Vol. 37 No. 8　　情　報　処　理　　Aug. 1996

解　説

計算機ベンチマークの最新動向

# 5. メインフレームの性能評価手法†

榊　　幹　　雄††

## 1. はじめに

最近の急速なプロセッサ・テクノロジーの進歩により，メインフレームの世界にも1チップCMOSプロセッサが導入され，並列処理が実用化される時代になった．そしてもはや「メインフレーム」という言葉自体が適切ではなく，むしろ集中サーバあるいはコーポレート・サーバと呼ぶべき時代になっている．

しかしここでは従来どおりワークステーションと区別するために「メインフレーム」という言葉を使用し続けることにする．

本章ではメインフレームの性能評価について概説する．

メインフレームの性能評価で一番重要視されるのは信頼性もしくは正確性である．正確なパフォーマンス予測が必要とされる背景は，メインフレームの使用環境に起因する．メインフレームでは主として企業の基幹業務を処理しているため，性能見積りの誤りが基幹業務処理を遅らせ企業活動に悪影響を与える可能性があるからである．

また，高性能のメインフレームは1台あたりの単価が高いため数十%の誤差が数億円，場合によっては数十億円の価値と等価になるためでもある．したがってメインフレームを購入する顧客の立場に立てば，新システムのパフォーマンスは可能な限り正確に知りたい，できれば数%以内の誤差で知りたいという強い要求がある．近年CMOS技術による並列型メインフレームが主流になりコスト・パフォーマンスが数倍改善されたためパフォーマンス要求に少し余裕が出てきたが，それでも次期機種の性能評価は正確であるに越したことはない．

一方，メインフレームの業務内容には連続性がある．つまりシステムが変わっても業務内容は連続している，または多くの部分が連続している傾向にある．したがって業務内容の方向から解析すれば，ある程度連続領域での予測としてとらえることが可能である．そこでこのような考え方に基づいて性能評価を行っているIBM社のLSPR (Large System Performance Reference)手法を参照しながらメインフレームの性能評価手法について記述を試みる．

## 2. システム性能向上の手法

一般的にメインフレーム使用者が現行使用システムの限界に遭遇し，または限界を予測した場合，そのシステム性能を向上させる手段として以下のケースが考えられる．

1) 構成要素をより速いものに取り替える
CPU，ディスク，チャネル，通信回線などを高速にする．
このうちCPUの高速化とディスクの高速化の効果が大きい．

2) 必要なデータをCPUにより近いキャッシュ（cache）内に蓄える
外部キャッシュ（L2/L3キャッシュ）よりはCPU内部キャッシュ（L1キャッシュ）
主記憶よりはCPUキャッシュ（L1/L2/L3キャッシュ）
ディスクよりはディスク制御装置内のキャッシュ
ディスクへのページングよりは大容量主記憶・拡張記憶へのページング
スワップよりは仮想スワップ（スワップの手続きは行うがスワップしない）

3) 平行処理の利点を使用する

† Performance Analytic Method for Main Frame Computer by Mikio SAKAKI (Mgr. Parallel Competency Center, Systems Laboratory, IBM Japan Corp.).

†† 日本IBM(株)システム研究所並列システム研究室室長

CPU 処理と I/O 処理を重複させる
I/O ドライバが行う処理を付帯チャネル・エンジンにオフロードする
複数 CPU を使った多重処理を行う
入出力装置へ複数本のパスを設定し，平行的に I/O 処理を行う
ディスク・アクセスを複数ディスクに分散させる
Vector 処理を行う（技術計算）
通信回線を多重化する

4) 入出力時間を短くする
I/O 回数を減らす（ロジック変更などによる）
より高速な装置を使用する
チャネル数を増やす
ディスク制御装置のキャッシュを強化する（大きくする）
I/O ブロック・サイズを大きくする
ディスク・アームの移動を最適化する（sort では実施済み）
データをメモリ内にキャッシュする
バッチ・パイプ処理により，中間ワークファイルにメモリを使用する

5) 使用頻度の高い機能をチューンアップする（80－20 ルール）
使用頻度の高い機能を重点的にチューンアップする方法が最も効果的

6) スループットとレスポンス・タイムのバランスを変更する
レスポンス・タイムを犠牲にしてスループットを上げる

以上の手法のうち，複数個の組合せにより高速化を図るのが一般的である．

## 3. 評価基準の設定

次のステップとして，選択した高速化手法の投資金額と向上性能を見積もる．一般的に入出力装置やチャネルなどは簡単なテストデータがあれば性能見積りがやさしく誤差も小さくて済む．他方 CPU の取り替えによる性能評価は，投資金額が大きいゆえに慎重にかつ正確に行う必要がある．

性能評価基準として最も広く使用され，かつ最も単純なものとして MIPS がある．

MIPS＝1/((cycles/instruction)
　　＊(cycle time))

MIPS：Million Instructions Per Second
cycles/instruction は 1 命令の平均実行サイクル数

MIPS は単純で分かりやすい反面，実務処理能力を見積もるには，あまりにもおおまかすぎる．実務処理では使用オペレーティング・システムや I/O 機器により性能差が現れるが，MIPS ではそこまで反映できない．使用者の立場に立てば，処理能力の評価基準としては単位時間あたりのトランザクション処理量またはジョブ処理量が望ましい．この評価方法をＥＴＲと呼ぶ．

ETR＝処理ジョブ数/測定時間　または
ETR＝処理トランザクション数/測定時間
ETR：External Throughput Ratio

ただし，この ETR 計算式では CPU 使用率を計算に入れていない．つまり測定時の CPU に余裕が残っているかどうかが分からない．比較条件を一定にするためには，同一 CPU 使用率で測定すべきであるが現実的には不可能である．

次善策として CPU 使用率を 100%に換算して比較すればよい．つまり ETR を CPU 使用率 100%に正規化(Normalize)すればよい．これを ITR と呼ぶ．

ITR＝ETR/(CPU 使用率)
ITR：Internal Throughput Ratio

なお ITR が意味を持つためには，OS が CPU 使用率 100%に耐えられる性能を持っていなければならない．一般的にメインフレーム用の OS は十数年の改良を経て CPU 使用率 100%まで使用できるように Dispatcher, Paging, Swapping, ResourceControl などの機能強化を行っている．IBM 社では ITR を自社メインフレームの評価基準として約 15 年間使用し続けており，その正確性では実績を積んでいる．

次章で ITR 手法の特長を述べる．

## 4. 評価目標

性能評価は使用者によりその目的・目標が違ってくる．一般的にコンピュータ・メーカは各コンポーネントの性能を知るために単体機能の測定を優先するが，使用者側では CPU・DISK・OS ごとの単体性能ではなく自社アプリケーションの実行性能を知りたいと思う．カストマ・サティスファ

表-1

| | MIPS | ITR |
|---|---|---|
| CPU，メモリ | * | * |
| I/O 装置 | | * |
| OS | | * |
| DBMS | | * |
| 通信ソフトウェア | | * |

クション（顧客満足度）の観点からすれば，単体性能の測定結果はメーカ内部に留めておき，ユーザにはアプリケーションの実行性能を提示すべきであると言える．

経験的に，メインフレーム購入時には以下の性能評価要求が発生する．

1) メインフレーム・システムの正確な性能を知りたい

- ●新規システム導入時の全体的な負荷予測
- ●現行業務移行時の主要アプリケーションの正確な負荷予測
- ●業務の拡大および新規アプリケーション追加時の負荷予測

2) 新規システムの長期運営計画を知りたい

- ●何年もつか（S/W および H/W）
- ●何年後にどのような強化をすればよいのか

メインフレームが1チップCMOS並列計算機への移行が進み，設置場所での自由な性能強化が可能になってはいるが，長期計画に対する要求は依然として強く，新規システムの性能見積り要求は根強く残る．

前章でITR方式がメインフレームの性能評価に有効であることを述べたが，それはシステム内のすべての要素を含めて測定できるからである．MIPSはCPU性能のみを測定対象としている．両者の測定対象を**表-1**に示す．

ITRは通信回線の測定は含んでいない．これは通信回線の遅延は大きいため別途見積りする場合が多いからである．

ITRはOSやDBMSを含んだ形で性能評価を行うため，使用者には分かりやすいが，メーカには測定ケースが増加するため大規模な測定作業が必要になる．

評価目標としては可能な限り実環境に近づけたい．そのため，ITR方式でCPU・OS・DBMSの代表的組合せを評価目標に定める．IBM社はこの評価目標に従って各種の性能測定を実行している．

## 5. 評価技術とその展開

LSPRパッケージの技術的背景は，メインフレーム型アプリケーションは連続的変化型であり，したがって要素分解と合成手法による性能予測が可能となる点である．またこの手法は実用化できる程度まで精度が向上している．要素分解と合成手法を簡単に説明すると以下の4ステップの仮説から構成されている．

仮説1) ユーザ・アプリケーションは種々のサブシステムの集合と見なす

例：バッチ，RDBMSによるOLTP，階層型DBによるOLTP，プログラム開発・テスト，など

仮説2) アプリケーションごとに主要サブシステムを特定できる

1つのアプリケーションは項目1のサブシステムに割当て可能である

仮説3) サブシステムに要素分解して測定することが可能である

各サブシステム単独で性能測定することはやさしい

仮説4) 実環境は複数サブシステムの加重平均で近似可能である

各サブシステムは互いに独立と考えられ，性能はそれらサブシステムの1次結合になる

$$\text{評価性能} = \sum_{i=1}^{n}(A_i * S_i)$$

ただし $A_i$ はサブシステム $i$ の割合

；$\Sigma(A_i)=1$

$S_i$ はサブシステム $i$ の性能

上記仮説をLSPRの15年間の使用経験で検証してみれば，現実に合致していると見なせる．

## 6. サブシステムの計測手順と性能評価手順

各要素であるサブシステムの測定は単独で行うが，測定にあたっては反復性をあげるために守るべき点がある．

1) サブシステムは単独で実アプリケーションを使用して測定する

人工的なカーネルより実アプリケーションの方が業務担当者にとり意味を持つ．実アプリケーションの選択に当手法の成否がかかってくるため，

幅広く複数のユーザから業界を代表するタイプのアプリケーションを選択する．

2) 実アプリケーションはコンピュータの世代交代を考慮する

実アプリケーションは作成された世代とその次世代では意味のある結果を出すが3世代目になると意義を失う場合が多い．それはCPU，メモリ，ディスクなどの性能向上が大きいため，古いアプリケーションでは現実に即さなくなるからである．その場合には測定用サブシステムを再構築しなければならない．

通常は既存サブシステムはそのまま測定を続行し過去の機種との連続性比較に使用する．そして新サブシステムを追加し新しいアプリケーションをメンバに加える．この新サブシステムは当世代とそれ以降の世代との比較に使用する．

3) 測定条件は各コンピュータの世代を通じて一定になるよう最大限の努力をする

I/Oボトルネックは避ける．

I/O装置も同じ物を使用し，新機種がでた場合には再測定も考慮する．

CPU使用率は同一条件で測定する．

例：CPU使用率70%と90%の2点で測定する

4) 測定結果をCPU使用率100%に正規化する

CPU使用率100%に正規化することにより，そのシステムの潜在能力を検出できる．

正規化した値を各サブシステムの能力とする．

以上の手順により各サブシステムの処理能力が計測されるが，その前提条件としてはソフトウェアはシステムの成長とともにチューンアップされ続けていることとする．さもないとCPU使用率100%に正規化した値を達成できなくなる．

一例をあげればタイムシェアリング・システムにおいて，Loginユーザが増加して1000人以上に達するとLarge System Effectという現象が発見されたことがある．これはDispatcherがReady Taskを探すチェインが長くなりそのためのオーバヘッドが5%程度になったためである．OSはその対策としてReady Taskのみをつなぐチェインを新設しオーバヘッドを解消した経験がある．

メインフレームの性能評価手順は比較的単純であり，それゆえに高い信頼性を実現できている．それはメインフレームの現行アプリケーションを基準点として，新システムでの性能を見積もる手法をとるからである．その手順を示す．

1) 現行機種で各サブシステムの比率を測定する

CPU使用量，I/O処理量などを標準ツールで計測する

例：バッチ　30%
　　RDBMSのOLTP　40%
　　プログラム開発　20%
　　その他　10%

2) 能力改善率をサブシステムごとに計算する

現行機種と新機種の性能比はサブシステムごとに異なるのが普通である

例：バッチ　2.1倍
　　RDBMSのOLTP　2.3倍　など

3) 新システムの処理能力を計算する

サブシステムごとの改善率の加重平均をとる

例：バッチ＋OLTP＋開発＋その他

30% * 2.1倍＋40% * 2.3倍＋20% * 2.2倍＋10% * 2.1倍＝2.2倍

4) 新規アプリケーションを追加する場合には補正する

新規の場合には比較すべき現行アプリケーションがないため，アプリケーションの割合を再計算して項目1を作りなおす．

## 7. パフォーマンス計測領域

実環境ではCPUの種類だけでも3世代4世代考慮するため約120種類になる．サブシステムは約10種類が必要になり，またオペレーティング・システムは複数リリースが要求されるため9

**表-2** バッチ

| | |
|---|---|
| 技術計算1 | 重い技術計算 |
| 技術計算2 | 大学コンピュータ・センタの技術計算 |
| 事務計算1 | RDBMSが少ない事務計算 |
| 事務計算2 | RDBMSが多い事務計算 |
| 単純事務計算 | ファイル更新処理を中心 |

**表-3** オンライン

| | |
|---|---|
| TSS | プログラム開発など |
| CICS | 簡易型オンライン，RDBMS少し |
| DB2 | RDBMSのオンライン |
| IMS1 | 階層型DBのオンライン1 |
| IMS2 | 階層型DBのオンライン2 |

表-4

| 対象機種 | 対象OS | OSの種類 |
|---|---|---|
| CMOS並列機用 | MVS/ESA | 3種類 |
| 大型機種用 | MVS/ESA | 3種類 |
| 中型機種用 | VM | 2種類 |
| 小型機種用 | VSE | 1種類 |

種類になる．(96年7月現在)

すべてのケースを網羅するとなると120＊10＊9＝10,800ケース程度の測定が必要となる．これは実質的に不可能であり，CPUモデルごとに必要とされる領域のみを測定する．具体的に計測領域を検討してみる．

1) サブシステム

サブシステムは大別してバッチとオンラインからなる

2) オペレーティング・システム

オペレーティング・システムは複数リリースを考慮しなければならないため**表-4**の9ケースが必要となる．

3) CPU機種

CPUが複数個搭載可能システムではCPU個数ごとに測定する．それは，CPU個数の増加と性能向上は正比例ではなく緩やかな成長曲線になるからである．

さらに信頼性を増すためにSMP構成を2台接続したマルチプロセッサを構成できるため，測定ケースが増加する．たとえば10CPUモデルでは合計14種類の機種がある．このうち類似パフォーマンスになるケースは測定せず論理的推論で計算する場合もある．

上記の計測領域のうち，優先度の高い分野を特定する．たとえば，新プロセッサには新OSを対応させ，古いOSは測定しないとか，大型機では大型機用のOSのみを測定する．また旧機種は前回の測定データを可能な限り流用し，計測領域を最小労力で埋めるように努める．このようにすれば，新機種および新リリース発売時の計測領域は数十カ所から百カ所程度に収まる．

## 8. 性能評価ツールの作成

上記の計測領域をすべてハンドブックにして保持し，必要に応じて1つずつ手作業で計算すれば新システムの性能評価が可能となる．

また，計測領域テーブルは大きくなるが性能評価計算は簡単であるため，PC上の表計算プログラムを使用した性能評価ツールを構築し対話的に情報を入力し計算することも可能となる．

実際IBM社ではLSPR(Large System Performance Reference)と呼ぶパッケージを作成して自社内配布している．LSPRはハンドブックとPCツールの2種類のフォーマットで社内配布されている．

PCツール版では，使用者がPC上で対話的に現行機種，現行アプリケーションとその比重%をマウスで選択した後，希望機種を4個まで指定すると性能評価値を計算しグラフ表示する．また詳細レポートも印刷可能になっている．このようにしてユーザは新機種の導入前にその処理能力を自社のアプリケーションに即した形で知ることができるようになっている．

このようなアプローチはメーカ固有の方式であり汎用性が低い反面，性能評価値は非常に正確になる．経験的にはバッチで誤差数%以内，RDBやOLTPで誤差10%以内に収まっている．(10%は保証値ではなく経験値であるが，実際にはOLTPでも数%程度に収まる場合が多い)

## 9. まとめ

メインフレームの性能評価手法をIBM社のLSPRを参照しながら述べた．当手法はメインフレーム環境の特性を十分理解した上で正確な性能予測を実現可能にした点において評価できる．ただし高い信頼性を維持するためにメインテナンス・コストも高くなる特性を持つ．

メインフレームのアプリケーションの連続性と相互独立性をうまく利用し，要素分解した状態で要素別に性能予測を行い，最後に統合することにより信頼性を上げている．

特にRDBやOLTPでの誤差を他の方式よりは小さくできる手法である．連続領域に存在するメインフレーム・アプリケーションに特化したゆえに達成された信頼性であるとも言える．特に業務系ではアプリケーションの連続性・類似性が高いためこの分野で威力を発揮できる．クライアント・サーバシステムも業務系になれば同様の連続性や類似性が出ると思われ，応用可能な手法であると思われる．

(平成8年2月7日受付)

**榊 幹雄**

昭和45年京都大学工学部数理工学科卒業．昭和48年日本アイ・ビー・エム（株）入社．以来メインフレーム・コンピュータの本社部門技術センターに従事．メインフレームのソフトウェアおよびハードウェア性能の比較研究を行うとともに新製品を日本国内に紹介・普及する職務に従事．途中3年間米国開発部門にて大型コンピュータの開発に従事．現在，システム研究所，並列研究室室長．