北海道大学工学部研究報告
第101号（昭和55年）

Bulletin of the Faculty of Engineering,
Hokkaido University, No. 101 (1980)

大型計算機センター・システムの性能評価

栃内　香次*　　石井　　裕**　　永田　邦一*

（昭和55年6月30日受理）

Performance Evaluation of the University Computing Center System

Koji TOCHINAI, Hiroshi ISHII and Kuniichi NAGATA

(Received June 30, 1980)

Abstract

This paper describes the performance evaluation of a computer system in operation at a university computing center. In a university computing environment, there exists a wide variety of computing demands. And they increase yearly with markedly high seasonal fluctuations. Therefore, it is important to evaluate the performance of the system and employ the best operation policy.

In this study, we investigated a method for performance evaluation based on simulation technique. The method is as follows :

1) to perform bench-mark tests and obtain characteristic data of the system to be evaluated,

2) to build a macro model of the system using measured data, and

3) to carry out simulation under various operation conditions and workloads.

We applied this method to the FACOM230−75 computer system installed at the Hokkaido University Computing Center during 1974 to 1979. And the results indicated the general applicabillity and effectiveness of the method.

1　はじめに

計算機システムの性能評価が行われる目的はさまざまであるが，大別すれば次の3項目に分けることができる。[1]

（1）機種の選定……新しく計算機システムを導入する際，又は機種更新の際に行われる。

（2）性能の改善……運用中のシステムについて，最適の運転条件を見出し，あるいはハードウェア，ソフトウェアの問題点を発見し，改善するために行われる。

（3）新機種の開発…新しいシステムを設計するための事前評価として行われる。

一方，評価を行う立場についても，利用者側と製造者側の両面がある。一般に上記(1)，(2)は利用者の側で，また(3)は製造者の側で行われる評価である。

ここでは，計算機システム利用者の立場から，現に設置され稼動しているシステムの動作状態を把握し，性能を改善する目的で行われる性能評価を考える。稼動中のシステムの性能が運転

*　電子工学科　電子機器工学講座
**　日本電信電話公社横須賀電気通信研究所

条件を最適化するのみで十分改善できることは少なく，ソフトウェアの改良やハードウェア構成の変更，増設が必要な場合が多い。したがってこのような性能評価の結果は新機種の検討へとつながる面をもち，上記(1)，(2)を明確に区分することはできない。

一般に，性能評価が重要になるのは比較的大規模な計算機システムにおいてであり，そのようなシステムは多数の利用者に共同利用される計算センターの形態で運営されるのがふつうである。それゆえ，ここでいう「利用者」は個々のエンドユーザではなく，センターの管理・運営にあたる者を指す。大学の計算センター，ことに北大など全国7大学に設置されている大型計算機センターはこのような形態で運営されるセンターの一つの典型であり，かつそこで処理される計算ジョブは質，量ともに非常に広範囲にわたり，システムの運転条件は複雑である。したがって，適切な性能評価によってシステムの状況をつねに把握しておくことが重要であり，評価手法もまたこのような環境に適合するものでなければならない。本論文では，われわれがこれらの観点から北大大型計算機センターの計算機システムを対象として行ってきた性能評価手法の検討とその結果について報告する。なお，対象とした計算機はFACOM230−75（稼動期間，1974年11月～1979年9月）である。

よく知られているように，大型計算機センターで処理されるジョブの規模は極めて広範囲に分布し，しかも分布の両端部（小規模，大規模）にあたるジョブが多い。年間計算需要には大巾な季節変動があり，図1に示すように卒業研究のための計算が集中する毎年12月～2月と，それらが一段落する4～5月との比は2倍強に達する。また，システムはバッチ処理，TSS処理の両形態で運転されるが，一方を優遇することなく，両者間のバランスを保つことが重視される。さらに，計算需要は年々大巾に増加し，北大大型計算機センターの場合，FACOM230−75が設置されていた期間中の計算件数は図2に示すように年率平均20%近い伸びを示している。一方，ジョ

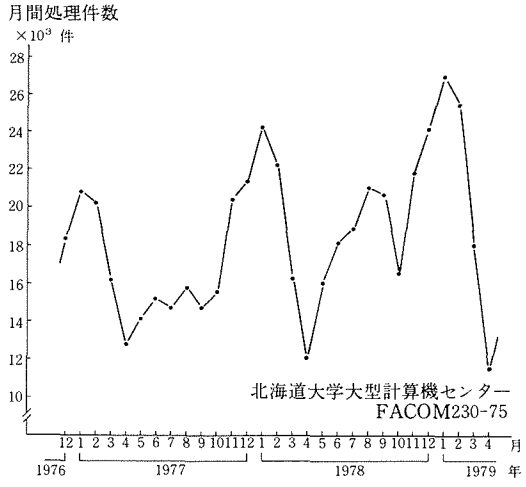

図1　計算需要の季節変動

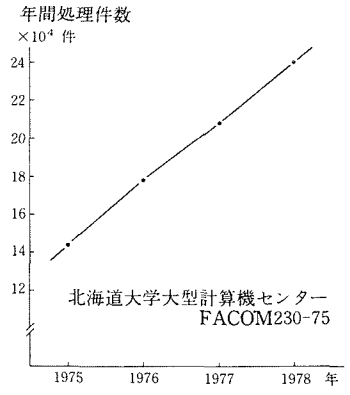

図2　年間処理件数の伸び

ブの処理形式，使用言語などは比較的単純かつ限定されており，大多数のジョブはコンパイル～実行型で処理され，その使用言語はFORTRANが大部分である。

われわれがここで取扱う北大大型計算機センター・システムの性能評価は以上のような環境の下で行われ，以下の諸点を主要な目的としている。

（1）バッチ処理，TSS処理の混在する状態の下で最適の運転条件を見出すこと。

（2）システム内で性能向上の隘路になっている要素を発見すること。

（3）想定される計算需要増にどれだけ耐えられるかを推定し，機種更新計画等それに対処する方策立案のための資料を得ること。

われわれはこのような目的に適した性能評価手法として，ベンチマーク・テストによる実測と，システムのマクロモデル化にもとづくシミュレーションとを組合せる方法を開発し，有効性を確認したのでここに報告する。以下，第2章ではわれわれが対象とする大型計算センター・システムの運転環境と性能評価指標について述べ，第3章でこれにもとづくシステム・モデルの構成を述べる。第4章ではシミュレーション結果のいくつかについて報告し，システムの運転計画，構成変更の検討などに有用であることを示す。

2 大型計算機センター・システムの運転環境と性能評価

2．1 センター・システムの運用条件

前述のように，大型計算機センターは多数の利用者を擁する共同利用組織で，極めて長大なものからごく小規模なものにわたる多量のジョブが毎日処理されている。それゆえ，計算機システムをできる限り効率よく運転すること，およびその程度をはかる手段である性能評価が重要な課題となっている。このような局面で行われる性能評価において最も関心がもたれる評価指標として以下の4項目があげられる。

（1）バッチ処理における，ジョブ処理のスループット。

（2）同じく，各ジョブのターンアラウンドタイム。

（3）TSS 処理における，コマンド処理のスループット。

（4）同じく，各コマンドのレスポンスタイム。

ここで(1), (3)はジョブまたはコマンドの時間あたり処理量，すなわちシステムの総体としての性能を，また(2), (4)は個々のジョブまたはコマンドに対するシステムの応答，すなわち利用者からみたシステムの性能をそれぞれ表わす量である。

ジョブおよびコマンドはシステムに処理を要求する負荷であり，システムを構成する諸資源を競合して使用しつつ処理が進行する。通常，これらはバッチ処理，TSS 処理という異なった制御方式の下に処理され，したがってシステムの負荷はバッチジョブ，TSS コマンドの2群に大別される。それゆえ，システムをバッチ処理中心に運転すれば TSS 処理能力が低下してレスポンスタイムがのび，TSS 処理中心に運転すれば逆にターンアラウンドタイムがのびる。したがって両者をバランスさせる最適な運転条件を見出すことが必要になる。しかしながら，バッチ処理における「ジョブ」と，TSS 処理における「コマンド」とは異質な負荷であり，両者を統合して表現できる評価指標は存在しない。すなわち，バッチ処理，TSS 処理の割合を定めてある運転条件を設定したとき，それが目標を実現しているか否かを判断するのは困難である。

2．2 ベンチマーク・テストによる性能評価

稼動中のシステムを対象とする性能評価手法として，ベンチマーク・テストがよく知られている。これは，性質のわかっているテストプログラム群を実際にシステムに投入して処理し，所要時間，ターンアラウンドタイム等を測定するものである。このとき，測定ツールとしては課金等のためにシステムに組込まれているいわゆる会計ルーチンが利用でき，特別なツールがなくてもある程度の性能評価を行うことが可能である。このように，ベンチマーク・テストは比較的容易に実施でき，かつ精度も高いものであるが，以下に示すようにいくつか不便な点がある。[2]

（1）日常の処理実態に合うように，テストプログラム群の構成には十分な注意が必要であり，

手間もかかる。

（2）センターで処理されるジョブの規模，構造は時間とともに変化するので，テストプログラムのアプデートが必要であり，かつ以前の結果との連続性を保つ必要もある。

（3）テストは現在設置されているシステム構成の範囲内に限られる。

（4）1回のテストには準備を含めかなりの手間と時間を要し，多数の運転条件を設定してテストをくり返すのは容易ではない。

（5）測定されるのはシステムの総合的な性能を表わす評価量であり，システム内各構成要素のふるまいを詳細に示すものではない。

以上のような問題点はあるが，ベンチマーク・テストによる性能評価は計算システムの利用者にとって，以下に示すように極めて有用な手法である。

（1）利用者にとって関心の強い，システムの総合的な性能を表わすのに適したデータが得られる。

（2）日常の業務運転において運転記録として得られる処理状況データとの整合性がよく，比較しやすい。

（3）特別な評価ツールがなくても実施可能である。

2．3　北大大型計算機センターにおけるベンチマーク・テスト

北大大型計算機センターでは，機器の変更，増設，OSその他ソフトウェアのレベルアップ等の評価のために，しばしばベンチマーク・テストを行っており，何種類かのテストジョブが用意されている。これらはcpu時間，メモリ使用量，入出力回数などの分布が日常の業務運転時のそれに一致するように選ばれている。表1はこのうちの1種についてその諸元を示したものである。このテストジョブは本研究において主に使用したものであり，各ジョブのcpu時間の分布は図3

表1　ベンチマーク・テスト用ジョブ

項目＼ジョブステップ	全ジョブ	ジョブステップ－1	ジョブステップ－2
処理所要時間	69.1秒	——	——
CPU使用時間	23.0秒	1.3秒	21.7秒
メモリ占有時間	60.6秒	15.3秒	45.3秒
メモリ占有量	——	40.0K語	35.3K語
入出力回数	111.6回	28.3回	83.3回

テストジョブの個数：100件，表中の値は1件あたり平均値
ジョブステップ－1：コンパイル及びリンク（"COMPLIED"）
ジョブステップ－2：実　　行　　（"RUN"）
使用システム，OS　：FACOM230-75,Monitor-Ⅶ
設置場所　　：北海道大学大型計算機センター
測定日時，条件　：1978年11月21日，多重度1にて測定

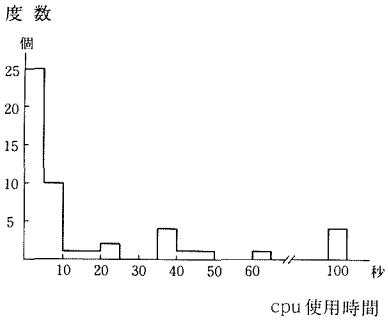

図3　テストジョブのcpu使用時間分布

のようになっている。前述のように，われわれが評価の対象とする計算機は1974年11月から1979年9月まで稼動したFACOM230－75システム（OS：Monitor-Ⅶ）であるが，このシステムには1975年より資源使用状況を測定するツールが組込まれており，[3]このツールと会計ルーチンおよび運転記録から得られるデータを用いた。

Monitor-Ⅶの下で，バッチ処理は図4に示すような形で行われる。ここでジョブの投入と出力結果の取出しは利用者自身の操作に委ねられ，これを円滑に行うために大容量のプリスタックファイルおよびシステム出力ファイルを置いている。それゆえ，ベンチマーク・テストではジョブがプリスタックファイルからシステム入力部に投入された時点から，システム出力を終了する時点までが主たる対象となる。いま，N個のテストジョブを多重度nで処理すると，処理は図5

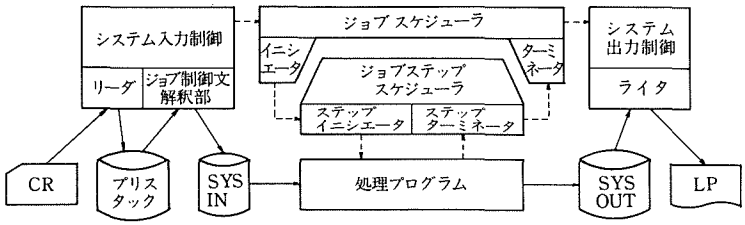

図4 FACOM230−75 Monitor Ⅶのバッチ処理

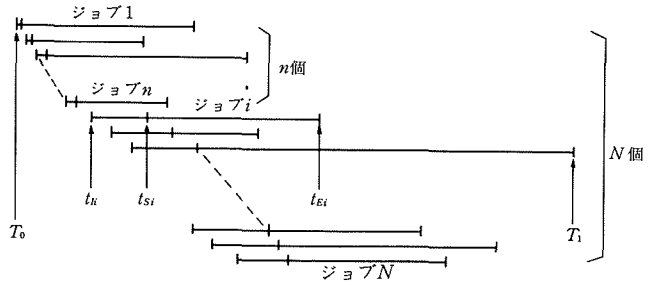

図5 ジョブの多重処理

に示すように進行する。ジョブ i について，システム入力時刻を t_{Ii}，処理開始時刻を t_{Si}，終了時刻を t_{Ei} とし，最初のジョブのシステム入力時刻を T_0，最後に処理を終了したジョブの終了時刻を T_1 とすると，バッチ処理について以下の諸量が定義される。

スループット : $\mathrm{BT}=N/(T_1-T_0)$ ……(1)

平均処理所要時間 : $\mathrm{ET}=\sum_{i=1}^{N}(t_{Ei}-t_{Si})/N$ ……(2)

平均ターンアラウンドタイム : $\mathrm{TT}=\sum_{i=1}^{N}(t_{Ei}-t_{Ii})/N$ ……(3)

一方，TSS 処理については，端末から投入されたコマンドは直ちに解釈，処理され，必要な応答を返す。コマンド j の処理に要する時間を t_j とし，時間間隔 T の間に処理されたコマンドの総数を M 個とすれば，コマンド処理について次の諸量が定義される。

スループット : $\mathrm{CT}=M/T$ ……(4)

平均レスポンスタイム : $\mathrm{RT}=\sum_{j=1}^{M}t_j/M$ ……(5)

図6に示すように，TSS 処理はこのようなコマンド処理のくり返しよりなる。1回のくり返しはインタラクションとよばれ，図示のとおりコマンド処理時間 t_P，送受信時間 t_T, t_R および端末利用者が次のコマンドを発するまでに要する時間 t_H の和よりなり，次式で表わされる。

インタラクション時間 : $\mathrm{IT}=t_P+t_T+t_H+t_R$ ……(6)

システム側からは，通常は t_T, t_H, t_R を区別する必要はないので，次のようにこれをまとめて思考

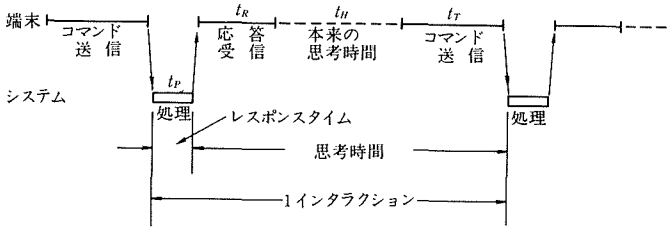

図6 コマンド処理のタイムチャート

時間とする。

思考時間 : $HT = t_T + t_H + t_R$ ……(7)

HT はt_Pに比べ1桁程度大きく，15秒程度である。

バッチ処理とTSS処理が混在する状態のテストは，端末からコマンドを投入しながら同様にベンチマーク・テスト用ジョブを処理させて行う。TSSコマンドは本来，端末キーボードから人間が投入するものであるが，一定の発生条件を保ちながらこれを行うのは極めて困難なので，現実には予めコマンド列を紙テープ上にさん孔しておき，紙テープリーダー付の端末から投入する方法を用いる。このとき，各コマンド間には紙テープ上で適当な個数のスペースをとり，思考時間を模擬する。ただし，各端末から投入されるコマンド列を各々異なるものにするのはあまりにも煩雑なので，すべて同一のコマンド列を用いる。

3 計算処理のマクロモデル

3．1 マクロモデルとシミュレーション

ベンチマーク・テストによって評価が可能なのは，現実のシステムかそれを多少組替えたもの程度に限られ，システムを大巾に変更，増強した場合等のテストはできない。一方，すでに述べたようにシステム増強の検討はセンター運営上重要な課題であり，増強システムの性能評価を行う必要性は高い。われわれはこれに対処するため，システムのマクロなモデル化を行い，その上でベンチマーク・テストに準じたシミュレーションによって，ジョブおよびコマンド処理の性能評価を行う方法を用いている。

計算機システム利用者の立場からは，システム各部の動作状態の詳細よりは各ジョブステップごとのリソース使用時間，量，リソース利用率等のマクロなデータに関心がもたれる。それゆえシミュレーションのためのモデル作成に際しても，システム各部の詳細なモデル化を行うことは無意味であり，収集すべきデータに合わせたマクロなモデル化を行うべきである。

3．2 バッチ処理のモデル

バッチジョブは一般に数個のジョブステップからなる。各ジョブステップはcpuの使用と入出力処理のくり返しからなり，あるジョブステップのプロセスが入出力処理を行っているとき，cpuは他のプロセスを処理することができ，多重処理が行われる。ジョブステップ処理の前後ではOSが介入し，開始，終了の制御を行う。さらに，cpu使用と入出力処理の切換え時にもOSのサービスを必要とするが，開始，終了制御に比べるとその時間はごくわずかである。また，ジョブ全体の開始，終了に際してもOSが介入し制御する。図7はこれらの関係を示したもので，バッチ処理モデルはこれにもとづき，ベンチマーク・テストによって得られる各部分の所要時間等の実測値を用いて構成される。

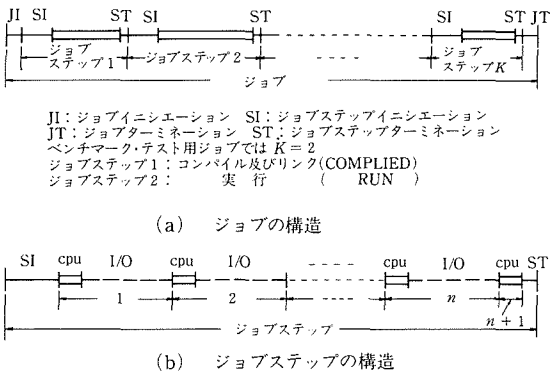

図7 バッチジョブのマクロモデル

ジョブステップの処理においてcpu使用と入出力処理のくり返しが何回行われるかは，入出力回数の記録から知ることができるが，各回ごとの所要時間はわからず，通常はステップ毎の総計のみが収集できる。そこで，われわれのモデルでは各回ごとの所要時間は一定であるとしている。センターで処理されるジョブの大部分はFORTRANで書かれ，コンパイル及びリンク，実行の2ジョブステップからなるコンパイル～実行型のジョブである。各ステップで行われる入出力はカードおよびラインプリンタを意識しており，カード1枚（80バイト），ラインプリンタ1行（約130バイト）を単位として十数～数十レコードを1ブロックとするものが大半と考えられる。それゆえ各回の入出力処理時間を一定とみなしてもよいと思われる。各回のcpu使用時間は変動するが，一般的なプログラムではその分散はそれほど大きくないと考えられるので，これも一定とみなしても問題はないと思われる。

多重処理を行っている場合には，cpuは多重のプロセスに競合使用され，図8に示すように待

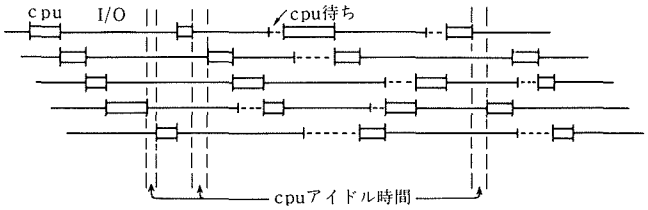

図8　多重処理

ちが生じる。ベンチマーク・テストの際に収集される時間はジョブステップの開始，終了時刻と，各ステップにおけるcpu使用時間（cput），メモリ占有時間（uset）である。したがって，入出力処理時間，OSで費された時間，cpu競合による待ち時間などをこれから分離する必要がある。われわれは以下のようにしてこれを行っている。

（1）cpu競合による待ちは多重処理の際に発生するので，多重度1でベンチマーク・テストを行うことにより，待ち時間を含まないデータが得られる。

（2）上記のデータにおいて，メモリ占有時間とcpu使用時間との差が入出力処理時間である。

（3）同じく上記のデータにおいて，各ジョブステップの開始から終了までの時間とメモリ占有時間との間に差があるが，これがOSで費された時間である。

（4）OSの費す時間はすべてジョブステップの開始，終了制御の段階で費されるものとみなす。

（5）あるジョブの終了時刻との間に差があるが，これがジョブ全体の開始，終了制御のためにOSで費された時間に相当する。

以上の各々は，時刻の記録がOSのサービス・プロセスと利用者プロセスとの切換え時に行われるということから導かれる。ただし，「OSのサービス・プロセス」と「利用者プロセス」の区分はOSによって異なるので，他機種でのベンチマーク・テスト結果との比較を行う際には注意を要する。これらの関係を図9に示す。

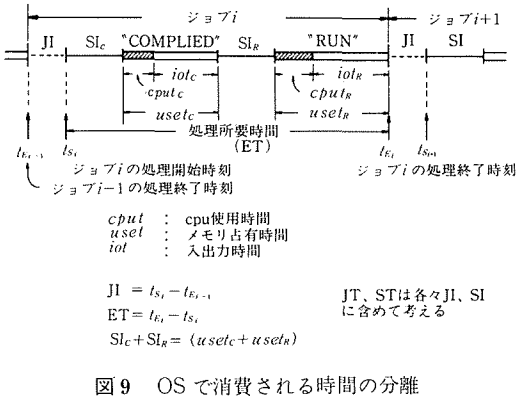

図9　OSで消費される時間の分離

3．3 コマンド処理のモデルとバッチ～TSS 混在処理系

TSS 処理における処理の単位はセッションおよびコマンドである。セッションはTSS 処理で行われる１回の計算であり，バッチ処理におけるジョブに相当する概念である。一方，コマンドはセッションを構成する処理の単位であるが，バッチ処理におけるジョブステップとは異なり，１セッションは通常数十個以上のコマンドからなり，個数は不定である。

TSS コマンドは，端末利用者が直接キーボードを操作してシステムに投入するもので，利用者は投入したコマンドに対するシスポンスを得てはじめて次のコマンドを投入できる。そこで，この間の待ち時間すなわちレスポンスタイムは平均１秒以下程度と極めと短いことが要求される。ただし，コマンドの中には大量のファイル入出力を行うものなど，処理に長時間を要し，レスポンスタイムが長くても構わないものがあるので，OS はコマンドをいくつかのタイプに分け，極めて短いレスポンスタイムを保証すべきものとそうでないものに分けて制御する。

１個のコマンドを処理し，端末に応答してから次のコマンドが到着するまでの時間，すなわち思考時間は十数秒程度に達する。したがって通常は次のコマンドが到着するまでの間，そのセッションのプログラムをメモリにおかず，ロールアウトする。ただし，この部分の制御は OS によて異なり，われわれの対象である FACOM230－75 Monitor-Ⅶシステムでは，即時応答を要するＴ－タイプのコマンドについてはロールアウトを行わないようにしている。ここでＴ－タイプコマンドとは，プログラムテキストの入力，編集等を行うためのコマンドである。この他，Ｆ－タイプ(ファイル操作)，Ｓ－タイプ(サブシステム制御)，Ｒ－タイプ（プログラム実行）の３種類のタイプがあるが，これらは一括して制御される。表２は北大大型計算機センターのTSS処理における各タイプの分布の測定例である。

表2 テスト用コマンド群

タイプ	cput	iot	メモリ使用量	個数	百分比
T	34ms	360ms	23k語	202	52%
F	50	765	29	111	29
S	150	869	20	18	5
R	205	4,155	52	55	14
備考	各タイプごとの平均値			総数 386	

バッチ，TSS の２種の処理形式が混在するシステムでは，バッチジョブの各ジョブステップと TSS セッションの各コマンドがともに cpu を使用し，競合を引起す。コマンド処理は短時間でなされるべきであるから，OS はコマンド処理を優先するように制御する。したがって，TSS 処理を行いながらバッチジョブ群を実行すると，バッチ処理単独の場合より処理に長時間を要し，バッチ処理能力は低下する。この低下の程度が混在系の性能評価指標となると考えられる。

4 バッチ～TSS 混在処理系のシミュレーション

4．1 シミュレーション手法

前述のように，実在のシステムを性能評価の対象とする場合は，ベンチマーク・テストが有効である。しかしながら，性能評価を行う主要な目的の一つであるシステムの増強計画に対しては無力である。また，TSS コマンドについてはバッチ処理用ベンチマーク・テストジョブに見合うようなテストコマンド群を用意し，一定の条件の下で多数の端末から投入することは容易でない。それゆえ，性能評価を実測のみによって行うことには限界があり，他の手段と併用することが必要である。シミュレーション手法はこのような場合に有力な手段の一つである。

計算システムの動作シミュレーション技法は多種類あり，われわれはこれまでいくつかの方法を用いてきたが，ここではシミュレーション専用言語として広く使用されている GPSS を用いている。ただし，モデルの記述自体はわれわれが以前から使用している E-net 手法にもとづいて行

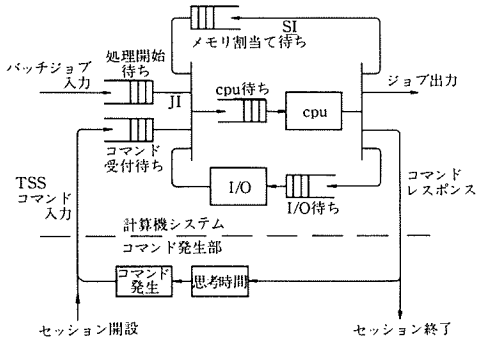

図10　シミュレーション・モデル概要

表3　シミュレーション環境

機種およびOS	
FACOM230-75, Monitor-Ⅶ	
ハードウェア構成	
CPU台数	：1
入出力チャネル台数	：2
メモリのユーザ使用可能領域：特記なきとき400k語	
処理負荷	
バッチジョブ：表1参照	
TSSコマンド：表2参照，思考時間15秒	
パラメータ	
バッチ処理多重度	：1～16, 主として6, 8, 10
T-タイプコマンド処理多重度	：1, 2, 主として1
F, S, R-タイプコマンド処理多重度	：2, 3
アクティヴ端末台数	：1～30

い[4][5][6]，シミュレーションプログラム作成の段階でGPSSを使用している。システムは図10に示すようにcpu，入出力チャネルおよびこれらが使用中のとき終了するのを待つバッファの組合せよりなる待ち行列系としてモデル化される。これをGPSSで記述し，テストジョブおよびコマンドを処理して得られた実測値をデータとして与えることによりシミュレーションが行われる。対象としたシステムは前記FACOM230−75であり，表3に示すような構成となっている。なお，プログラムはHITAC VOS 3 GPSSを用いて作成されている。[7]

4．2　バッチ処理のみシミュレーション

実測の場合と同様，シミュレーションにおいてもバッチ処理のみの動作を基準とし，TSSコマンドの投入による変化（低下）の程度をもって混在系を評価する。バッチ処理の運転においては，多重度をどのように設定するかが重要なファクタなので，実測，シミュレーションいずれの場合も多重度を変数として処理能力を求める。図11は結果の1例で，スループット最大値を与える多重度と，ターンアラウンドタイム最小値を与える多重度とに差があることが認められる。なお，ターンアラウンドタイムはジョブ処理の正味の所要時間にシステム入力ファイルでの待ち時間を加えたものである。図中△印は実測値で，シミュレーションの精度が極めて良いことがわかる。

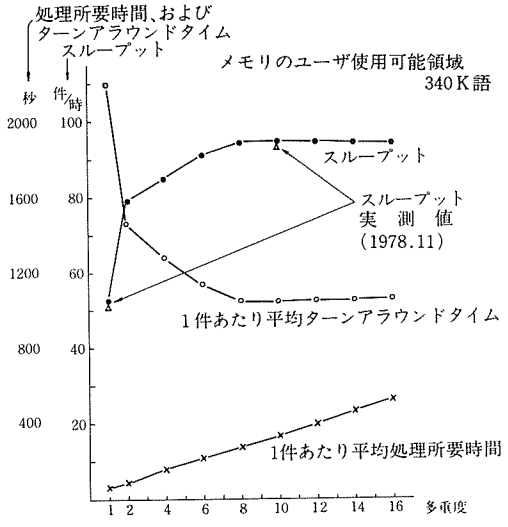

図11　シミュレーション結果－1（バッチ処理）

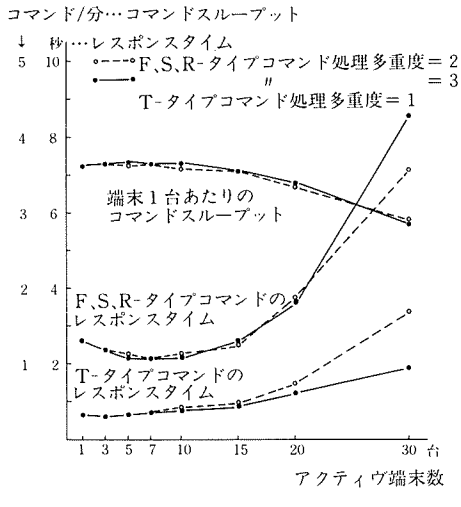

図12　シミュレーション結果－2（TSS処理）

4．3　TSS 処理のみのシミュレーション

TSS 処理についても，バッチ処理の場合と同様，TSS 処理のみの場合を基準にし，混在による低下をもって性能指標とする。TSS コマンド処理に関しては，同時に何台の端末が動作しているかが重要なファクタとなる。すなわち，TSS 処理能力は基本的にはコマンド・スループットで表わされるが，端末ごとの単位時間あたりコマンド発生量はほぼ一様であると考えられるので，システムに投入されるコマンド数は同時動作中の端末台数（アクティヴ端末数）に比例する。図12はこのシミュレーション結果の例である。前述のように FACOM230－75 の TSS 処理には4種のコマンド・タイプがあるが，制御はＴ－タイプとその他（F，S，R）タイプの2群に分けて行われる。図からわかるように，アクティヴ端末数の大きいところでは，F，S，Ｒ－タイプコマンドの制御条件によりレスポンスタイムが相当異なってくる。

4．4　バッチ～TSS 混在系のシミュレーション

バッチ処理，TSS 処理が混在する場合についても，シミュレーション自体は全く同様であり，問題はむしろ混在系の性能をどう定義するかにある。前述のように，センター運営の面で重要な性能評価指標として TT, RT, BT, CT の4種の量がある。このうち前2者は時間の次元をもち，後2者は時間の逆数の次元をもつ。しかしながら，TT は分～10分のオーダであるのに対して RT のそれは秒～10秒で 10^2～10^3 のひらきがあり，また BT の単位であるバッチ処理と TSS 処理は異質の概念であり，両者を共通の指標で表現することはできない。

そこで，ここでは以下に示すように2種の評価指標を2次元表示する方法を用いる。この方法は TSS 処理，バッチ処理各々の評価指標を XY 座標軸にとり，運転条件をパラメータとして表示するものである。

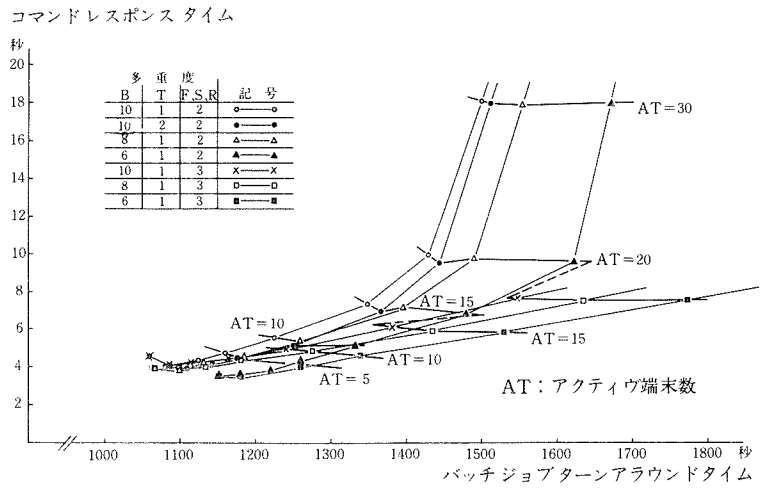

図13　バッチ処理～TSS 処理混在システムの性能

図13はバッチジョブのターンアラウンドタイム TT とコマンドのレスポンスタイム RT が，運転条件によりどのように変化するかを示している。ここで RT はＴ－および F，S，Ｒ－両タイプの平均値を用いている。この結果から，F，S，Ｒ－タイプコマンド処理多重度を一定にしたとき，バッチ処理多重度は RT にほとんど影響を与えないことがわかる。逆に，バッチ処理多重度を一定として F，S，Ｒ－タイプコマンド処理多重度を変えると TT は大きく変動する。したがって，F，S，Ｒ－タイプコマンド処理多重度の設定は十分慎重になされるべきである。図13の例

では，バッチ処理多重度 10，F，S，R－タイプコマンド処理多重度 3 の場合がほぼ最良の運転条件である。一方，T－タイプコマンド処理多重度は TT，RT ともに大きな影響は与えない。また，AT=20 から 30 の間で RT は急激に増大しており，ここで用いた処理負荷の下では AT=20 が限界であることがわかる。

図 14 には cpu 増設における性能向上が示されている。cpu 増設の効果は TSS 負荷の大きいところで著しく，RT については 1/2 に，TT については 1/2.5 に減少している。これから，マルチプロセッサ構成の採用による性能の改善が著しいことがわかる。また，cpu 台数を 3 台以上にすれば性能はさらに向上するが，図からわかるようにそれほど大巾ではない。なお，ここで入出力チャネルは cpu がアイドルにならぬよう十分な台数が選ばれている。

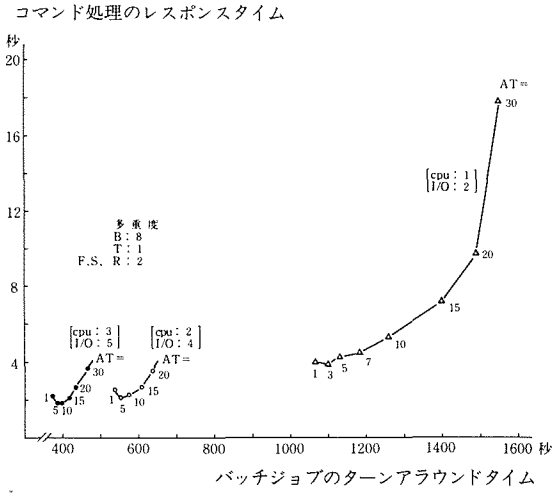

図14 機器増設の効果

5 おわりに

計算機システムの性能評価技法は極めて多数あるが，バッチ処理，TSS 処理が混在するシステムを扱い，しかもシステム運用に直ちに反映できる形で実施できる手法はあまり見当らない。ここに述べた方法は，このような環境下で使用することを目的とするもので，その特徴は以下のように要約される。

（1）ベンマーク・テストにより得られる実測値にもとづいてシミュレーション・モデルを構成している。

（2）運転条件，システム構成ならびに負荷を任意に選んでシミュレーションを行い，実測値に連続する形で結果を得ることができる。

（3）大学計算センターに限らず，汎用 OS の下でバッチ処理，TSS 処理を行う任意の計算機システムに適用可能である。

（4）ジョブ実行の開始，終了時刻，cpu 使用時間，メモリ占有時間及び占有量，入出力回数等，通常の会計ルーチンで収集される程度のデータからモデルを構成できる。

（5）スループットのような総合評価量に関しては十分良い精度のシミュレーション結果が得られ，たとえばバッチ処理スループットについて，実測値との差は 2％程度である。

（6）コマンドレスポンスタイムに関しては，アクティヴ端末数の大きいところでの実測値が得られないことと，時間測定を精密に行うことができなかったために，シミュレーション結果と実測値との比較は厳密には行っていないが，業務運転時に得られた値から推定するとほぼ良好な精度でシミュレーションが行われている。

（7）シミュレーション結果は負荷および運転条件をパラメータとして表示でき，容易に解釈することが可能で，従来「勘」にたよって行われていた運転条件の設定を定量的に行うことができる。

シミュレーション・モデルを構成するのに用いられる諸量の定義はOSによって異なる。したがってこの手法は，異機種間で性能比較を行うのには未だ不十分である。また，モデル構成に際して，実測値として得られる諸量に加え，OSの制御方式（スケジューリング・アルゴリズム，プロセスの状態遷移制御，ロールイン／アウト機構など）の概要を知る必要があり，シミュレーションの精度はこれらの情報の詳細さに影響される。しかし，モデル構成ならびにシミュレーション実行の手順はどのシステムに対しても全く同様であり，上述のように広範囲にわたるシステムおよび運用環境に対して適用可能である。

終りにあたり，データの収集その他でお世話頂いた北大大型計算機センターの江丸敏夫，相良砌，大島雅明各氏をはじめ，同センターおよび富士通株式会社の関係各位に深く感謝する。また，ここで述べた手法は当研究室に在籍された阿部政樹，橋津正晶，針原森夫，我妻新吉各氏により，バッチ処理，TSS処理各々単独に順次開発されてきたものを基礎として展開されてきたものであることを付言する。

参考文献

(1) 萩原　宏　情報処理，**13**（昭47），11，p. 740
(2) 栃内香次，永田邦一　電気学会情報処理研究会資料，IP－77－24（昭52）
(3) 相良　砌，大島雅明，他　北大大型計算機センターテクニカル・レポート，**1**（昭53），p. 74
(4) 橋津正晶，阿部政樹，栃内香次，永田邦一；北大工学部研究報告，**78**（昭51），p. 59
(5) 針原森夫　北大大学院工学研究科修士論文，（昭53）
(6) 我妻新吉　同　（昭54）
(7) 石井　裕　同　（昭55）