OUKA
The University of Osaka
Institutional Knowledge Archive

Title	解説　大阪大学大型計算機センターニュース　No.3
Author(s)	
Citation	大阪大学大型計算機センターニュース. 1969, 3, p. 40-46
Version Type	VoR
URL	https://hdl.handle.net/11094/65127
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

◉ 解　　説

コア増設後のシステム稼動状況について

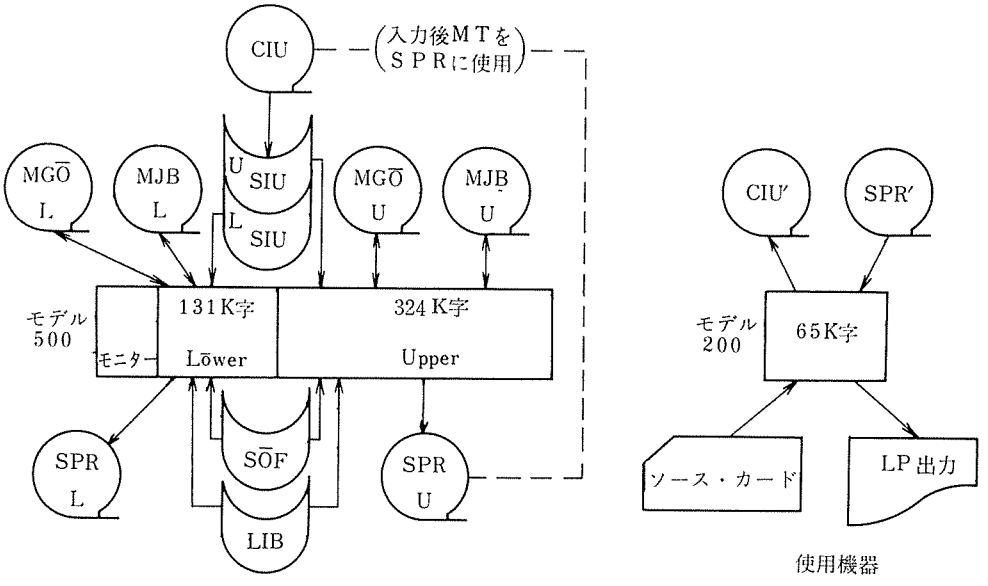
センター・ニュース NO. 2 において，システムの運用方針についてその概要を述べ，稼動効率の報告を行なうと共に，昭和44年1月のコアの増設によって稼動効率が向上する旨のことにも触れておいた。ここではその後のシステムの稼動状況について報告する。

1)　コア 131 K 字の増設に伴う運用方式の変更

昭和44年1月にコア 131 K字が増設され，さらにライブラリー・ファイル（LIB）および利用者ソース・プログラムの入力ファイル（SIU）にディスク・パックを使用することによって，バッチ処理の際に二つの利用者ジョブの並行処理が可能となった。この場合のハードウェア構成を第1図に示す。

そして，運用スケジュールも第2図に示されるように変更された。新しいスケジュールでは，CPU の使用効率の悪い短時間ジョブは，長時間ジョブとの並行処理によって CPU の遊び時間を吸収させるようにし，TSS サービス時のバックグラウンド・ジョブとしては，標準ジョブを処理するようにした。いずれもシステム全体としての処理効率を上げることを目的とした変更である。

第1図　2 job stream バッチ処理におけるハードウェア構成



SŌF：システム・オペレーティング・ファイル
LIB：システム・ライブラリ・ファイル
MGŌ：ゴー・ファイル(相対形式オブジェクト・プログラム)
MJB：ジョブ・ファイル(絶対形式オブジェクト・プログラム)
CIU：コモン・インプット・ユニット
SIU：スタンダード・インプット・ユニット　（利用者ソース・プログラム）
SPR：スタンダード・プリント・ユニット(LP出力イメージ)

使用機器

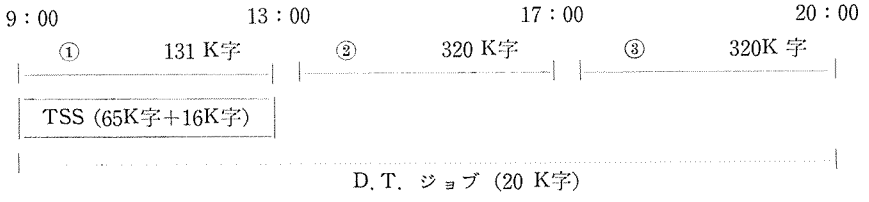
モデル500
MT　6台
ディスク・パック　2台

モデル200
CDR　1台
LP　1台
MT　2台

利用者ジョブは短期間ジョブから長時間ジョブまで様々な性質を持っている。センターがジョブを主として時間的な面から4種に分けて受け付けているのは，短時間ジョブのターン・アラウンド・タイムを早くするために優先処理をすることが目的である。しかし，それにも限度があって，長時間ジョブがいつまでたっても返却されないようでは困る。その適度なバランス（これを数量的に定めることは困難であるが）を保つことが運用スケジュールを決める際の目標となる。他方，運用スケジュールを固定して考えた場合，利用者ジョブの統計的性格が変われば バランスがくずれてしまうことも当然である。

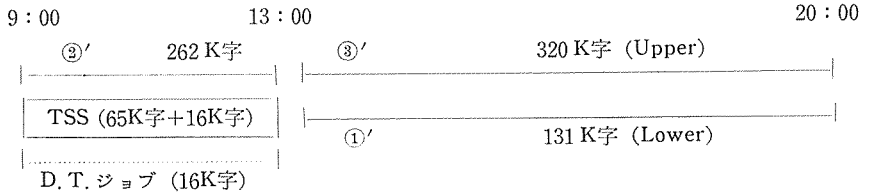
第2図　運用スケジュールの比較

◦昭和44年1月15日までの運用スケジュール



① 131 K字までの短時間ジョブ（3分まで）
（時間が余れば 131 K字までの標準ジョブ）
② 標準ジョブ，131 K字以上の短時間ジョブ（15分まで）
③ 長時間ジョブ，特殊ジョブ（15分以上）

◦ 昭和44年1月16日以降の運用スケジュール



①′ 131 K字までの短時間ジョブ
②′ 262 K字までの標準ジョブ
③′ 長時間ジョブ，特殊ジョブ
ならびに ①′ ②′ でメモリ・サイズをこえるジョブ

（備考）
2 job stream（バッチ処理）では，原則として Lower ジョブに処理優先権が与えられる。

このような利用者ジョブの統計的性格の変化の大きなものとして，年度末に短時間ジョブが増加することがまずあげられる。その他に小さな変化は常にあると考えなければならない。ただ，そのような変化が顕著に現われるのは比較的短時間のジョブの増減であって，長時間のジョブについては，それ程大きな件数の変化が現われるわけではない。実際の所，長時間ジョブの件数が急に増大したとしても，計算機使用時間の絶対量の問題になってしまい，スケジュールをどう変更した所で結局役には立たない。

このような観点から見ると，新しいスケジュールでは前のスケジュールにくらべて短時間ジョブのために長い時間帯が用意されているので，変化に応じて柔軟な態勢がとれる。短時間ジョブが多いときには，午後の時間帯の多くの部分でこれらを処理しても，長時間ジョブとの並行処理によって，システムの稼動効率の低下はある程度の所でおさえられる。また，短時間ジョブがそれ程多くないときには，長時間ジョブの方に処理優先権を与えればよい。

なお，入出力機器と磁気テープ間のデータの移送を行なう D. T. ジョブが，午後の時間帯にはモデル 200 だけで実行されることになった。これでも一応処理可能であるが，出力結果を少しでも早くライン・プリンター上に出して返却にまわすためには，時間的に余裕があることが望ましいので，その後改良が加えられ，11月現在のシステムでは午後の時間帯に二つの利用者ジョブとさらに D. T. ジョブも併せて並行処理ができるようになっている。

2) 2 JOB STREAM 実施後の稼動状況

二つのジョブの並行バッチ処理実施後の稼動状況として，昭和44年2月（第1表，第3図）および4月（第2表，第4図）の場合をあげる。1件当りの平均 CPUtime に見られるように，2月は短時間ジョブの多い時期にあたり，4月はその逆である。

オペレーティング・システム(以下 OS と略記する)の機能を調べるために，2月において2 job stream の行なわれた時間帯に注目してみる。過去の統計データから，この時間帯で処理されたすべてのジョブ（1489件）を1 job stream で処理したと仮定したときの USE time が推定できる。(前回の解説参照。) これによれば CPU 稼動効率は 1 job stream で推定約63%，2 job streamで約75%となり効率の上昇率は約20%である。これは1ヶ月の平均としての値であるから，日によっては上昇率のもっと高い日もある。

ここで 2 job stream 実施後の CPU 稼動効率75%という数字であるが，これがそのまま OS

第1表　昭和44年2月　1日当りの平均処理状況

時間帯区分	処理状況別	件数	1件当り平均 CPU time	CPU time 合計	稼動時間	$\frac{\Sigma \text{ CPU}}{\text{稼動時間}}$
TSS サービス時	フォア グラウンド			28分48秒 (7.3%)	201分 (35.6%)	0.614
	バック グラウンド	23.0 件 (22.5 %)	4分6秒	94分36秒 (23.9%)		
TSS サービスなし	Lower	65.8 件 (64.5 %)	1分51秒	121分36秒 (30.8%)	363分 (64.4%)	0.749
	Upper	13.3 件 (13.0 %)	11分18秒	150分18秒 (38.0%)		
計（総平均）		102.1 件 (100 %)	3分34秒	395分18秒 (100 %)	564分 (100 %)	0.701

注(1) 平均対象日数はTSSサービス時15日間。TSS サービスなし16日間。
特殊なスケジュールの日は除外されている。
(2) 表中（ ）内の数字（%）は処理状況別の割合を示す。
(3) 稼動時間は少くとも一つの利用者ジョブが処理されていた時間。

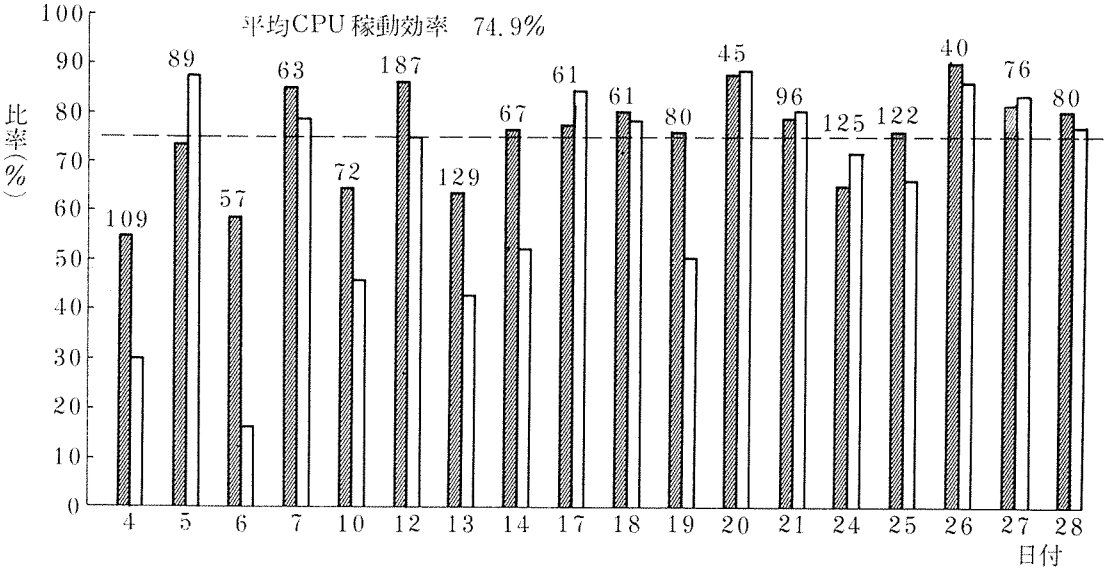
の機能を表わしているとは云い難い面がある。運用スケジュールと現実のジョブの流れとの間には常に若干のずれがあって，その影響が入って来るからである。長時間ジョブとは云っても，エラーのために短時間で終るものもあれば，コア・サイズの関係で短時間ジョブが二つ並行処理されることもあり，オペレーションの態勢としても柔軟性を失わないためには，オペレーターの介入時間が

第2表　昭和44年4月　1日当りの平均処理状況

時間帯区分	処理状況別	件数	1件当り平均 CPU time	CPU time 合計	稼動時間	Σ CPU / 稼動時間
TSS サービス時	フォア グラウンド			17分11秒 (5.3 %)	162 分 (36.4%)	0.610
	バック グラウンド	27.1 件 (38.6 %)	3 分 1 秒	81分22秒 (25.1%)		
TSS サービスなし	Lower	26.2 件 (37.3 %)	1 分19秒	34分25秒 (10.6%)	283 分 (63.6 %)	0.799
	Upper	16.9 件 (24.1 %)	11分 0 秒	191分23秒 (59.0%)		
計（総平均）		70.2 件 (100 %)	4 分32秒	324分21秒 (100 %)	445 分 (100 %)	0.730

注(1)　平均対象日数は TSS サービス時17日間。TSS サービスなし18日間。
(2)　表中（ ）内の数字（%）は処理状況別の割合を示す。
(3)　稼動時間は少くとも一つの利用者ジョブが処理されていた時間。

第3図　昭和44年2月
2 job stream バッチ処理における日別 CPU 稼動効率



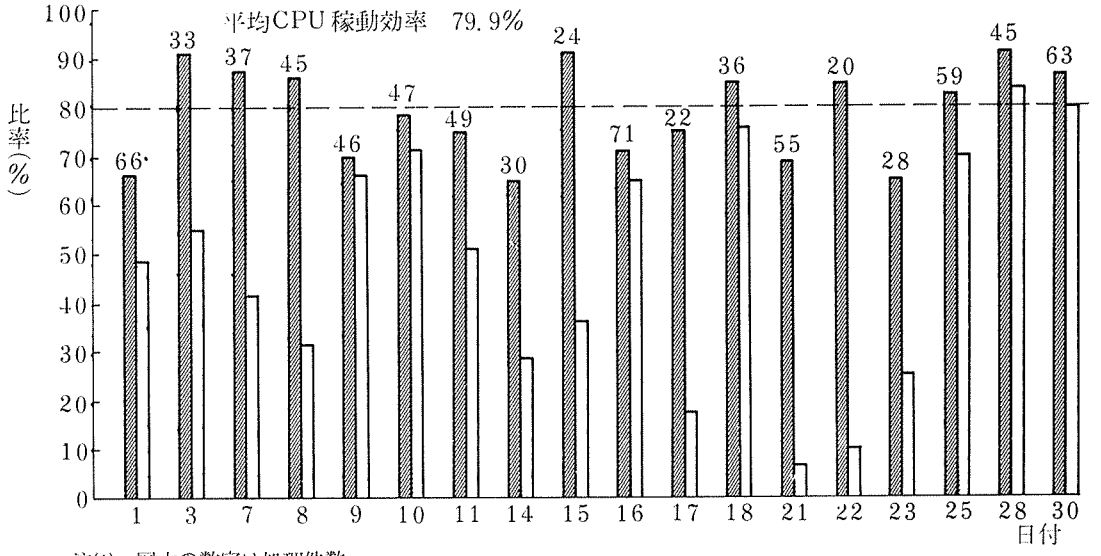
注(1)　図中の数字は処理件数

▨ $\frac{\text{CPU time 合計}}{\text{稼動時間}} \times 100$　　□ $\frac{(\text{USE time 合計}) - (\text{稼動時間})}{\text{稼動時間}} \times 100$

ただし，稼動時間は少くとも一つの利用者ジョブが処理されている時間。

第4図　昭和44年4月

2 job stream バッチ処理における日別CPU稼動効率



注(1)　図中の数字は処理件数

▨ CPU time 合計 / 稼動時間 ×100　　□ (USE time 合計)−(稼動時間) / 稼動時間 ×100

ただし，稼動時間は少くとも一つの利用者ジョブが処理されている時間。

ある程度必要である。さらには講習会あるいはシステム障害などのために，スケジュールが一部くずれる日もある。上に述べた数値はそれら一切を含めた1ヶ月の平均値である。（ただし，1日中を通じてスケジュールの異常な日は除外されている。）

ここにあげた2月および4月のデータは TSS サービス時の処理状況をも含めた1ヶ月の平均である。午後のバッチ処理の時間帯で並行処理される二つのジョブを，Lowerジョブ，Upperジョブと呼んでいる。（Lower，Upper はそのジョブが処理されるコア領域の位置を示している。）スケジュールの上では，この二種のジョブは性質の違うものであるから，区別して集計してある。

2月という時期は年度末に近く，短時間ジョブが多いため，第1表のデータは稼動効率が低い場合の例と考えてよいであろう。1件当りの平均 CPU time は3分24秒（4月では4分32秒）である。第1表でTSSサービス時の CPU 稼動効率は約60%で，バックグラウンド・ジョブが，おもに標準ジョブである関係から，この値は季節によってあまり大きな変動はないと考えられる。バッチ処理のみの時間帯では，4月の場合にくらべて多少効率が低く約75%である。この効率は必ずしも高いものではないが，Lowerジョブが4月の場合にくらべて，件数が約2.5倍になっている点を考慮すれば，2 job stream の効果はあがっていると見るべきであろう。なお，全時間帯での CPU 稼動効率は70%となる。

4月という時期は，他の時期にくらべて短時間ジョブが比較的少く，標準ジョブあるいは長時間ジョブの処理時間が大きなウェイトを占める。第2表でTSSサービス時の効率は約60%で2月と変わらない。バッチ処理のみを行なっているときの効率が約80%で，全時間帯では約73%となる。

3) CPU 稼動効率の吟味と新しい問題点

バッチ処理のみを行なっているときの CPU 稼動効率をより詳しく調べるために，日別に効率を調査した結果が第3図および第4図である。CPU 稼動効率は斜線の入った棒グラフで表わされ，斜線の入らない棒グラフは稼動時間中に二つのジョブが並行して処理されている時間の全稼動時間に対する割合を表わしている。書き込んである数値は処理件数である。

4月の場合の CPU 稼動効率を見ると，85%～90%の効率を得ている日が18日中6日ある。この効率は本来性質のよい長時間ジョブばかり処理してはじめて得られる値であって，モニター・タイムが零にはできないこと，ならびにたとえ 2 job stream であっても，入出力の行なわれるタイミングがランダムであれば CPU の遊び時間を完全に吸収することは不可能であることを考慮すれば，もはやこれが限界であると云わざるを得ないであろう。

残された問題はそれだけの効率が常時得られるというわけには行かないという点である。その理由は前にも触れておいたが，集約すれば主に二つの問題に帰着される。一つはオペレーションの態勢の問題であって，実際の処理の仕方をできるだけ運用スケジュールに一致させるようにし，さらにはオペレーターの介入時間を極力減らすことである。これは人間の側の問題として注意しなければならない問題であると同時に，OS の全体的な機能としてオペレーターの介入を必要としないシステムへと改善して行かなければならない問題でもある。しかし，オペレーションの柔軟性ということまで考えるとここには相対立する二つの要求が生じて来る。

もう一つの問題は，周辺機器特にシステム・ファイルに対して“待ち”が生ずることである。第1図に示すようにかなりの数のシステム・ファイルを使用するが，これらは必ずしもすべてが独立にアクセスできるとは限らない。特に短時間ジョブが多い場合にはシステム・ファイルにアクセスする頻度が多くなる。二つのジョブから同じようなタイミングでシステム・ファイルにアクセスすると，そのファイルが同じものである場合には無論のことであるが，別個のファイルでもファイルの組み合せによっては，一方のジョブが待たされることになる。これに対する改善は各システム・ファイルがなるべく独立に動作する機器におかれるようにすればよいわけで，少しずつではあるが，改善が実現されつつある。

以上二つの問題のために，稼動効率の低い日があったり，並行処理の時間帯が短いものが現われることになる。これらの問題については，センターで常に現状に対する反省を行ない，改善をつみ重ねて行かなければならないであろう。

4) 利用者の要求と稼動効率

2回にわたって稼動状況の解説を行なって来たが，ここで稼動効率の意味について考えておきたいと思う。これまで主に扱って来たのは CPU についての稼動効率であった。これがセンターの運用上一つの重要なファクターであることは確かである。しかし，これ自体が目的ではない。極論すれば，単に稼動効率をあげるだけならば，ジョブそれ自体の性質のよい長時間のものばかりを処理しておれば，スケジュールをとやかく云わなくても自然に80%程度の効率はあげられるのである。

しかし，利用者の側から見れば短時間のジョブについては早く返却してもらいたいのが当然の要求であって，そのためにはセンターでも短時間ジョブを多く処理しなければならない。また，稼動効率を上げるためにという理由から，入出力の量あるいは質について画一的な強い制限にしばられたのでは，本当に使いよいシステムとは云えなくなる。したがって，利用者の要求をできるだけ受け入れながらなおかつ能率よくシステムを稼動させることが，センター側で考えなければならないスケジュールの課題である。

現在のように数日分のジョブが常時たまっているような状態では，CPU 稼動効率が非常に大きなウェイトを持つことになるが，これが2次的な意味しか持つようにならないと望ましいシステムとは云えない。よほど長時間のジョブでない限り利用者としては結果が早く返って来てほしいわけで，これが運用スケジュールの第一目的となるべきである。無論長時間のジョブについては，ターン・アラウンド・タイムが長くなるのは止むを得ないことであるから，このようなジョブについては稼動効率が考慮されなければならないが，比較的短時間のものについては受け付けから処理に入るまでの時間は短い程望ましいわけで，そのために CPU の稼動効率が低下したとしても（オペレーターの介入，あるいは短時間ジョブばかりの並行処理によって）それは許されるべきであろう。

稼動効率云々の問題よりも，利用者の満足の行くサービスということが第一目的となるような，それだけの余裕のあるシステムの実現をセンターとしても大いに望んでいる。

（安井　　裕，山縣敬一）