情報処理最前線

富士通（株）コンピュータ事業本部　黒木 伸一
skr@yk.fujitsu.co.jp
富士通（株）コンピュータ事業本部　泉田 直樹
izuta@cs.fujitsu.co.jp
富士通（株）ソフトウェア事業本部　小澤 一孝
ozawa.kazutaka@jp.fujitsu.com
（株）富士通研究所 コンピュータシステム研究所　土肥 実久
micky@flab.fujitsu.co.jp

大規模SMPによるTPC-Cベンチマーク

我々は，128CPU PRIMEPOWER2000とSymfoWARE Server（RDBMS）を使用してTPC-Cベンチマークの測定を行い，非クラスタシステムでの世界最高性能455,818tpmC（2001年9月末現在）を実現した．それ以前の性能値からみても，十分なスケーラビリティが得られている．これは64CPU程度がSMPシステムにおけるスケーラビリティの上限であるとのこれまでの通説を覆し，128CPU SMPシステムであっても性能のスケーラビリティを確保することができるとの実証でもある．本稿では，TPC-Cの簡単な説明と今回使用したシステムの紹介をし，このような大規模SMPシステムにおけるチューニング，またシステム構成上およびプログラミング上の留意点をあげる．本稿がこのような大規模システムを構築する上での一助となることを願っている．

TPC-Cとは

《概　要》

TPC-CはTPC（Transaction Processing Performance Council)が制定したOLTPのベンチマークテストである．OLTPシステムのベンチマークテストとして現在最も権威のあるものの1つで，以下の特徴がある．

- 評価基準や評価方法の統一
- 規定の厳密化
- ハードウェア／ソフトウェア性能を統合した評価
- 第三者による監査の義務化
- 性能，価格，システム構成情報の公開→詳細文書（FDR）の提出

TPCが制定したベンチマークテストには，TPC-Cの他にTPC-H，TPC-R（いずれもDSS系）およびTPC-W（Webシステム系），現在使われていないがTPC-A，TPC-B（いずれもバンキング系）およびTPC-D（DSS系，TPC-HとTPC-Rに分化）がある．

《モデル》

TPC-Cモデルは，小売，流通業向けの受発注システムで，**図-1**のようにW軒の問屋がそれぞれ10軒の支店を持ち，それぞれの支店には3,000の顧客があると想定している．

実行されるトランザクションは，①注文，②支払い，③注文問合せ，④配送，⑤在庫管理の5種類である．これは，現在の大規模流通システムの典型例となっている．

《性能指標》

TPC-Cの性能指標としては以下の3項目がある．

- 処理性能：tpmC（毎分の処理件数）
- 価格性能比：$/tpmC（通貨はドルの必要はないが，各社ほとんどドルで登録している）
- 利用可能日：Availability Date（評価システムの市場での利用可能日）

評価結果は監査人による監査を受けた後，TPC委員会に，性能指標，再現性，各トランザクションのレスポンス，トランザクションミックス，キーイン時間，思考時間，実測時間（Ramp Up，測定時間），Checkpoint，ソースコード，実行パラメタ等をFDR（Full Disclosure Report）として登録する．FDRは委員会のホームページ（http://www.tpc.org）から参照することができる．

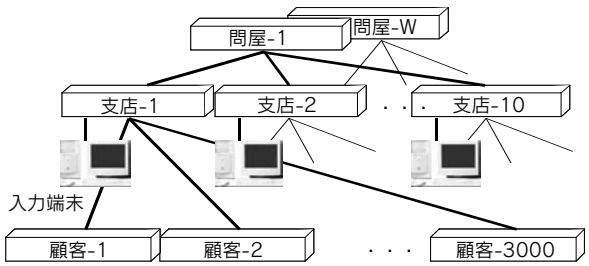

図-1　TPC-Cモデル

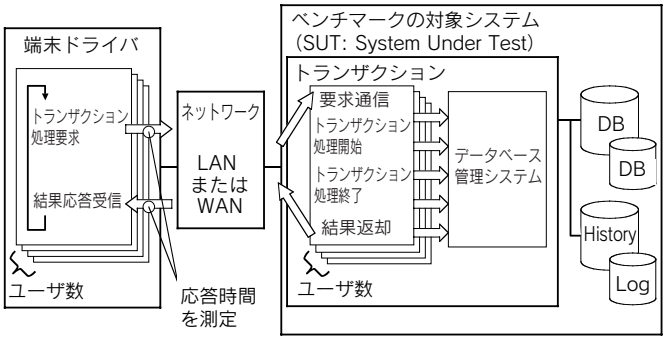

図-2　評価システム構成

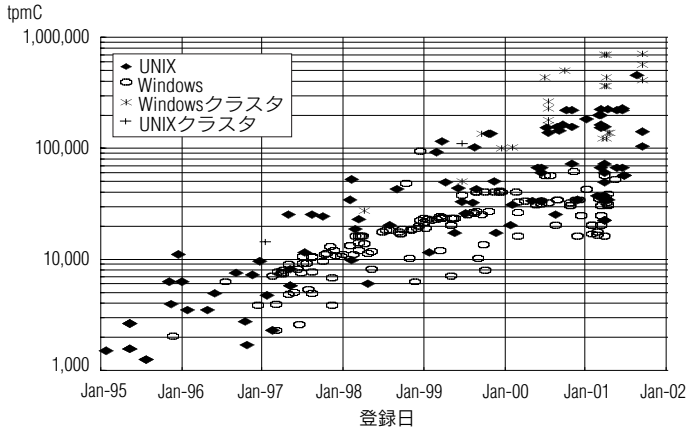

図-3　TPC-C性能値の推移

《評価システム》

評価システムは**図-2**に示すような構成である．これを後述の規定のもとで120分測定し，1分当たりのトランザクション処理数から性能値（tpmC）を求める．

データベース：

- 8種類のテーブルと履歴ファイル
- 性能に応じたデータベース規模（問屋-Wの数を増やす）

応答時間：

- 90%が5秒以下

トランザクションミックス：

①注文（この処理件数で性能を評価）
②支払い（43%以上）
③注文問合せ（4%以上）
④配送（4%以上）
⑤在庫管理（4%以上）

《測定システム構成例》

非クラスタ構成での世界記録である455,818tpmCを2001年8月に登録した時に使用したシステムは39万台の端末が繋がる大規模システムで，ハードウェア構成は以下の通りである．

データベースサーバ：

SPARC64 GP 563MHz×128CPU，メモリ256GB，
RAID 40set（1,624disk 28TB）

クライアント：

PentiumIII 1GHz×2CPU，550MHz×2CPU
合計66set

端末ドライバ：

PentiumIII 700MHz×2CPU×33set
（39万台の端末をエミュレート）

実際の作業では上記の他にDBのバックアップシステム，調査用のミニ環境，予備機等が必要となるので345平方メートルのスペースに上記のほぼ2倍の設備を展開し作業を行った．

《TPC-Cの性能動向》

図-3に性能値（tpmC）の推移を示す．最高性能値は年々2倍程度の伸びを続けており，2001年9月末現在における最高性能値はクラスタシステムで709,220tpmC（8CPU×32ノード構成）．非クラスタシステムでは455,818tpmC（128CPU構成）である．

価格性能比は，システムの性能向上につれて向上しており，TPC-C初期にはおよそ$600/tpmCであったが，価格性能比を主体にチューニングした現在最も価格性能比のよい構成（PentiumIII 1GHz×1CPU）で$4.83/tpmCと約100分の1に向上している．性能主体にチューニングしたトップ性能クラスのシステムであっ

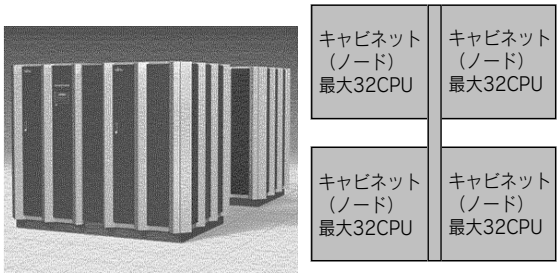

図-4　PRIMEPOWER2000の外観

ても，Windowsクラスタ構成で$13/tpmC～$25/tpmC，UNIXで$28/tpmC～$45/tpmCであり，約10分の1から50分の1に向上している．

測定に使用したシステム

《使用ハードウェアPRIMEPOWER2000について》

PRIMEPOWER2000は，PRIMEPOWERファミリーのカバーレンジを大幅に拡大するモデルである．ブロードバンド時代の中核サーバとして最大128CPU，256Gバイトメモリ，192PCIカードまで拡張可能なSMP（Symmetric Multi Processing）アーキテクチャを採用している．このアーキテクチャにより，ユーザアプリケーションを変更することなく，システム規模に応じた性能を引き出すことが可能である．そのために，最大57.6Gバイト／秒（システムクロック225MHz動作時）の性能を持つ大規模クロスバスイッチを搭載している．

●高いスケーラビリティ

PRIMEPOWER2000は，最大32CPUを実装するキャビネット（ノード）を最大4台まで，高性能クロスバスイッチで接続することで業界最大の128CPUのSMP構成におけるスケーラビリティを実現している．PRIMEPOWER2000の装置の外観，およびノード構成を図-4に示す．

●大容量・高性能メモリサブシステム

最大256Gバイトのメモリをサポートするとともに，最大128ウェイメモリを実現している．また，高性能I/Oインタフェースとして，システムボードあたり64ビット/66MHz PCIスロットを3スロット，64ビット/33MHzを3スロット用意し，高速なI/O接続を可能としている．また，最大構成時は，192PCIスロット構成が使用可能であり，多くの外部インタフェースを必要とする大規模システムにも対応可能である．

●高性能・高信頼のSPARC64 GPプロセッサを採用

CPUとしてSPARC V9アーキテクチャに準拠したSPARC64 GPプロセッサを採用している．本プロセッサは，命令のアウトオブオーダ実行，レジスタのリネーミング機能，大容量キャッシュメモリ，高度な命令分岐予測機能などの先進マイクロアーキテクチャを採用している．また，チップ内蔵キャッシュおよび外部キャッシュをECC（Error Checking and Correcting）保護しており，高い信頼性を有している．PRIMEPOWER2000では，8Mバイトの大容量外部キャッシュを持つ563MHzの動作周波数のSPARC64 GPプロセッサを採用している．また，今後提供するより高い周波数のCPUへアップグレード可能となっている．

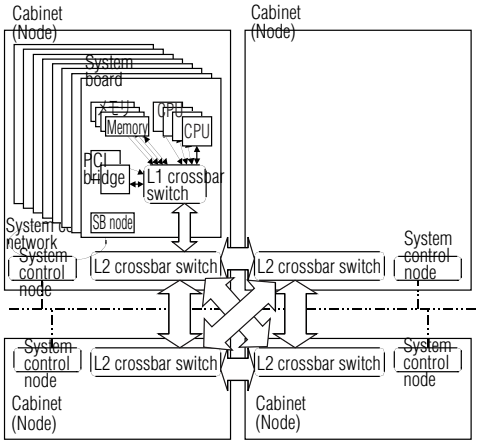

図-5　PRIMEPOWER2000のハードウェア構成

●ハイアベイラビリティ機能のサポート

24時間365日運転を実現するため，主要なコンポーネントの冗長構成および活性交換機能などのハイアベイラビリティ機能をサポートしている．具体的には，システムボード（CPU・メモリ・PCIカードを搭載），電源ユニット，ファン，システム監視機構（システム制御ボード）およびディスク装置で本機能をサポートしている．これらのコンポーネントに故障が発生しても，残ったコンポーネントで業務の継続が可能である．また，活性交換機能により業務を停止することなく故障部品を交換することができる．

●装置構成概要

PRIMEPOWER2000では最大32CPUを実装するキャビネット（ノード）を最大4台まで接続することで最大128CPUを構成可能であり，高い拡張性を有している．

PRIMEPOWER2000のキャビネット（ノード）間を2階層クロスバスイッチで結合している．この2階層クロスバスイッチは，すべてのCPUがすべてのメモリにアクセス可能とし，かつメモリ内容の一貫性を保証している．この2階層クロスバスイッチは1つのクロックソースによって同期動作し，CPUからのアクセス要求を全プロセッサによりチェック可能としている．この動作をスヌープといい，その性能はSMPシステムの基

本性能を示す．本システムでは，最大57.6Gバイト／秒のスヌープ性能を有しており，CPU数の拡大に対してスケーラブルな性能を実現している．PRIMEPOWER2000のハードウェア構成を図-5に示す．

《使用ソフトウェアSymfoWAREについて》

マルチクラスタ機能やホットスタンバイ機能を含む高信頼性と高性能をうたったDBMSであり，大型汎用機，unix系，Windows系上の製品がある．

●トップレベルのOLTP性能

性能については，Windows上とSolaris上の大規模から小規模までのTPC-C登録値を持つ．少し古い登録値もあるが，それぞれ，登録時点でほぼ同規模のハードウェアと比較して遜色ないものであり，DBMSとして大規模から小規模までトップレベルのOLTP性能を持つDBMSであるといえる．これには，以降で説明するようなSymfoWAREの基本的な構造が大いに関係している．

●多重実行を高速にする仕組み　ステージング制御

SymfoWAREはOSによらず，OLTPも含めた多重実行を高速にする仕組みとして，独自のユーザスレッドを使ったディスパッチングの仕組みを持っている．これをステージング制御と呼んでいる．これは，多重実行において必然的に発生する待ち（I/O待ち，ロック待ち，通信待ち等）の発生時にOSのディスパッチ制御や待ち制御を動作させずに，atomic機械命令を直接使用する高速のロック制御と待ち制御を行うというものである（たとえば，応用プログラム1がI/Oを発行した時，その完了を待つことなく，応用プログラム2の処理を行う）．

これはCPU数が少ない時にもOSカーネルやスレッドライブラリ等の持つ汎用の制御に比較して大きな優位性を持っているが，CPU数が多い時にはさらに優位性がある．というのも，OSのディスパッチ制御は，汎用であるが故にCPU間での干渉が大きい．これは，簡単なテストプログラムを作って，OSの各種の同期や排他のためのシステムコール（unix系なら，semop，sema_wait/sema_post，mutex_lock等）を複数CPUで多重に実行してみると容易に分かる．SymfoWAREのステージング制御では，他に実行可能な処理がある限り，これらのOS機能を呼ばないようにしている．また，このような切り換えは多くの場合CPU間の干渉なく行え，もしCPU間の干渉のある場合にも，可能な限り最小限度の干渉（atomic機械命令の実行時間程度に）に止めるように制御している．

●SMP性能にも有効なパーティション分割機能

SymfoWAREは論理的には1つのデータベースのテーブルをAPIを変えずに分割する仕組みを当初からサポートしている．これは元々，性能のためのものではなく，1本のディスクに入らないような大きなテーブルを，そのユーザにとって運用しやすい単位（支店ごと，データ発生月ごと等）に分割するためのものであるが，DBキャッシュのバッファプール分割機能やログ負荷分散機能と連動することも可能で，後述のSMPに向いたCPU間の負荷分散チューニングをする上で大いに有効なものとなっている．

●OLTPに限らない高性能

比較データはないが，TPC-C以外にもDBローダについては世界最高速ではないかと思っている．というのも，TPC-Cで使用するDB（今回の場合約5TBもあるDB）が約一晩でロードできるからである．この理由は，TPCCでは，性能にほぼ比例してDBを大きくする必要があるが，DBロードについてもスケーラビリティがTPC-C処理とほぼ同程度以上にあることにより，DBが大きくなった分DBロードのスループット性能も上がっているためである．

性能チューニングとは

《チューニング作業の実際》

チューニングというと，たくさんあるパラメタの組合せを最適化しているというイメージを抱く人が多いかもしれないが，実際にやっていることはそのイメージとかなり違っている．

まず最初に必要なことは，予定性能を出すために「必要なハードウェア資源を見積もること」である．

- 必要ディスクI/O性能→ディスク数／種類，ディスクコントローラ数，I/Oアダプタ数／種類
- 必要通信性能→ネットワークアダプタ数／種類，ハブ数／種類／性能

次に「取得可能なハードウェアの評価/選定し取得すること」が重要である．この選定の際に単にスループットだけでなく，CPUオーバヘッドやレスポンス等の考慮も必要である．ただし，予算や納期の都合で最もよいと評価したハードウェアを使えない場合もある．

そして「実際の接続とテスト」である．普通は，本番環境と別の小規模な環境を事前に作成した方がよい．動作確認用の最小限のものと，性能測定もできるような中規模環境（本番環境の数分の1の縮小モデル）の両方を作成できれば理想的である．

実際のチューニング作業に入ってからは，「個々のパラメタを最適値ではなく安定／余裕のある値に設定すること」がポイントとなる．最初は個々のパラメタをチューニングするよりも，他のパラメタをチューニング

可能にすることの方が重要だからである．すなわち，ここでいっているのは『最適化をしようとしては駄目だ』ということである．実際のシステムは複雑でパラメタが多く，個々も単純な関数でないので，方程式を立てて解くような具合にはいかない．それどころか，当初は性能に関する重要なパラメタや制約条件のごく一部しか分かっていないというのが普通である．また，このように大規模になってくると，測定や評価にも時間がかかるため，仮に方程式ができたとしてもその変数を決定するのに必要なだけの十分な回数の測定をすることすらできないからである．

そして最後に忘れてはならないのは「問題点／疑問点があれば，ソフトやハードの開発元等に相談すること」である．必ずしもすぐに解決するとは限らないので，他作業と並行してやることが重要ではあるが，開発元であればこそ理解できる現象というものも多々あり，必ずやよい解決法を考え出してくれるだろう．

チューニングを始めようという人へのアドバイスとしては，「まず最初に2重以上のバックアップの確認！！」である．これで少なくとも悪くなることはなくなる．悪くならなければ良くなるしかない．システムが複雑だということは，最適からは遠い状態であるのが普通であり，チューニングでよくなるネタはそこらじゅうに転がっているのである．

《SMPにおけるチューニングとは》

後述の具体的な話をすると，SMPのチューニングが難しいように誤解されそうなので，まずSMPはチューニングしやすいということを強調しておきたい．

システムを構成するプログラム群は元々マルチプロセス（一部はマルチスレッド）で動作するようにできているので，自然に複数のプロセッサを使うのである．したがって，CPU数に見合う『実質』多重度を確保しさえすればよい．以下で述べるようなSMPの性能上の注意点についても，特に性能にとって重要な部分のみを考慮すれば済むのである．これは，たとえば，キャッシュミスのレスポンスが最高速のシステム間通信に比較してもケタ違いに小さいことをみても容易に理解できると思う．

この意味でSMPのチューニングでは特に問題のある部分のみを解消すればよい．問題のある可能性のあるものを大量のデータからふるい分ける上で，以下に出てくる数値例が参考になると思う．筆者の経験では，下記の回数程度になると，レスポンスやロック衝突等を心配／確認する必要があるという意味である．

秒あたりの回数	処理内容
50～200	diskあたりの乱処理I/O
500～数千	SCSIやFCやコントローラのI/O
数百～数万	CPUあたりのシステムコール回数
5,000～数万	イーサネットカードあたりの通信
数万～	システム全体のシステムコール（システムコール種類によって大差）
数万～	CPUあたりのキャッシュミス
数百万～	システム全体のキャッシュミス

── SMPでのチューニングのポイント

チューニングのポイントは以下であり，普通は以下の順に進めていくことになる．

- I/O待ちの削減（これはSMPに限らず重要）
- ロック待ちの削減（スピンロックを含む）
  DBの排他待ちのように簡単に調べられるものだけでなく，ミドルウェアやOS内部の制御上のロックもある
- 実質多重度の確保
- キャッシュミス回数の削減
- TLBミス回数の削減
  通常のアドレスを物理アドレスに変換する時のオーバヘッドの問題

── I/O待ちの削減

SMPに限らず重要なことだが，大規模SMPではスループットが巨大であるため，早いディスクを十分に用意するというだけでなく，以下のようなことに，より注意をはらう必要がある．

- I/Oボードもしくは SCSIアダプタ，SCSIやファイバケーブル，ディスクコントローラあたりの負荷
  装置ごとに大きな性能差はあるが，経験的には，1,000回／装置秒を超えるあたりからレスポンスやオーバヘッド増加等に注意をはらう必要がある
- RAIDを実現するためのオーバヘッドや待ち
  この意味で乱処理（特に乱更新が多い場合）にはRAID5は適さないが，コントローラの種類や設定によってはRAID5でなくても大きなオーバヘッドがある場合があるので注意が必要である．
- I/Oドライバのオーバヘッドや待ち
  できれば，必要なスループットを出すだけの多重実行をした時のI/OあたりのCPU使用量が大きいか少ないかで測定したい．ただし，これが予想外に大きい場合には後述のスピンロックが起きている可能性があるので，単にI/O装置やドライバのみの問題ではない可能性がある．

── ロック待ちの削減

多重度を上げてもCPUが使い切れないという時に疑

うべきなのがロック待ちである．また，CPU数小の時に比較して，処理量あたりのCPU使用量が多いという時には，スピンロックによるロック待ちが起きている可能性がある．

このような時には，プロセスごとに使用するCPUを限定（Solarisでは，pbindやpsrset）した上で，多重度等の条件を変えてCPUの使用状況がどう変わるかを調べる（Solarisならmpstatやlockstat）ことが効果的である．同時にミドルウェア等の性能情報採取機能も利用するとよい．どのプロセスが問題か（もしくは割り込みの延長で動作する処理が問題か）といった切りわけができれば解決につながることが多い．

── 実質多重度の確保

上記で削減されずに残った待ちを考慮した上での，実質多重度をCPU数分確保する．実質多重度は単純にいえば以下のように計算できる．

たとえば，1つの応用プログラムの処理が，10msのI/Oを16回発行し，40msのCPUを使い，かつ他の待ちがないとすると，1多重あたりの実質多重度は以下のように計算できる．

単体の実質多重度=CPU使用時間／（CPU使用時間+待ち時間）
=40ms／（40ms+10ms * 16回）
=0.2

このような場合，1CPUを使い切るのに，最低でも5多重が必要であるが，実際には，I/O待ちが最適のタイミングで解消するとは限らないので，その倍程度の多重度が必要となる．

── キャッシュミス回数の削減

SMPはCPUごとに数MBの2次キャッシュがついているのが普通である．これは，一昔前の大型汎用機の主記憶に匹敵する量であり，一部の処理以外はほとんどキャッシュ上のデータを処理しているといっても過言ではない．大雑把にいってキャッシュミスは1%前後であるのが普通である（科学技術計算等の巨大配列処理のようなプログラムは例外）．

とはいえ，この1%未満の差が大きく何割もの性能差になり得るのでおろそかにはできない．

多くの場合，全体のごく一部のプログラムが必要以上かつ大半のキャッシュミスを発生させているので，これをつきとめられれば対策を考えることができる．

── CPU間の負荷分散

結果的には，SMPではCPU間で負荷分散して処理していることになるが，主記憶が共通であることに加えてOSが自動的にディスパッチしてくれることでこれが楽になっているといえる．

とはいえ，負荷分散の一部にCPUを1つのマシンと考えたLCMP（疎結合のマルチプロセッサ）的な考え方を組み合わせることで汎用的なOSのディスパッチを効率化することができる．

負荷分散には，垂直分散（機能分散とその間のパイプライン）と水平分散（通常の負荷分散）があり，この両方の考え方が利用できる．

垂直分散としては，I/O割り込み処理専用CPUとか，DBMSのデーモン処理（応用プログラムの処理と同期しないバックグラウンド処理と応用プログラムの処理と同期するが，複数の応用プログラムからの要求を別スレッドで実行するものがある）用のCPUを専用にすることが考えられる．

水平分散としては，個々の応用プログラムに対し，使用するCPUを限定することでキャッシュミスを減らすことができる．また，この時，応用プログラムごとに使用する共用の資源（DBのバッファプールやログ等）を限定できればさらに効果的である．なお，完全に専用にしなくても，特定の資源へのアクセス頻度がCPUごとに偏在化するだけで効果がある．

水平分散を行うためにはDBMSの持つスケーラビリティのための機能が有効である．中でも特に有効なのが，DBのパーティション分割機能，バッファ分割機能，ログ負荷分散機能である．

垂直分散や水平分散により，使用メモリが限定されキャッシュミスやTLBミスが削減されるだけではなく，CPUごとに処理対象が限定されることによりロック衝突も削減される場合が多い．

大規模SMPを利用するにあたって

今回，128CPU PRIMEPOWER2000を用いたTPC-C測定作業を行ったが，大規模SMPを利用する場合にシステム構成およびプログラミングをどうしたらよいのかといったことをまとめてみたい．

《システム構成について》

●バススヌープ性能

スヌープ機構を単純に実装した場合，1つのプロセッサがスヌープ動作に入ると，他のプロセッサはスヌープ動作に入ることができず，メモリシステムに対してアクセスすることができない．すなわち，N-CPUのSMPシステムでは，全時間の1/Nしかスヌープ動作をする機会が与えられないことになる．このため大規模なSMPシステムでは，複数のプロセッサが同時にスヌープ動作を行うことができるような設計上の工夫がなされている．このことは『毎秒何万回のスヌープが可能か？』というようなスヌープ性能といった指標でカタロ

グ性能として公表されていることが多く，よりスヌープ性能の高いものを選択する方がよいだろう．

●メモリ性能

大規模なSMPはプロセッサが多数であるが故に，それらに対してデータを供給するためのメモリシステムも相応のスループット性能が必要になる．このためメモリシステム全体のスループットを高めるために，主記憶を多数のメモリバンクに分けて，それぞれのメモリバンクが独立に動作ができるようにしてある．これによって複数プロセッサからのメモリアクセス要求に対して，同時に多数のメモリアクセスを行うことができる．

●I/O性能

大規模なSMPでは，サーバとして運用されるが故にI/Oに対するスループット性能も高いことが期待される．しかし，1つのI/Oアダプタが発揮することができる性能には自ずと上限があり，より高い性能を求める場合には多数のI/Oアダプタを装着できる必要がある．

《プログラミング上の留意点》

SMPシステムは共有メモリ型並列計算機である．したがって，その計算パワーを真に取り出すためには，マルチプロセスやマルチスレッドによる並列プログラミングを行う必要がある．しかし，小規模なSMPシステムの時にはほとんど問題とならなかったことが，大規模SMPシステムではいろいろと問題になる場合がある．ここでは，それらのうち典型的なものをあげる．

●メモリ共有領域のアクセス

共有空間に対するメモリアクセスを無秩序に各プロセッサが行った場合，キャッシュ競合のため性能が劣化することがある．多数のプロセッサのキャッシュが共有状態になっているとき，どれかのプロセッサがデータの更新を行うと，他のプロセッサのキャッシュは無効化されるので，次のアクセス時にミスヒットとなってしまう．このためキャッシュのヒット率が下がり，性能の劣化要因となる．これを避けるためには，多プロセッサが参照するメモリ領域（キャッシュライン）を極力更新しないように注意すべきである．

●排他ロック

マルチスレッド機構やプロセス間共有メモリを用いてプログラムを作成する場合，共有データアクセスの際に排他ロックをとることになるが，この排他ロックの取り方にも注意が必要である．排他ロックをとっている区間では，1つのプロセッサしか動作できないので，性能を出すためには極力ロック期間を短くすべきである．また，同一ロックを複数のプロセッサが取り合ってしまうと性能が劣化するので，排他ロックは基本的に衝突しないように使用すべきである．たとえば，大きなテーブル全体を1つの排他ロックで対応するのではなく，個別のエントリごとに排他ロックをとるといったような細分化が必要である．

他にも，システムで提供されている排他ロックの実装にもよるが，ブロッキング型のロックの場合は，ロックがとれないときには，プロセッサを解放してスリープするようになっている場合が多い．しかしいったんスリープすると，当該スレッドが次にプロセッサを獲得して動作を開始できるまで，相当の時間待つことになってしまう．このことが全体の性能を下げる要因になるので，ノンブロッキング型のロック関数を使って，ロックがとれないときにはロックを必要としない別の仕事をするといったような工夫が必要である．

●使用するシステムコール

プログラムには必ず何らかの入出力があるので，このためにシステムコールを使用することになるが，同等の機能を実現するシステムコール群のうちどれを使用するべきなのか？　またどのような使い方をするとよいのか？　といったことを検討する必要がある．

大規模なSMPシステム上では，小規模なシステムでは想像もしなかった現象が起こることがある．使用するシステムコールがプログラムの必要とする性能を得られるのかどうかを，同じような呼び出しパターンとなるような小規模なテストプログラムにより確認するべきであろう．また，性能が得られない場合には，その原因がどこにあるのかを各種の統計情報を得るためのツールを使って調査し，どのような使い方を採れば問題が解決するのかを検討すべきである．

●各種性能調査ツールの利用

UNIX系OSであれば，従来からvmstatやiostat，sarといったカーネル内部の統計情報を取り出すコマンドが用意されている．また，我々の使用したSolaris環境ではmpstatといったプロセッサごとのCPU使用率やページングの状況といった情報が取り出せるコマンドやlockstatというカーネル内ロックの統計情報を出力するコマンドが用意されている．また，最近のマイクロプロセッサには，キャッシュのヒット率を調べられるといった性能調査用の機構（PA機構）を持っているものが多く，この情報を取り出すことができるツールが提供されるようになってきている．SolarisもSolaris8になってからは，cpcと呼ばれる機能モジュールの追加によって，cpustatやcputrackというツールが提供されている．これらのツールを使って，キャッシュミスやTLBミスといったプロセッサレベルでの性能劣化要因がどの程度でているのか？　また，プログラムを書き換えたときに，それらがどう変化したのかを調査することも，

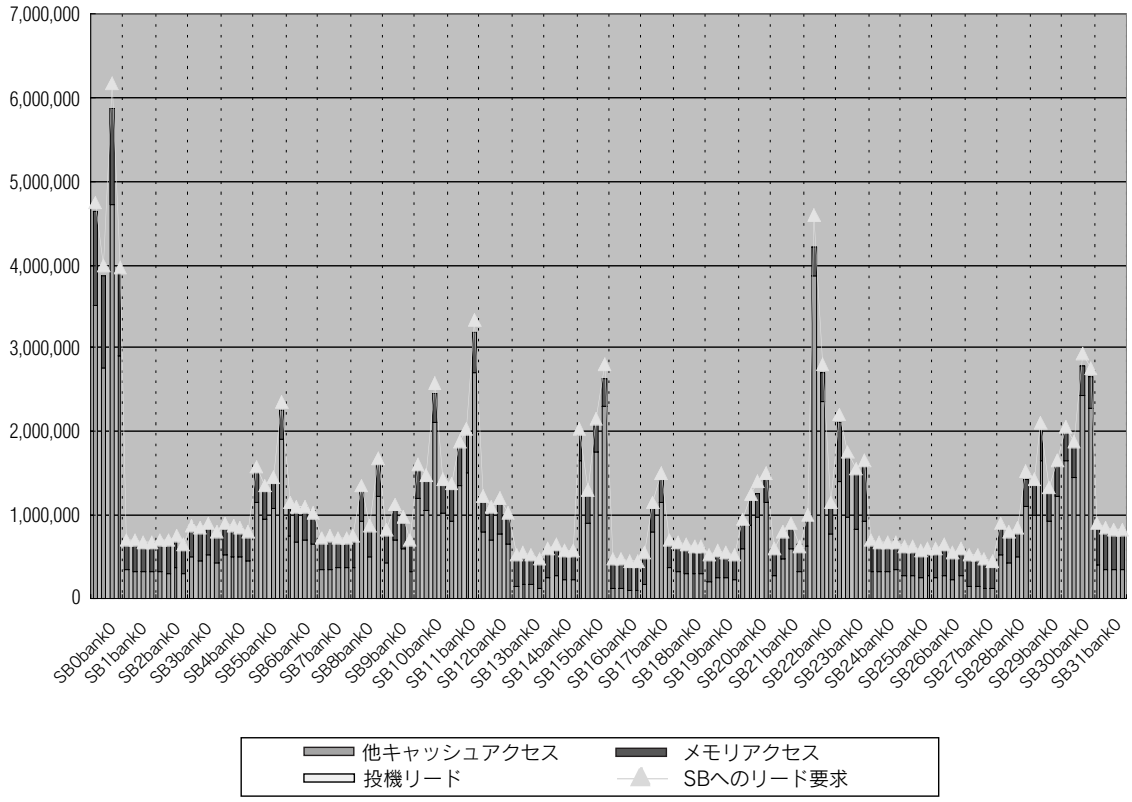

図-6　メモリバンクの使用頻度

これからのプログラムチューニングには必要なことだろう．

また，最近のプログラム開発ツールでは，このPA機構を利用してプログラム中のキャッシュミスヒットが多い命令を調べるといった機能を持ったものもある．こういったツールを駆使して，プログラムの性能的な問題点を明らかにしていくことも，今後は必要になってくるものと思われる．

多くのSMPシステムでは，システム内部バスの中のバストランザクションがどのようになっているかといった性能調査のためのハードウェア機構が埋め込まれている．我々の使用したPRIMEPOWER2000にも，システムボード内およびクロスバ機構内にそのようなPA機構が含まれており，これらの統計データを取り出すためのツールを開発した．これらを使ってメモリアクセスにホットスポットがないかといった調査を行い（**図-6**），チューニングの一助とした．大規模SMPシステムにおいて，今後もこのようなツールがより必要となってくるだろう．

今後の展望

我々は，今回TPC-Cにおける非クラスタシステムでの世界最高性能値455,818tpmC（2001年9月末現在）を実現した．現在，TPC-C性能値における最上位はクラスタシステムが占めているが，現実に運用されているデータベースシステムはSMPシステムが大半を占めている．IBMやSunといったコンピュータメーカからも新しい大規模SMPシステムが発表されており，今後もSMPシステムをデータベースシステムに適用することの重要性が失われることはないだろう．

参考文献

1）TPCのホームページ（http://www.tpc.org/）．
2）雑誌FUJITSU（http://magazine.fujitsu.com/），2000-7月号特集，インターネット時代の中核サーバ「PRIMEPOWER」．

（平成13年10月1日受付）