

## 性能試験の一方法に関する報告

植田 英範\*・福嶋 輝久\*\*

### A Methodological Approach to Bench Mark Test

HIDENORI UEDA\* AND TERUHISA FUKUSHIMA\*\*

#### 1. はじめに

大学における共同利用を目的とするコンピュータには、その用途、目的のレベル、性能とコストのレベル、設備環境のレベル等々、夫々のレベルで多種の問題を考慮する必要がある。

これらの問題については、当計算機センターの主計算機のリプレースの時期を機会に整理、検討し、最適システムの導入と諸問題の対応に関する提案が長きに亘って行われてきた。

教育利用と研究利用双方のニーズを十分に満たすためには、多重処理性能と演算速度の高性能化という極めて整合の計り難い問題に直面する。これらの問題は極言すればバッチ主体、またはトランザクション主体、さらに TSS 主体等の選択に依存するが、ニーズとの矛盾によりそのどちらかに偏ると、実際運用上では計算機特性を生かせない弊害が生ずる。

そこで、望まれる性能条件の整合を計り、その細部の検証に当って各種システムに公正な相対評価を与えるために必要な性能試験の3方法を提唱し、これを試みた (FLOW Vol7, No3, 1981. 8. 1)。

一つは、具体的な実施が比較的簡単であるという理由から、代表的なプログラム数本を用いたベンチマークテストを採用したものである。テストに当っては、各メーカーの計算センター、およびそのメーカーの紹介により他大学の共同利用センターを利用させて戴いた。その結果、これらの成績を観るとおよそ各フレーム・メーカーの推定値性能 (MIPS, KOPS) 値に沿っていることが確認できた。

次は、TSS 性能の適合性を観るために各フレーム・メーカーで用いている測定ソフトウェア・ツールにより、TSS 端末数とレスポンス・タイムの相対関係のシミュレーションを実施して戴えたものである。しかしこの方法は、試験環境に共通性を欠くばかりでなく、実際環境に於て想定し得る性能を正確に反映するツールであるかの判断規準が無いため、カタログ性能の基

\* 電子計算機センター主任, \*\* 電子計算機センター職員  
Staff, Computer Center

礎を裏付ける意味合いを持たしたものとしてみ参考とした。

これらの試験からは、試験環境と実際環境とのギャップが夫々のメーカー毎に異なるという点で相対比較の判断上危険が伴う。そこで実際環境に幾分でも貢献すると思われる試験方法を試みることになった。これが性能検証をさらに工夫して試みようとした動機である。

その一方法として、マルチプログラミング性能指向のベンチマークテストを試みた。これを便宜上以降において、K モデル・ベンチマークテストと呼ぶこととする。その結果、OS のオーバーヘッドのかなり小さい CPU バウンダリのジョブでも、MIPS 値には影響を受けないトータルスループット性能が存在することが実証された。すなわち、多重度を徐々に上げると MIPS 値の高い計算機でも、そうでない計算機と ELAPS タイムが変わらなくなり、単位時間内に処理可能なジョブ数に差が無くなるという現象である。

またこれは、端末台数が多くなった場合や周辺装置が多様化した場合等に準じてジョブの内容を変化させたときは、OS のオーバーヘッドがどのような要因で大きくなるかを観る可能性を導き、極めて興味を引く事例であると思われる。

## 2. 試験の概要

### 2.1 試験環境

試験をするには、所要時間が少なくとも1時間は専有することが必須なので、必ずしも提案マシンと同一システムでなくても良いことを前提としたが、フレーム・メーカーの計算センターで、提案システムと同一 OS を用いることを条件とした。

試験に影響する要因、リソース等については、表1のように明らかにさせた。

表1 試験システムの環境

項目 \ 試験機種別	A	B	C	D	E	F
プロセッサ数	1	1	2	1	1	1
主記憶容量	8 MB	6 MB	4 MB	8 MB	4 MB	2 MB
キャッシュメモリ容量	16KB	8 KB	8 KB	8 KB	4 KB	8 KB
推定 MIPS 値*1	4.1	3.4	2.4	0.8	0.4	0.7
システムの発表年	1978*2	1977	1979	1978	1980	1980
テストジョブの起動方法	コンソール入力	TSS 端末入力	システムスケジューラ	システムスケジューラ	システムスケジューラ	コンソール入力

\*1 必ずしもメーカーの公表値とは限らない

\*2 試験対象マシンは発表時期の改造機

### 2.2 テスト・プログラム

テスト・プログラムは次の点を考慮して、図1のような FLOW に表わされるイン・コア・

ソートの単位時間（3600秒）内アドレスの繰返しジョブとし、60秒間隙に同一ジョブを起動して多重度を上げた。

(1) シングルランの状態の処理速度を見るために、どの試験機でも、1サイクルの実行時間（厳密な意味とは異なるが、近似しているので ELAPSE タイムと表わしている）が60秒以内で終了するような短い JOB とした。

(2) コンパイラの影響を少なくすることと、メモリ容量の制約を少なくするためにプログラムを小さくした（JOB 当り約 16 KB のメモリ使用量）。

(3) 実際環境下に反映するよう、計算時間の対象を、CPU タイムではなく、ELAPSE タイム（この場合、プログラムのある区間のパススルータイム）とした。

(4) I/O 機器性能の影響を避けて、入出力は極力少なくした。

(5) OS のオーバ・ヘッドを極力少なくするような JOB の性格を図った。

(6) 相対比較の信憑性を高めるために、CPU 能力のみを対象とした。

(7) 代表的 JOB と比較的性格の傾向が似ているコア・ソートとし、言語は FORTRAN を用いた。

(8) FORTRAN コンパイラの種別、最適化の採択は、メーカーの任意とした。

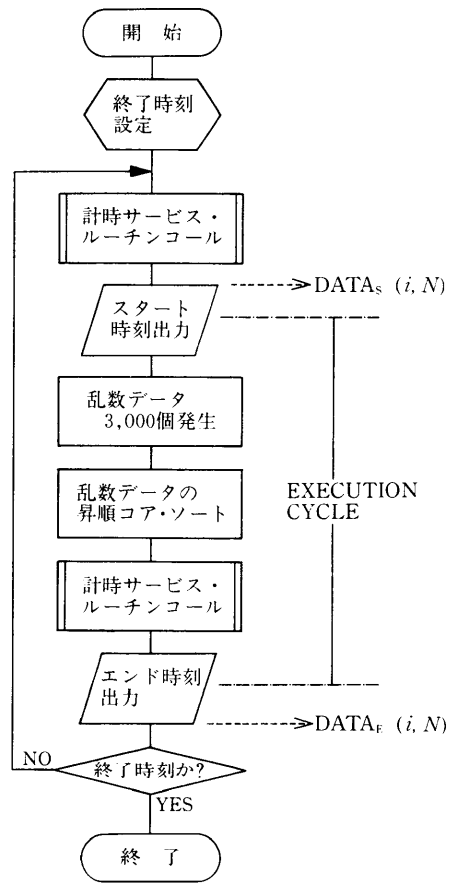
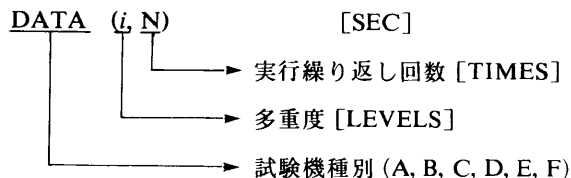


図1

### 3. 試験結果

#### 3.1 テスト・プログラムの実行状況

テスト・プログラムにより出力されたデータを



と表わすとき、テスト開始時刻：DATA (1, 1) を用いて

$$DATA^R(i, N) = DATA(i, N) - DATA(1, 1)$$

とし、タイミング・チャートに表わすと図2が得られる。

これらを試験機別に描画すると図3～図8の結果が得られた。

### 3.2 多重度と時間推移

試験機の種別により、第1回目のジョブ (DATA<sup>R</sup>(1, 1)=0) を採り、DATA<sup>R</sup>(1, N) をプロットすることにより、図9が得られる。すなわち図9は、多重度の向上と共に ELAPSE タイムが、各試験機毎にどのように推移したかを観ることが出来る。

次に、このデータより実行回数毎の増分、すなわち

$$DATA^{INC}(1, N+1) = DATA(1, N+1) - DATA(1, N)$$

を求め、これをプロットすることにより、図10が得られる。

図9, 図10の二つの図から、多重度増加が、各試験機でどのように、ジョブの ELAPSE を遅延させているかの傾向を観ることが出来る。

### 3.3 試験結果

以上のテスト・データより表2のような試験結果を得る。

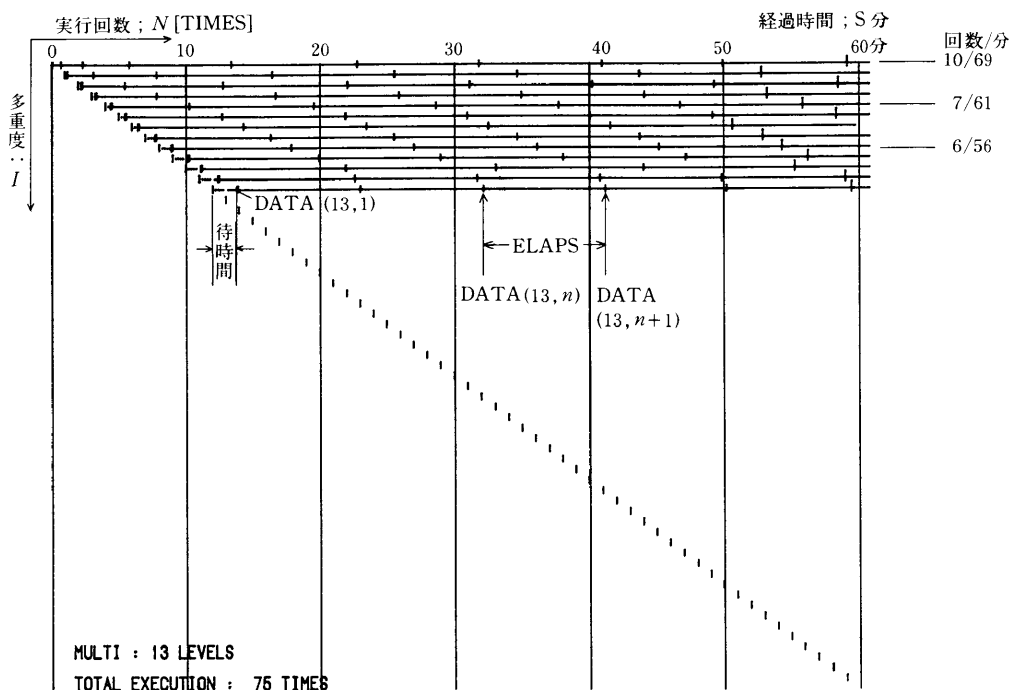


図2 実行状況の表示例

資料：性能試験の一方法に関する報告

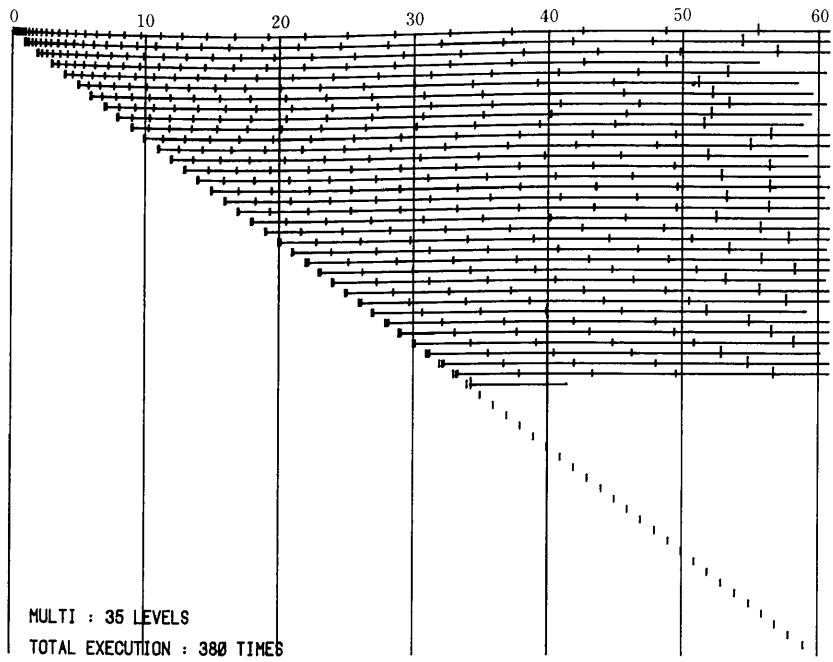


図3 機種A 実行状況

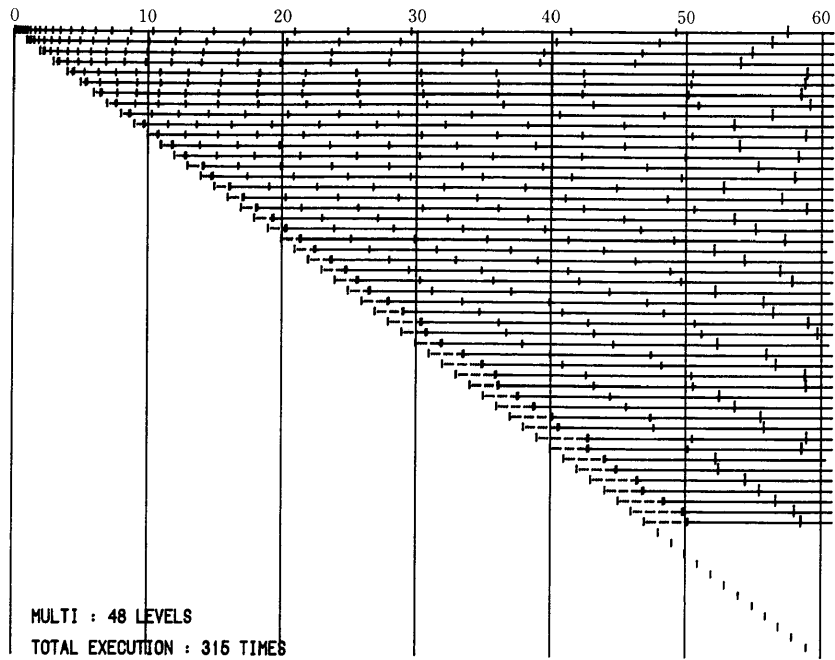


図4 機種B\*2 実行状況

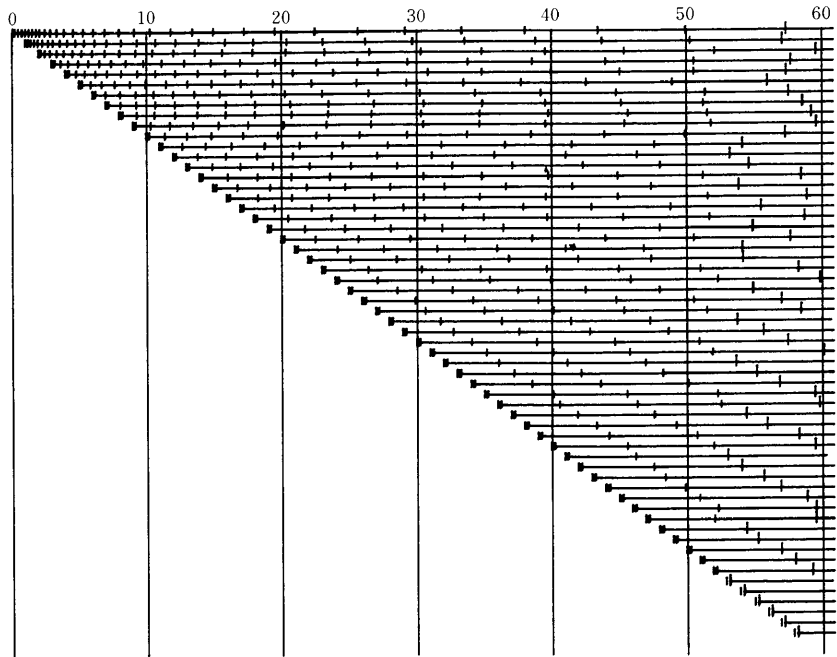


図5 機種 C\*2 実行状況

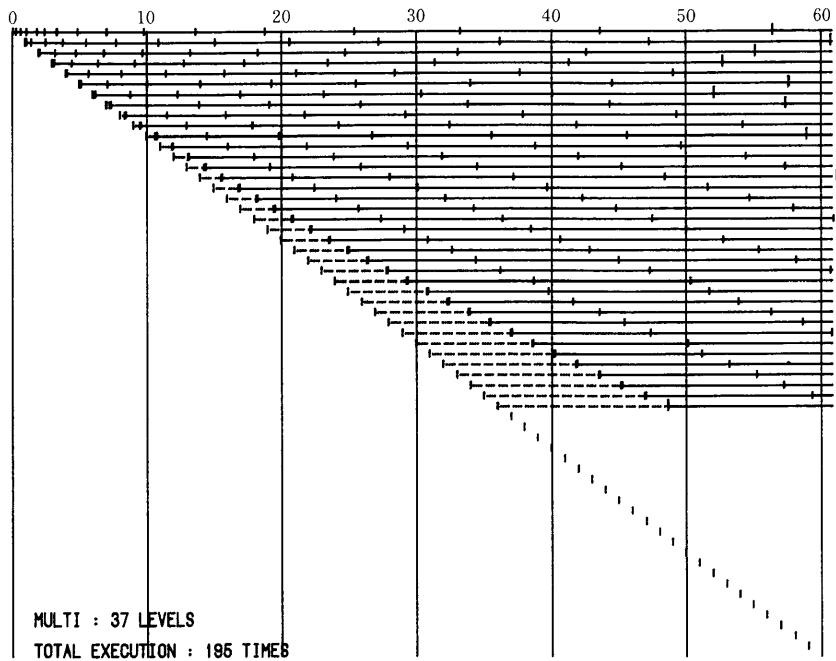


図6 機種 D 実行状況

資料：性能試験の一方法に関する報告

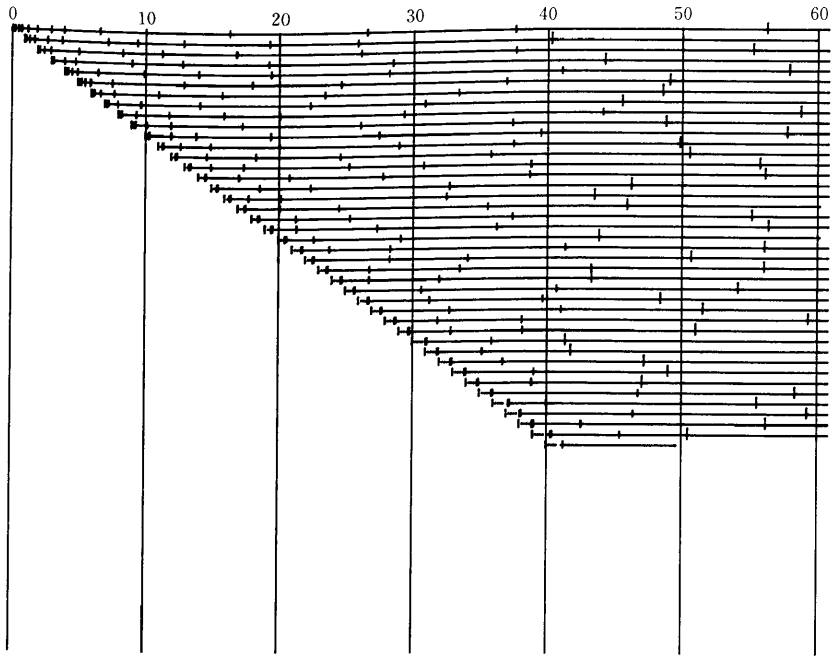


図7 機種 E 実行状況

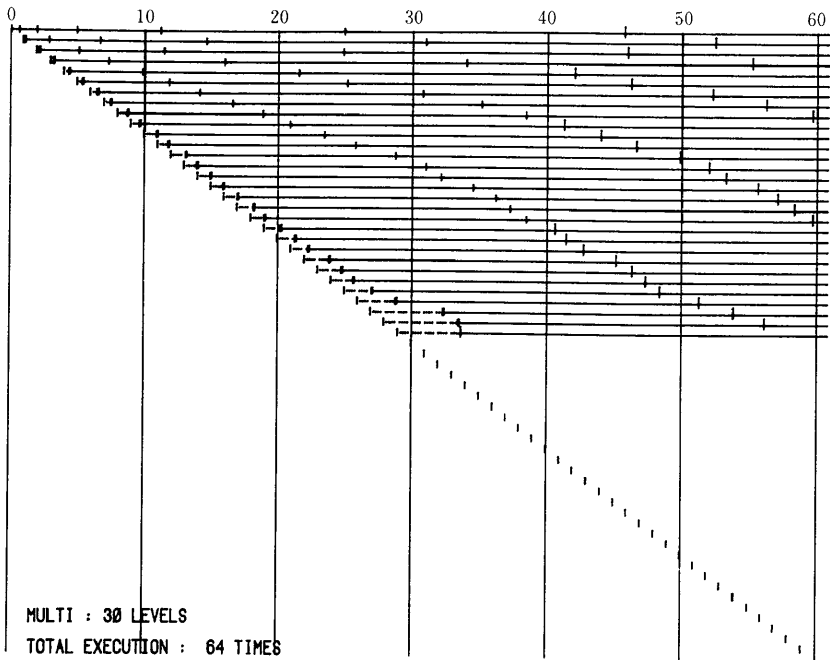


図8 機種 F\*2 実行状況

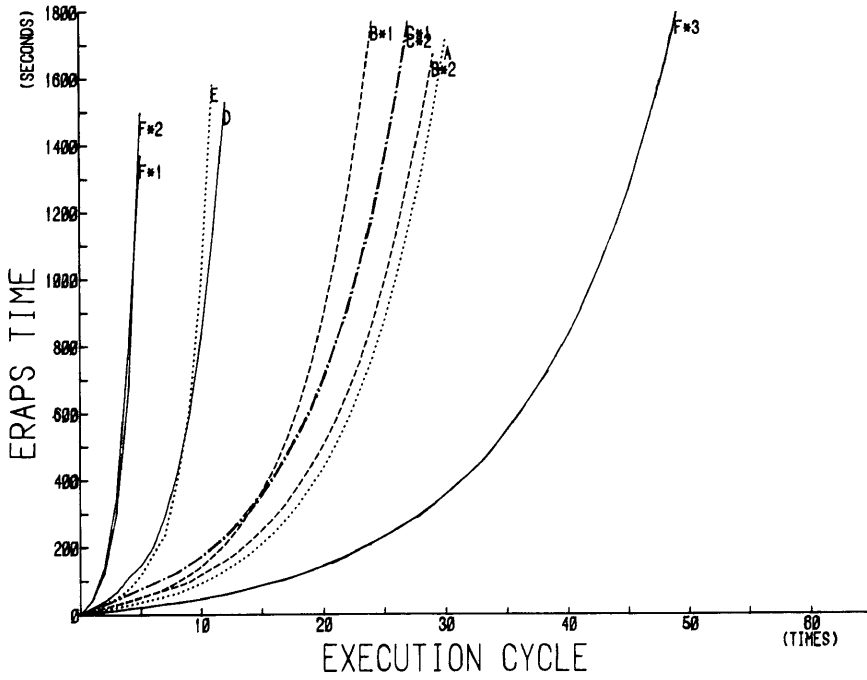


図9 実行回数とエラップス

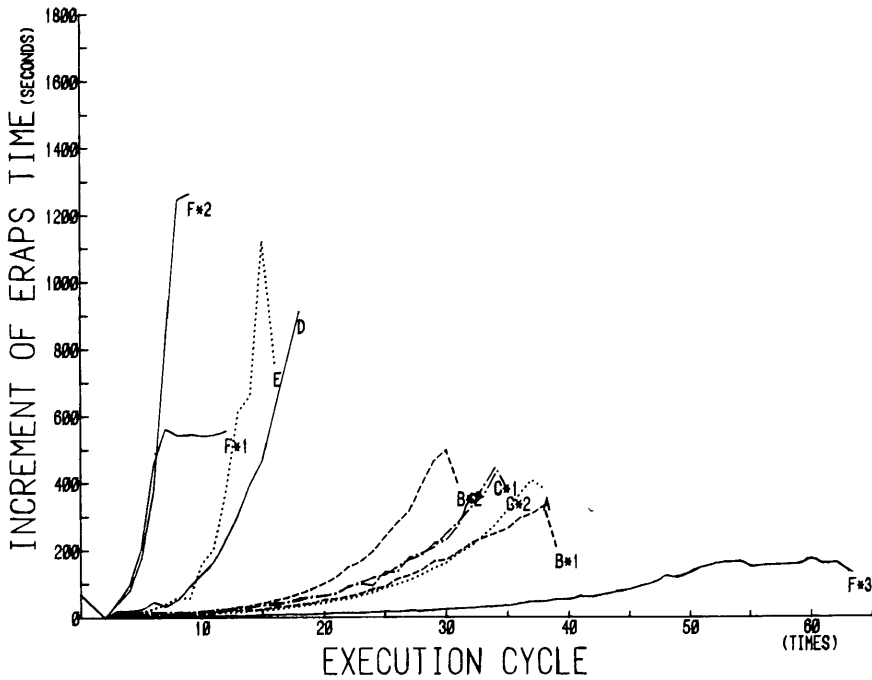


図10 実行回数とエラップスの増分



表2 試験結果

項目	試験機種別									
	A	B *2		C *3		D	E *1	F *9 *1		
シングルランELAPSEタイム	7.5	10.5	10.5	15.5	15.5	20.5	16.0	43.0	44.0	4.6
10多重時の平均 ELAPSE*4	97	111	103	80	80	210	55	678	588	53
20多重時の平均 ELAPSE	160	211	220	158	158	419	111	1221	—	103
最大多重度 (I <sub>max</sub> )*5	60	32	48	57	59	43	59	30	13	36
単位時間内の合計実行回数 (N <sub>TOTAL</sub> )*6	428	306	315	428	434	153	196	64	75	688
最長ELAPSE/多重度*7	7.2	9.9	10.6	7.2	7.0	17.4	21.5	45.5	42.0	4.8
OSのオーバヘッド率(H)*8	10.8	10.8	8.1	7.9	6.7	12.9	12.9	23.5	8.3	12.1

注)

- \*1 これらの縦系列は、ソート・データの個数を3000回から1000回に減じたときの結果。  
 \*2 同一機種に於て、ディスパッチャーに割り当てられているプライオリティの変更を行ったケース（後述4.2の(2)）。  
 \*3 単位時間（3600秒）以内に於ける各ジョブの終了が不規則なので再試験を行った。  
 \*4 経過時間600秒時に近似している各ジョブの繰り返し時の平均 ELAPSE すなわち

$$X_{10} = \left\{ \sum_{i=1}^{10} \text{DATA}^R(i, N) \right\} / 10$$

但し、ここでの  $N$  は、1 から  $n_{\max}$  までのシーケンシャルで  $600 - \text{DATA}^R(i, N)$  が最も 0 に近いときの  $N$

- \*5 単位時間内（3600秒）に実行を開始することが出来た最終ジョブまでの多重度：  $I_{\max}$ （図2～図8中 MULTI:  $I_{\max}$  LEVELS と表示）  
 \*6 単位時間内（3600秒）に実行されたサイクル数の総計。

$$: N_{TOTAL} = \sum_{i=1}^{I_{\max}} \left\{ \sum_{n=1}^{I_{\max}} N(i, n) \right\} - 1$$

（図2～図8中には、“TOTAL EXECUTION :  $N_{TOTAL}$  TIMES” と表示）

- \*7 最も長時間を要した測定時間（ELAPSE）データを捜し出し、このときの多重度で割った値。  
 \*8 OS のオーバヘッド率:  $H$

$$H = \frac{\{(3600/S_E) - N_{TOTAL}\}}{3600/S_E} \times 100$$

$S_E$  : シングルラン状態での ELAPSE（測定時間）

$N_{TOTAL}$ : 単位時間（3600秒）内での合計実行回数

- \*9 同一機種で多重度を抑制したときの結果（このときは、当然シングルラン状態に近づいて行く）

## 4. 比較試験結果と問題点

### 4.1 MIPS 値, KOPS 値について

実際にコンピュータを使用する場合、一般的にはメモリ容量が大きく、処理速度が速いものは性能が高いと言える。しかし、処理速度はカタログ値や、MIPS 値, KOPS 値が高くて、

実際の処理は遅くなるケースもしばしばある。これは MIPS 値は命令の種類によって数字が変わることに因る。

対象となる命令（インストラクション）は、目的やハードの特徴によってあるカテゴリについて使用頻度の高いものが選択される。たとえば、大型科学技術計算のプログラム、すなわち、FORTRAN の実行時間の割合が大きいカテゴリ、あるいは事務計算、そしてオンライン処理のように制御プログラム走行時間の割合が比較的大きい場合のカテゴリでは、頻発するインストラクションは夫々に偏る。また、同一機種であってもカテゴリにより数値が異なる。このようなことから、汎用機の場合、その使用目的に僅かでも正確な処理速度の目安とするために、General Mix 値を MIPS 値の平均命令時間に用いて、この General Mix 値は、事務計算、科学計算、オンライン処理の3者夫々の重みを変化させ、3～4種のカテゴリで抽出している所もある。

このとき、どのカテゴリの Mix を用いたかによって当然 MIPS 値は変わる。また、一般には、アーキテクチャーのちがいが、インストラクション・セットのちがいなどにより、基準を標準として促えられないので、あくまでも推定 MIPS 値として、競合機種を意識した MIPS 値をカタログに記載するケースが多いと考えられる。

また、メーカ公表では、最適条件下における MIPS 値という前提で、最大限の MIPS 値を発表することも見受けられる。

また、基準を一概に定められない理由としては、アーキテクチャーの違いに因る所が大きい。その一つは、バイトマシンとワードマシンのアドレス方式、データ形式の違いである。このため、ある機能は前者が1インストラクション、後者は2～4インストラクションを用い、別の機能ではその逆もあり得る。

また、最近では従来幾つかのインストラクションを用いて果していた機能を、1つのマイクロ・コードで実行してしまうインストラクション・セットに組み込まれるケースがある。この場合インストラクション・セットのレパートリーは、バイト系、ワード系夫々の強化を独自で行ってきているので、換算ケースによって大きな差異が生じることは当然の理である。

さらに、HSB (High Speed Buffer) を持った場合、その大きさによっても速度は異なる。このように、処理速度の標準値として、MIPS (KOPS) 値をそのまま適用するには問題が多い。

このため、MIPS 値は大きくても、このテストの結果では必ずしも単位時間当りの実行回数が比例して大きくなっていないことに興味を持たれる。

## 4.2 主記憶容量

メイン・メモリ容量に関係して、テストプログラムの大きさと、マルチ度の向上は、直接試

験結果に影響した。

今回のテスト・プログラムは、内部メモリで概算 16 KB を要していたものと思われる。また入出力命令の比率が極めて低い（無視できる程）静的なプログラムである。しかも、単位時間内は、プログラム・コントロールで再実行しているので、エンドレスループの処理と同様、一度メモリにロードされると、見掛け上は常駐している形となる。

また、オペレーティングシステム (OS), TSS, コンパイラ等の常駐、非常駐プログラムが占有する領域を合わせ考えると、約30多重で推定 2 MB, 60多重で 4 MB の領域は最低必要であろう。

テスト・プログラムの多重度を向上することによってメモリ不足になり、このときスワップの対象となるのは、スケジューラ、Sysin/Sysout、アロケータ等が考えられる。これに因り、OS のオーバヘッドは上昇し、テストプログラムが投入されてもアロケーション待ちになるのは必至である。使用メモリ容量の大小は、スケジューリング作業に影響し、カタログの MIPS 値からの推測を外れて OS のオーバヘッド・アクセスタイムを含む実測の ELAPSE タイムにより影響を受けたものと思われる。

#### 4.3 キャッシュ・メモリ容量

テスト・プログラムは前述の性格上、キャッシュメモリの速度や、キャッシュサイズが結果に絶大な影響を及ぼしたと思われる。

キャッシュメモリは、メイン・メモリとの相対速度差が大きい程、効果が大きいことはいうまでもない。

また、サイズについては、ヒット率との関係によりその影響は想像できる。ヒット率とキャッシュサイズの関係は、図11のような傾向にあるので、ある程度のキャッシュサイズがあれば、それ

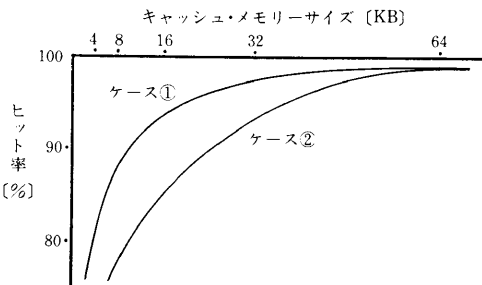


図11 キャッシュ・サイズとヒット率の傾向

以上大きくなっても性能には徐々に反映の度合が少なくなる。但し、図中曲線①のケースと曲線②のケースとでは差が大きいので一概に判断することは危険であろう。現に、キャッシュサイズが2倍になると、性能は大巾に向上するので、機種モデル名を変えているフレーム・メーカーもあるほどである。これは、図中曲線②のケースであるヒット率1%の向上は、実際性能上はそれ以上に向上する。かつ、テストプログラムは、極めて局所性が大きいので、予想以上にキャッシュの影響は大きかったと思われ、テスト機種のテスト環境で、キャッシュ・サイズの違いは十分な考慮を必要とした。

#### 4.4 計測対象時間

JOB がシステムに登録されてから (Last Key in) から、最初の出力が得られる (First Character Out) までの経過時間を通常のターン・アラウンド・タイムとする。この内、プログラムの実行がスタートした時刻と終了した時刻との差を ELAPSE タイムとすると、今回のテストデータの ELAPSE タイムは、これよりも若干小さい値となる。ところで CPU タイムは、ユーザプログラムが実質消費した CPU タイムに、ユーザプログラムが起動したスーパーバイザコールによって OS が消費した CPU タイムを加えているケースと、そうでないケースがある。

ベンチ・マークテストを行う場合、MIPS 値の大方の確証を得るために、テストプログラムの CPU タイムを見て来たが、このときの OS の効率によってこの上の二つのケースから、CPU タイムでは速いが、ELAPSE やターンアラウンドが見掛け程実際には速くない誤算が生ずる恐れは十分にある。そこで今回のテストでは定義の曖昧なデータを用いず、プログラム内のある二つのポイント間の通過時間を ELAPSE タイムとして、実際環境での見掛け速度の比較により近い測定データを採用した。

#### 4.5 マルチ・プロセッサ

OS の CPU 割り当て単位がタスク、またはプログラム単位であるとき、このような性格のテスト・プログラムは、プログラム個々の CPU タイムではなく、スループット性能に絶大な影響を与える。マルチ CPU 環境下では、同時に CPU の数だけの並行処理であるからである。

その相対性能は、マルチ度が向上するに従い、例えば、1, 1.8, 2.6, 3.4と向上する。しかしながら、ここでは MIPS 値とテストプログラムの CPU タイムとの関連は崩れ、マシン性能をこの2つのデータから推測することはナンセンスとなる。

#### 4.6 その他

##### (1) ディスパッチャーのプライオリティ

テストデータ B\*<sup>1</sup> と B\*<sup>2</sup> は、同一機種、同一プログラム環境での結果である。計算機室運用上の考えにより、投入プログラムの終了を幾分でも早めようとして、ディスパッチャーのスケジューリングプライオリティは、通常のユーザ・プログラムよりも僅かに低い。そのために多重度が向上したとき、後続する JOB は徐々に待時間が長くなり、ついに制限時間まで実行されないままに打ち切られる結果となっている (B\*<sup>1</sup>)。そこで次に、ディスパッチャーのプライオリティを、ユーザプログラムと同等にした (B\*<sup>2</sup>)。このケースでは、待時間は無くなるが当然ユーザプログラムの ELAPSE タイムは増加し、単位時間内での合計実行回数は少なくなってしまう。

## 資料：性能試験の一方法に関する報告

### (2) ジョブ・スケジューリング・ロジック

テストプログラムのように、通常 CPU バウンダリの性格を持つプログラムは、ジョブ・スケジューラがその時間経過に従ってプライオリティを徐々に下げてしまう（ダイナミック・スケジューリング）。

テスト機 E の実行状況を見ると、同一時刻に投入したジョブは、同一環境下にあるので、ELAPSE は近似していても良いと思われるが、大巾なバラツキが見られる。その原因は、次の理由に起因していた。

テスト機のプライオリティ・デフォルト値は、50.0000である。

小数点以下の定数 ( $0.0001 < C < 1.0$ ) は、OS の Fine Priority のロジックにより

$$C \doteq \frac{I/O \text{ タイム}}{CPU \text{ タイム} + I/O \text{ タイム}} \text{ の計算が成され}$$

1 秒間隔毎にダイナミックに変動する。

ジョブ投入の最初のイニシエートのときは、ディスクの起動が掛かるので I/O タイムが大きくなり、C の値は1.0に近づく。ところが、投入後の経過時間が増加するに従い、CPU タイムが大きくなり、C は0に近くなり、プライオリティは下がることになる。

このように、同一プログラムだけにより多重度を向上して行けば、この定数 C の僅かの変化が大きく影響し、そのタイミング（タイマ・インタラプト）によって ELAPSE が左右されることにもなる。

しかし、定数 C の値を変化させないことも可能である。

## 5. おわりに

コンピュータ導入のための基礎資料として、各種のテストを行うケースは少なくないが、定常業務を主とするコンピュータ運用の場合、その主な業務プログラムを抜粋し、比較的容易に実際的な性能検証を果すことが可能であるように思われる。しかし、大学の共同利用センターはその処理の多様さから必要性能として一定の要求を満たすテストは多様すぎて、実施は困難となる。それでも、各種プログラムに工夫して、ベンチ・マーク・テストを行っている例は多い。

われわれが今回実施したベンチ・マーク・テストは、あくまでも性能の内、処理速度とトータルスループット性能との関係をマクロ的に掴むためのものであった。

速度の速い機械は、その速度比に比例して、スループット性能が向上しているか否かについては、必ずしも肯定的結果が得られなかったことについて十分な興味を持たれたように思われる。さらに、同じ方法を用い、JOB の性格を、I/O 主体とするもの、TSS 端末からのプログ

ラム開発を主とするもの、ファイル処理を主とするもの等々に変化させ、同じような測定を行えば、デバイス性能に大きく左右されるとしても、さらに有為なデータ、OS の特徴などが見られるのでは無いかと思われる。

最終的には、試験時間レンジを大きくして、これら混在のプログラムにして行えば、マシンの処理限界までのテストは可能である。

今後更に、簡単な方法で実施でき、かつ比較基準を備え易いテスト方法を考究し、有為な資料を提出して行き度いと望んでいる。

おわりに、今回のテスト実施にあたり、終始アドバイスをいただき、かつセンターに貴重な時間を裂いて、テストを実施していただいた、関係各フレームメーカーの各位に、心から謝意を表します。

(1983年1月25日 受理)