データ入出力処理に関する管理システムの開発

杉江日出澄・尾崎正弘・貝谷邦夫・武藤三郎

情報処理教育センター

(1977年9月10日受理)

Development on Management System of Input/output Data Processings

Hidezumi SUGIE, Masahiro OZAKI, Kunio KAIYA, and Saburo MUTO

Center of Information Processing Education

(Received Stptember 10, 1977)

The management system for the job schedule, error information processings and system recovery, and accounting information processings are developed in N.I.T. computer system I. Jobs are scheduled to equalize service by the input data impartial restriction, based on the statistical data analysis, and by calculated group priority given to job classes. The accumulated total accountings since the beginning of the year may be informed for each individual user at every job process.

1. 緒　言

近年，コンピュータのハードウェアの進歩にともない，その利用分野の拡大が急速に進められており，その運用形態も多様化されてきている[1]。

大学等におけるシステム運用形態は，一般の企業におけるそれとはかなり異にする。学術研究や教育などが主たる使用目的となっているために，FORTRAN プログラムによる数値計算が大部分を占め，CPU の占有率が極めて高い。さらに，利用率が卒業時期に著しく関連し，特定の時期（年度末）に急激に増大し，ハードウェアの処理能力の数倍に達し，残留ジョブの累積によりターン・アラウンドが極度に長くなる現象が多々見うけられる。

一方，経理関係においては，各研究室ごとにおける計算機使用料のための予算管理があり，大学等では，年度初めからの累積使用料が管理面における重要な要素となっている。これら大学等の固有の条件下におけるシステムの効率・公平運用のための管理システムの開発が必要急務とされている背景がある。

本研究は，本学のシステムⅠ（H-8450 システム）における従来の管理システム[2]をさらに発展させ，特に年末の混雑時における公平利用の対策，および予算管理に関する対策について研究したものである。

2. 概　要

システム構成は，Fig. 1 に示すとおりであるが，このシステムの特長は，本学の特殊なニーズに対応すべく独特のスケジューリングを行う点，また，それに伴うシステムの負荷をほとんど無視できるようにしたことである。

さらに，このシステムの実施により，各利用者に対し1件ごとの課金情報が提供できるようになったことである。

このシステムは，4つのモジュールから構成され，スケジューリング処理，エラー情報処理および課金情報処

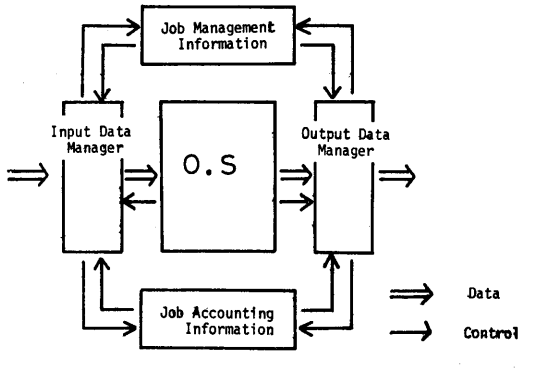


Fig. 1 Module organization of input/output data management system

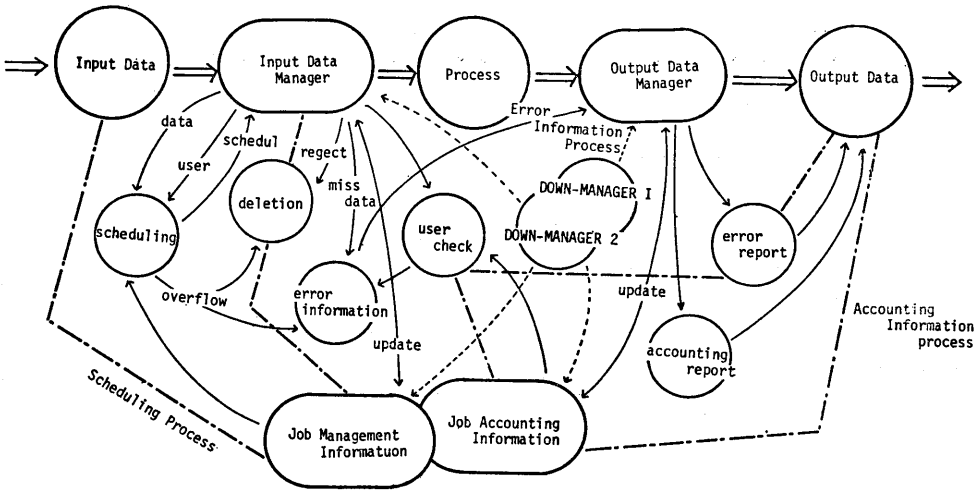


Fig. 2 Flow chart of data processings

理の3つの機能を持っている。データの処理過程を流れ図で示すと Fig. 2 のとおりである。

3. ジョブ・スケジューリング

3.1 期間とスケジューリング

システムIの利用者需要動向は，昨年度の実積から推定するとほぼ Fig. 3 に示すおりである。

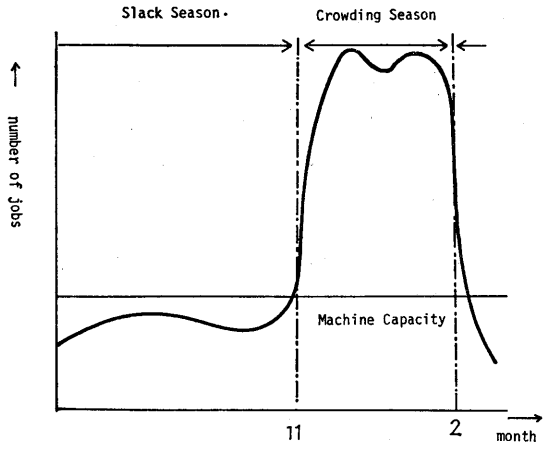


Fig. 3 Trend of number of jobs submitted monthly

大別して，1年を3月～10月の閑散期と11月～2月の混雑期に分けることができる。閑散期は，システムIの処理能力を越える需要があることはほとんどないと推定できる。すなわち，この期間は需要と供給のバランスがとれているため，特別なスケジューリングを必要としない。したがって，FIFO(First in First Out) 方式[3] で十分である。

一方，混雑期には，全般にわたり処理能力のほぼ3倍の需要がある。したがって，各利用者に対してシステム資源（ここでは，CPU タイム）を公平に配分するためのスケジューリングの必要性が生ずる。

以下そのスケジューリングの方法について述べる。

3.2 ジョブの入力制限

前述のごとく，有限な資源を考慮した上で，スケジューリング問題を考えるならば，利用者に対するサービスに一定のルールを設ける必要がある。

(1) 一日の投入ジョブの総本数による制限

これは，システムIの処理能力上限までのジョブ投入を許し，その限度を越えた場合は無条件でその日の入力を中止する。この方法によると，その日に投入されたジョブは，すべて処理を終えることができるが，各利用者に対して均等なサービスができなくなる可能性があり，かえって混乱を招くおそれがある。

そこで，各利用者に対するサービスの均等性という点を重視し，次の方法を考えた。

(2) 各利用者によるジョブの入力制限

各ジョブ・クラスに対して，Table 1 のような点数を持たせ，その合計値によって制限する方法である。

この方法によれば，特別なジョブ・クラスのみを使用する場合や各ジョブ・クラスを平均的に使用する場合のいずれにも平均的なサービスが行なえる。

また，各利用者は個々に与えられた持ち点までは，いつもジョブを投入することができる。

しかし，これを実施するにあたり，各ジョブ・クラスに対する点数の決定方法には，十分注意する必要がある。

Table 1 Mean values of CPU time for processings and points of each class for job-input control

JOB CLASS NAME	MEAN CPU TIME (MINITES)	POINTS
EXPRESS 1	0.31	5
EXPRESS 2	0.58	6
REGULAR 1	0.67	6
REGULAR 2	1.94	7
REGULAR 3	3.63	10
LONG	9.78	18

それは，点数の与え方によってサービスの均等化に大きな影響を与えるからである。

システムⅠにおける昭和51年度の利用者は約200名であり，毎日50名程度（約25%）が利用している。またシステムⅠの処理能力は，平均 CPU タイム（一日）約5時間である。これから，一日平均1人当たりの CPU タイムを求めると，

一日1人当たりの CPU タイム＝約6分

Table 1 の平均値（CPU タイム）は，多数のほゞ正規分布で近似できる実測値から得たものである。各ジョブ・クラスの平均 CPU タイムを，express 1, … long の順に $C_1, \cdots C_6$ とし，それらの各々について分散を求め，ほゞ95%の信頼区間の上限 R_i を求めれば以下のとおりである。

$<(x_{1j}-C_1)^2>=0.03$　　$R_1=0.6$

$<(x_{2j}-C_2)^2>=0.04$　　$R_2=0.97$

$<(x_{3j}-C_3)^2>=0.06$　　$R_3=1.14$

$<(x_{4j}-C_4)^2>=0.12$　　$R_4=2.62$

$<(x_{5j}-C_5)^2>=1.18$　　$R_5=5.76$

$<(x_{6j}-C_6)^2>=3.4$　　$R_6=13.21$

ここで，x_{ij} は，各ジョブの実 CPU タイムである。

$\overline{R}=4.05$

R_i 平均値 $\overline{R}$ を，基本値とし，それに各々 R_i の値を加え，それを各ジョブ・クラスの点数とした。

しかし，C_6 は 9.8 分であり，各利用者の1日当たりの CPU タイム（6分）を越えてしまう。そこで，C_6 を利用者に配分する CPU タイムの上限とし，そのウエイト値 18を各利用者の一日当たりの持ち点とした。

したがって，各利用者は一日に各ジョブの点数の合計が，18を越えるまでジョブを投入できることになる。

3.3 優先度によるジョブ・スケジューリング

OS に対する負荷を極力小さくするには，OS の持っている機能を最大限利用して，ジョブ・スケジューリングによる CPU タイムの専有率を，無視できるほど少なくすることである。そのため，システムⅠの優先度によるジョブ・スケジューリング機能を採用することにした。ただし，express 1，express 2 は，他のクラスと処理方法が異なり，スケジューリングの影響をあまり受けないため除外した。

(1) 毎日，優先度を変化させる方法

regular 1, 2, 3, long のジョブ・クラス順に優先度 (F_{ij}) を付ける。そして，1日ごとに優先度を低くしていく。すなわち，j 日目の優先度をジョブ・クラスごとに，F_{1j}, F_{2j}, F_{3j}, F_{4j} とすれば，(ただし，$F_{1j}>F_{2j}>F_{3j}>F_{4j}$) $j+1$ 日目は，$F_{1,j+1}>F_{2,j+1}>F_{3,j+1}>F_{4,j+1}$

ところが，$F_{2j}=F_{1,j+1}$, $F_{3j}=F_{2,j+1}$, $F_{4j}=F_{3,j+1}$ であるから，

$F_{1j}>F_{2j}=F_{1,j+1}>F_{3j}=F_{2j+1}>F_{4j}=F_{3,j+1}$ となる。

これは，3.2 に比べ，ジョブの入力制限を実施した場合，どのジョブ・クラスも必ずある一定のサイクルにより処理できる利点がある。

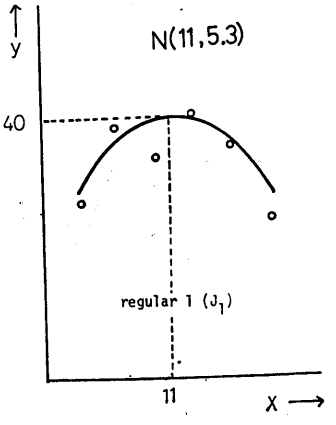
しかし，特定の利用者やジョブ・クラスに片寄りがある場合，その利用者またはジョブ・クラスが有利になる，いわゆる「早いもの勝ち」の現象が起きる。

特に，一部のジョブ・クラスに片寄った場合にその影響が著しい。そこで，各ジョブ・クラスに対して，均等なサービス（CPUタイム）をすべく，次の方法を実施した。

(2) ジョブ・クラスのグループ化（一日の処理グループ化）

過去の統計データに基づき，一日の処理ジョブのグループ化を行うことにより，サービスの均等化を図る。

各ジョブ・クラスを，regular 1, 2, 3, long の順に J_1, J_2, J_3, J_4 とすれば，昭和51年度の実積から $J_1 \sim J_4$ の一日当たりのジョブ数は，ほゞ正規分布で近似できる。**Fig. 4** にそれぞれについて平均値，分散および平滑化度数分布曲線を示した。



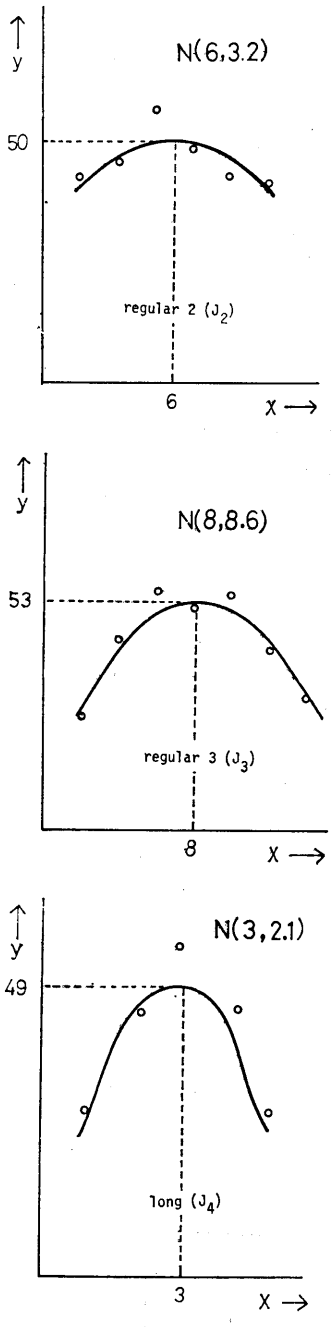


Fig. 4 Smoothed frequency distribution curves
y: frequency (days)
x: processed number of jobs in a day

Fig. 4 から，平均 CPU タイムと同様にして，ほゞ95%の信頼区間の上限 S_i を求めると，

$S_1=16$, $S_2=10$, $S_3=14$, $S_4=6$

しかし，S_i は一日のジョブの最多出現頻度であって，必ずしもジョブ・クラスのCPUタイムに比例していない。

そこで，一日の平均処理可能 CPU タイムを約5時間とすると J_1～J_4 までのジョブ・クラスに平均75分割当てることができる。それを 3.2で求めた Points で割って，一日の処理可能ジョブ数を求めると，

$j_1=13$, $j_2=11$, $j_3=8$, $j_4=5$

となり，j_3 を除きほゞ S_i に近い値となる。そこで，S_i を1つのグループとし，そのグループ内を同一優先度とする。しかし，この方法でも特定ジョブ・クラスのみが，急激に増加するとそのクラスの優先度が他のクラスに比べて，著しく低下する可能性がある。

これを防ぐ方法として，S_i に変化をもたせ，下記の判定を追加した。

$$\{M<S_n(F_{MAX}-F_n)\}\wedge\{(T_n-S_n)\leqq S_n/N\}$$

M：各グループに対する較差許容のための基準変数

S_n：各ジョブ・クラスのグループ内の規定ジョブ数

T_n：各ジョブ・クラスのジョブ数

F_n：各グループの優先度

F_{MAX}：$F_{MAX}=MAX\{F_{n,n=1\sim4}\}$

N：ジョブ数制限のための変数

各ジョブ・クラスの現在持っている優先度と最高位（現時点）の優先度の格差を比較し，グループ内の規定ジョブ数に乗ずる。そして，較差許容のための変数と比較し，それよりも大きければ，(S/N) まで前グループと同一優先度とする。

M, N は，システムの状態により変化する値である。

このスケジューリングを実施することにより，各利用者に公平なサービスが行なえるものと考える。

4. エラー情報処理機能

プログラムの論理ミスや利用者の登録番号ミス等，従来のシステムにある機能に，下記の機能を加えシステムの充実をはかった。

(1) 一日のジョブ入力制限によるエラー情報

各利用者個人に対するジョブ入力制限により，それを越えるジョブに対しては，メッセージを出力して各個人に通知する。

(2) 停電等の原因により電源が OFF になった場合

OS が管理する部分については OS によって回復されるが，データ入出力管理システムによって管理されているジョブ管理情報やジョブ課金情報は，本システムで回復する必要がある。そこで OS を起動させたのち，メモリ上の情報を回復するルーチン (DOWN-MANAGER

Ⅰ) を起動することにより，電源 OFF になる以前に終了したジョブの状態まで回復できる。

(3) データ入出力管理システムまたは OS のダウン

本システムまたは OS が，何らかの原因でシステム・ダウンした場合は，ダウンする直前の状態まで本システム関係の情報を回復ルーチン (DOWN-MANAGER Ⅱ) を，起動することによって回復できる。

(2), (3) のルーチンは，単独のバッチ処理用に開発されたものである。

5. 課金情報処理機能

1件のジョブを，規則に基づく算定方法により計算しそれを個人ファイルに登録し，また，課金情報として過去の累計料金とともに，1件の料金をジョブ結果の終りに毎回出力するものである。個人ファイルに累積された個人料金は，請求金額として半年毎に各支払責任者に通知される。さらに，個人ファイルや課金統計ファイルには，システムに関する種々の統計データが採取できるようになっている。

6. 結　言

電子計算機の急激な利用率の向上に対しては，ソフトウェアでもって補うには限界があり，ハードウェアのレベル・アップ以外には，満足な解決策がないであろう。

しかし，それは多額の費用を必要とし，容易に可能ならしめるものではない。ここでは，混雑状態を解決するには，各利用者に対する投入データの制限によって，システムのバランスを保ち，それにより，円滑なスケジューリングを行う方法をもって解決策とした。

これは，現状において全利用者に対するサービスの均等化により，各利用者に少なくとも混雑状態から解放させることが，出来るだろうと確信する。

電子計算機システムが，利用率に対し小規模であることから，今後新たな問題が生じてくると思われるが，それは，Try & Error として解決していかなければならない。

文　献

1) 木村一嘉・荒井祐蔵・佐野英之・千葉一郎：コンピュータの共同利用，日刊工業新聞社 (1971)
2) 杉江日出澄・尾崎正弘・貝谷邦夫・武藤三郎：名古屋工業大学学報，28, 371, (1976)
3) 池田克夫：オペレーティング・システム，日本コンピュータ協会，(1976)