可視化ツール，第 44 回情報処理学会全国大会論文集，5H-8 (1992).

(平成 4 年 5 月 13 日受付)
(平成 5 年 1 月 18 日採録)

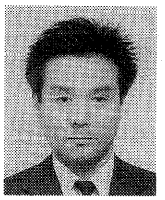
喜連川 優 (正会員)

昭和 30 年生．昭和 53 年東京大学電子工学科卒業．昭和 58 年東京大学工学系研究科情報工学博士課程修了．工学博士．昭和 59 年東京大学生産技術研究所第 3 部講師．昭和 60 年同所機能エレクトロニクス研究センター助教授，現在に至る．並列コンピュータアーキテクチャ，データベース工学の研究に従事．DataBase Machines and KnowledgeBase Machines (Kluwer),「コネクションマシン」(著訳),「データベースベンチマーキング」(訳)．電子情報通信学会，IEEE，ACM 各会員．



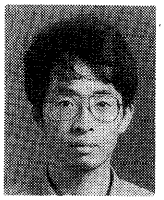
鈴木 和宏

昭和 41 年生．平成 2 年横浜国立大学工学部電子情報工学科卒業．平成 4 年東京大学工学系研究科情報工学修士課程修了．同年(株)富士通研究所入社．並列処理，および，ユーザインタフェース構築環境に興味を持つ．



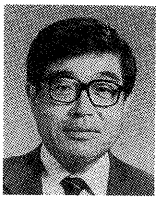
原田 昌信

1963 年生．1986 年防衛大学校理工学部電気工学科卒業．1989 年東京大学大学院工学系研究科電気工学専攻修士課程修了．現在，同博士課程在学中．



平野 聡 (正会員)

1962 年生．1985 年電気通信大学材料科学科卒業．1987 年同大大学院経営工学専攻修了．1992 年東京大学大学院情報工学専攻博士課程修了．同年電子技術総合研究所入所．工学博士．大規模並列システムソフトウェア，オブジェクト指向システム，データベースマシンの研究に従事．IEEE CS 会員．



高木 幹雄 (正会員)

昭和 11 年生．昭和 35 年東京大学工学部電気工学科卒業．昭和 40 年東京大学工学部大学院博士課程修了．工学博士．昭和 40 年東京大学助教授生産技術研究所第 3 部勤務．昭和 54 年東京大学教授．生産技術研究所多次元画像情報処理センター．昭和 59 年東京大学生産技術研究所機能エレクトロニクス研究センター．ディジタル画像情報処理の研究に従事．電気，電子情報通信，TV，ME，IEEE，SPIE，SID 等の会員，画像電子学会評議員．