OS カタログへのアクセスの解析と効率改善について†

金沢正憲†† 柴山 守†† 北川 一††

大規模の計算機システムを多人数で共同利用する場合，利用者の保存データセットを格納する数多くのダイレクト・アクセス装置の管理，および利用上の便宜，特に会話型処理におけるデータセットの扱いやすさに関して，データセットのカタログ簿（OS カタログ）およびカタログ管理が非常に有用である．しかし，OS カタログがシステムに唯一つ存在するリソースであるため，OS カタログへのアクセスがシステム全体の処理能力を低減させる一因となりうる．

この論文では，OS カタログにおいて必要な情報の検索および登録に要する平均時間を解析的に求め，さらに，通常の利用法で最もよく現われる形態に関して具体的装置を想定して解析を行った．また，実システムでも典型的な状態での測定を行い，解析の結果と比較した．解析の結果に基づいて，OS カタログの構造に変更を加えないで，アクセスに要する時間を短縮するブロック・リザーブ法について述べる．また，OS カタログの構造の特徴を捉え，カタログ管理の一部の簡単な変更を行い，アクセス回数を著しく減少させるエントリ・セーブ法について述べる．

1. はじめに

数多くの利用者が共同で使用する大規模計算機システムでは，カードリーダからジョブを入力しラインプリンタに出力するという単純な利用形態が，以前は大きな部分を占めていたが，個人的なライブラリ，データおよびプログラム等の保存のためにダイレクト・アクセス装置 (DASD) が，今日，非常によく使用されるようになってきた．その結果，個人的なデータセットの総数が増大し，バッチおよび会話型ジョブで個人的なデータセットへアクセスする回数が頻繁になっている．一方，DASD の台数の増加に伴う，ボリュームの種々の管理の問題とボリューム通番を指定する利用上の煩しさとを除去するため，データセットのカタログ簿，および，カタログ管理がオペレーティング・システム (OS) に用意されている．さらに，会話型処理を有する OS にはカタログ簿の使用を必須化しているものが少なくない．

カタログ簿は，個人的な保存用データセットを作成したり，参照したりするための割当て時に必ずアクセスされるデータセットであり，OS に唯一つ存在する．そのために，カタログ簿へのアクセス時間は，データセットの割当て時間に大きく影響を及ぼす可能性がある．事実，我々の計算機センターにおいて DASD を使用する利用者数，データセットの数，データセットの割当て回数の増大に伴い，システム全体の処理能力が著しく低下し，その原因の一つにカタログ簿へのアクセスが考えられた．

ここでは，我々のセンター* の OS (OSIV/F 4) におけるカタログ簿 (OS カタログ) へのアクセス時間を解析し，アクセス時間の短縮およびアクセス回数の減少をもたらす方法について述べる．

2. OS カタログの構造とアクセス法

a) OS カタログの構造

OS カタログに登録されるデータセット名は，英字で始まる 8 文字以内の英数字からなる単純名をいくつかピリオドでつないだ形式，すなわち，

単純名 1. 単純名 2.…. 単純名 i $(i \geq 1)$

と表わされる．また，単純名 j を第 j レベルの単純名と呼ぶ．OS カタログへのデータセット名の登録は，単純名を単位とした階層構造に基づいている．**図 1** に OS カタログの階層構造を例で示す．なお，データセット名は，DSN 1. DSN 2. DSN 3 とする．

OS カタログは，8 バイトのキー部と 256 バイトのデータ部を有するデータセットである．データ部はインデックス・ブロック（単にブロックともいう）と呼ばれ，一般に複数個の，同一レベルの単純名と必要な情報からなるエントリが格納されている．図 1 におい

† An Analysis and Some Improvements of OS Catalogue Access by MASANORI KANAZAWA, MAMORU SHIBAYAMA and HAJIME KITAGAWA (Data Processing Center, Kyoto University).

†† 京都大学大型計算機センター

* 大学間全国共同利用の大型計算機センターで，利用登録者数 2,500 以上，年間約 50 万件のバッチ・ジョブと約 20 万件の会話型ジョブを処理している．

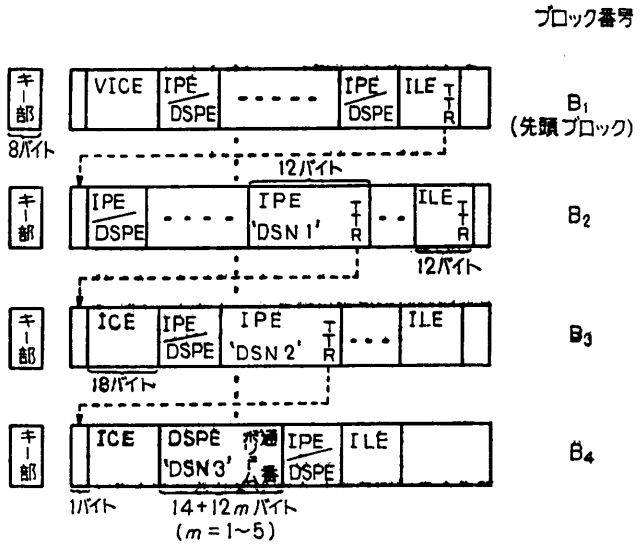

図 1 OS カタログの階層構造

Fig. 1 Structure of OS catalogue.

て，ブロック B_1，B_2 には第1レベルの単純名をもつエントリのみが格納され，ブロック B_3 には第1レベルの単純名が DSN 1 であるデータセット名の第2レベルの単純名をもつエントリのみが格納されている．同様に，ブロック B_4 には，第1レベルの単純名が DSN 1 で，かつ，第2レベルの単純名が DSN 2 であるデータセット名の第3レベルの単純名をもつエントリのみが格納されている．

OS カタログの先頭ブロックの最初には VICE(Volume Index Control Entry) があり，最終ブロック・アドレス (TTR)，先頭の空きブロック・アドレス等の情報を有する．VICE に続いて，第1レベルの単純名を単なるバイナリ・データ（符号なし）と見なした場合の昇順に，エントリが登録されている．

当該レベルが最終レベルの単純名のときには DSPE (Data Set Pointer Entry) が，次レベルがある単純名のときには IPE (Index Pointer Entry) が登録されている．DSPE には単純名とボリューム通番等が，IPE には単純名と次レベルの先頭ブロックのアドレスが格納されている．各ブロックの最後には，ILE (Index Link Entry) があり，同一レベルの次のブロックのアドレス，および，ブロック中の最終の単純名が格納されている．当該レベルの最終ブロックのときは，単純名の代りに最大値 $(FFF\cdots F)_{16}$ が入る．

第2レベル以降のブロックは先頭の VICE が ICE (Index Control Entry) になる以外，第1レベルと同じである．

キー部には，物理的に次のブロックが同一レベルで，かつ，昇順に並べられた続きのエントリが登録されている場合は，当該ブロック内の最終エントリの単純名が格納されている．そのほかの場合は，最大値 $(FFF\cdots F)_{16}$ が格納されている．

b) アクセス法

OS カタログにアクセスして必要な情報を検索または登録するには，その階層構造に基づいて，順にエントリを追う方法が採られる．また，実際のアクセス回数を減少させるために，キー部を利用し，入出力コマンドのキー・サーチ (条件は high or equal) を用いている．

キー・サーチは，CPU の介入なしに，条件の満足されるキー部が現われるまで次々と連続的に空読みし，条件の満足されたデータのみが読込まれるので，余分な回転遅れ時間を必要としない．したがって，連続的なブロックの中から必要な情報を探し出すには非常に効率が良い．

検索時において，次のようなアクセスの方式が採られている．キー・サーチ* を始めるブロックのアドレスから，条件 high or equal でキー・サーチを行う．サーチされたブロックに関して，求める単純名をもつエントリが入っている可能性があるブロックか，ブロックが不連続でキーが $(FFF\cdots F)_{16}$ であるために読込まれたブロックかを ILE の内容と比較して調べる．もし後者であれば，ILE から次にキー・サーチを開始するブロックのアドレスを求めて，再度キー・サーチをする．前者の場合には，単純名の一致するエントリがあるか調べる．

一致したエントリが IPE ならば，次レベルの先頭ブロックのアドレスを求めて再度キー・サーチをする．DSPE ならば，最終的に必要な情報であるボリューム通番が求められ，検索が終了する．図2を参照されたい．

登録時のアクセスの方式は，検索時と途中まで同様である．図2において，KS と同一名のエントリが存在しなくなった時点からアルゴリズムが異なる．第 i レベルで不一致になると，$I, I-1, \cdots, i+1$ の各レベルのブロックのために，OS カタログ内の先頭から最も近い空きブロックを $(I-i)$ 個探し，各ブロックを

* キー部のサーチと条件が満足されたブロックの読込みは，チェーン・コマンドを用いて一回転の遅れなしに行われるので，以後この一連の動作をキー・サーチという．

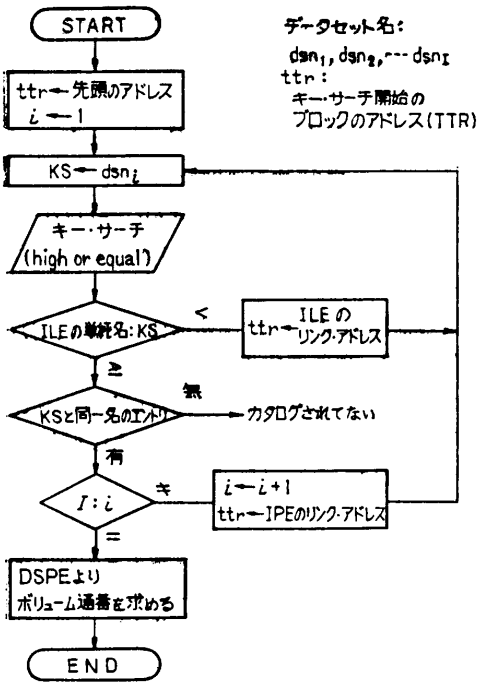

図 2　検索時のアルゴリズム

Fig. 2　Access logic for reference to OS catalogue.

作成し，VICE を更新する．次に，不一致になった第 i レベルのブロックに，単純名 dsn_i を持つエントリを昇順になるように挿入する．この挿入によって最後のエントリがはみ出せば，リンクされた次のブロックの先頭のエントリとして挿入する．挿入とシフトは必要ならばリンクされた最終ブロックまで続けられる．最終ブロックではみ出せば，空きブロックを探し，最終ブロックとして追加される．

3. アクセス時間の解析

OS カタログにアクセスして，ボリューム通番を検索する，あるいは，エントリを登録するのに要する時間を求めるが，CPU 時間は DASD へのアクセス時間に比べて十分短いので，DASD 上で費やされる時間を解析する．

DASD の特性は次のとおりとする．

回転時間：$2r$ 秒，

シーク時間：$S(j)$秒，(j：移動シリンダ数)，

転送時間：W 秒，

シリンダ容量：C トラック．

OS カタログについては次のとおりとする．

トラック内のブロック数：L 個／トラック，

第 i レベルのブロック数*：D_i ブロック，

第 i レベルのブロックの

不連続な確率*：　p_i.

平均シーク時間を s とすると，ある 1 つのブロックにアクセスする平均時間 A は，

$$A=s+r+W \qquad (1)$$

となる．

検索時間および登録時間を求めるが，OS カタログのある DASD へのアクセスはほかになく，また，キー・サーチの途中でシリンダの移動はないと仮定する．

(a)　平均検索時間

第 1 レベルにおいて，x_1 番目のブロックにあるエントリを検索するには，シークを伴う読込みが x_1p_1 回で，前のブロックに続く読込みが $x_1(1-p_1)$ 回であるため，平均検索時間 $T_1(x_1, p_1)$ は

$$T_1(x_1, p_1)=\sum_{l=1}^{x_1p_1} A+\sum_{l=1}^{x_1(1-p_1)} W$$
$$=x_1\{p_1(s+r)+W\}, \qquad (2)$$

となる．同様に，第 i レベルにおける x_i 番目のブロックにあるエントリへの平均検索時間 $T_i(x_i, p_i)$ は，

$$T_i(x_i, p_i)=x_i\{p_i(s+r)+W\}, \qquad (3)$$

となる．したがって，$\boldsymbol{x}=(x_1, x_2, \cdots, x_I)$ に格納されたエントリの平均検索時間 $T(\boldsymbol{x}, \boldsymbol{p})$ は

$$T(\boldsymbol{x}, \boldsymbol{p})=\sum_{i=1}^{I} x_i\{p_i(s+r)+W\}, \qquad (4)$$

となる．ただし，$\boldsymbol{p}=(p_1, p_2, \cdots, p_I)$.

データセット名の付け方は，利用者が任意に命名でき，かつ，利用者間で同一名とならないように，第 1 レベルの単純名を利用者固有の名，たとえば，利用者の識別名とし，第 2 レベル以降は利用者の任意とする場合が多い．通常の利用においては，記述の簡単さにより，第 3 レベルまでであり**，また，我々のセンターでは利用者ごとのデータセット数は比較的少ない．

そこで，$I=3$，$D_2=D_3=1$ とすると，$x_2=x_3=1$，$p_2=p_3=1$ と定数化され，式(4)は

$$T(x_1, p_1)=x_1\{p_1(s+r)+W\}+2(s+r+W), \qquad (5)$$

となる．平均検索時間は，第 1 レベルの不連続さ p_1 に比例する．

(b)　平均登録時間

---

* 第 1 レベルから第 $i-1$ レベルまでのエントリに依存するが，記述は簡便にされている．

** OSIV/F 4 の会話型処理においては，利用者識別名と，内容を示す名称が付加されるため，データセット名は

利用者識別名．単純名．内容識別修飾名

という形式になる．バッチ処理においても，会話型処理での扱いやすさを考慮して，同形式で命名していることがほとんどである．

まず，第iレベルにおいて，1番目からz_i番目までのブロックを更新するのに要する平均時間を求める．シークを伴う読込みはz_ip_i回で1回当りのアクセス時間は平均Aかかり，前のブロックに続く読込みは，$z_i(1-p_i)$回で，前ブロックの書込み後1回転して読込むため1回当り$2r+W$時間かかる．書込みはz_i回で当該ブロックを読込んで後に1回転要するため1回当り$2r$時間かかる．したがって，平均更新時間$U_i(z_i, p_i)$は，

$$\begin{aligned}U_i(z_i, p_i)&=z_ip_i(s+r+W)\\&\quad+z_i(1-p_i)(2r+W)+2rz_i\\&=z_ip_i(s-r)+z_i(4r+W),\end{aligned}\qquad(6)$$

となる．

第jレベルにおいて該当エントリがなく，y_j番目のブロック（Yブロックと呼ぶ）に挿入し，(y_j+1)番目からD_j番目までのブロックがすべて更新され，D_j番目のブロックからのはみ出しはないと仮定する．$\boldsymbol{y}=(y_1, y_2, \cdots, y_j, \cdots, y_I)$なるエントリの平均登録時間$V_j(\boldsymbol{y}, \boldsymbol{p})$は，

$$\begin{aligned}V_j(\boldsymbol{y}, \boldsymbol{p})&=\sum_{i=1}^{j}T_i(y_i, p_i)+U_j(D_j-y_j, p_j)\\&\quad+\sum_{i=j+1}^{I}M_i+N+u_Y\end{aligned}\qquad(7)$$

となる．ただし，M_iは空きブロックのサーチと第iレベルのブロック作成時間で，NはVICEの，u_YはYブロックの更新時間である．Nとu_Yは，シークを伴うブロックの書込みであるため，

$$\left.\begin{aligned}N&=s+r+W\\u_Y&=s+r+W\end{aligned}\right\}\qquad(8)$$

となる．空きブロックの平均サーチ時間はレベルに関係がないので，E時間とすると，式（9）のようになる*．

$$M_i=\begin{cases}E+(s+r+W) & (i=I\text{ のとき})\\ E+2r & (i\neq I\text{ のとき})\end{cases}\qquad(9)$$

検索時間の場合と同様に，$I=3, D_2=1$と仮定し，$j=1,2$の場合について式(7)を考える．

$j=2$は，第1レベルがすでに登録されている場合で，$y_2=1, p_2=1$と定数化されるので，式(7)は

$$\begin{aligned}V_2(y_1, p_1)&=T_1(y_1, p_1)+T_2(1, 1)\\&\quad+E+3(s+r+W)\\&=T_1(y_1, p_1)+E+4(r+s+W),\end{aligned}\qquad(10)$$

となり，平均検索時間に定数を加えた値となる．

$j=1$は，第1レベルから登録されていない場合で，式(7)は，

$$\begin{aligned}V_1(y_1, p_1)&=T_1(y_1, p_1)+U(D_1-y_1, p_1)\\&\quad+2E+3s+5r+3W\\&=p_1\{D_1(s-r)+2ry_1\}+4r(D_1-y_1)\\&\quad+D_1W+2E+3s+5r+3W,\end{aligned}\qquad(11)$$

となる．

平均登録時間において，検索部分$(T_1(y_1, p_1))$はp_1に比例するが，更新部分$(U(D_1-y_1, p_1))$は式(6)からsとrの時間によってp_1に比例したり，反比例したりする．このために，p_1とどのような関係にあるかは一定しない．

次に，実際にどの程度の時間を要するか我々のセンターの装置を想定して計算をしてみる．

ブロックはnシリンダに一様に分布し，jシリンダ離れた位置へのシーク時間を$(\alpha j+\beta)$とすると，

$$\begin{aligned}s&=\frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{n}\left\{\sum_{j=1}^{n-i}(\alpha j+\beta)+\sum_{j=1}^{i-1}(\alpha j+\beta)\right\}\right)\\&=\frac{\alpha}{3}\left(n-\frac{1}{n}\right)+\beta\left(1-\frac{1}{n}\right)\end{aligned}\qquad(12)$$

となる．

OSカタログおよび格納されているDASDの特性は次のとおりである．

α: 0.0001秒，　β: 0.006秒（ただし，$j\leq 100$），
r: 0.0084秒，　C: 19トラック，
L: 28個，　W: $2r/L$，
ブロック長: 256バイト．

利用者は平均10個のデータセットを持ち，第2レベルで同一名を用いることはないと仮定すると，1ブロックに20個（最初のブロックのみ18個）のIPEが入るので，第2レベルのブロック総数は，$20D_1-2$，第3レベルのブロック総数は，10×（第2レベルのブロック総数）となる．したがって，使用シリンダ数nは，

$$\begin{aligned}n&=\{(\text{総使用ブロック数}-1)/L\cdot C\}+1\\&=\{221D_1-23)/(28\times 19)\}+1,\end{aligned}$$

となる．ただし，{ }はガウス記号である．

上記の条件を式(5)，(11)に代入し，p_1, D_1をパラメータとして，Tとx_1，V_1とy_1の関係を図3，4に示す．また，式(10)のV_2とy_1の関係は図3と同等になる．

この解析の結果，多人数が個人のデータセットを使

* 最初に作成される第Iレベルはすでに空きブロック位置が判っているので，シークを伴うブロックの書込みとなる．それ以外のレベルは空きブロックを探して，1回転後に書込まれる．なお，最後に行われるVICEへ登録する空きブロックのサーチは便宜上$i=I$に含めた．

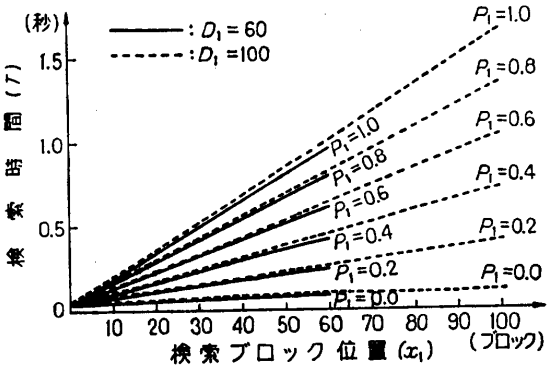

図 3　平均検索時間

Fig. 3　Average time for reference to OS catalogue.

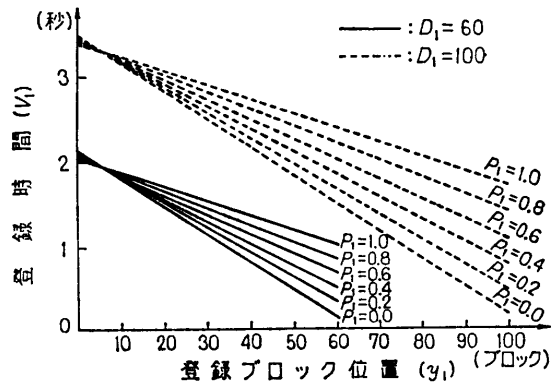

図 4　平均登録時間

Fig. 4　Average time for entry to OS catalogue.

用し，各利用者ごとには余りデータセット数が多くない場合，OS カタログのアクセス時間に関して次のことがいえる．

（i）　第1レベルのブロックの不連続性に比例して，検索時間および第2レベル以降へのエントリ登録時間の平均が長くなる．

（ii）　第1レベルに新しくエントリを登録する場合，挿入すべきブロックの位置が論理的に後方のブロックになるにつれて，不連続性が平均登録時間の増大をもたらす．

解析結果を検証するために，我々のセンターのシステムでユティリティ・プログラムを実行させ，アカウント・データから求めた測定結果を**表1**に示す．この測定では，第1レベルのブロック数 (D_1) は 60 である．

平均検索時間に関しては，解析結果と測定結果が，ほぼ一致している．平均登録時間に関しては，測定システムでは更新ブロックの物理的に次のブロックを読

表 1　アクセス時間の測定結果

Table 1　Observerd time for access to OS catalogue.

第1レベルのブロック位置	p_1=1.0		p_1=0.0	
	検　索	登　録	検　索	登　録
先頭ブロック	0.2 秒	3.8 秒	0.1 秒	3.2 秒
最終ブロック	0.9 秒	2.4 秒	0.2 秒	0.2 秒

注 1. 測定時間の精度は 0.1 秒

込むため*，値は異なるが，先頭ブロックに挿入する場合は不連続性によって余り変らないのに対して，最終ブロックに挿入する場合は，連続性によって時間の短くなる程度が大きいという傾向が見られる．

OS カタログにおいては，最初のブロックに 18 個，以後のブロックに 20 個の IPE が格納される．また，想定した DASD には，1シリンダに 532 ブロック格納されるため，キー・サーチにおいてヘッドのシリンダ間の移動を無視したことによる影響は実際的にはほとんどない．

4. アクセス時間の短縮法

前述したように同一データセット名を回避するため，および，記述の簡便化のため，第1レベルの単純名が利用者の識別名と1対1になっていることが多い．このような場合，第1レベルの登録は利用者に対して初めてデータセットをカタログする際の1回でよい．一方，式(5)，(10)で表わされるアクセスは，データセットの参照または作成ごとに生じるので，このアクセスがほとんどと考えられる．そこで，アクセス時間の短縮を図るためには，$T_1(x_1, p_1)$ の値を小さくすることが効果的である．

（i）　ブロック・リザーブ法

アクセス時間の短縮には，すでに解析したように第1レベルのブロックの不連続性を小さくすることが有効である．しかし，2(b) の登録時のアクセス法で説明したように第1レベルのブロックに IPE を作成すると，必ず最も先頭の空きブロックが第2レベル以降のブロックとして割当てられ各レベルのブロックが混在するため，第1レベルのブロックが連続して取られることは確率的に少ない**．

ブロック・リザーブ法は，エントリが常に昇順に並

* 次のブロックが空きならば，そのブロックにあふれたエントリを入れて連続性を期待するが，この測定ではすべて使用中のため無駄な読込みとなっている．

** p_1 は通常 1.0 に近い値で，更新ブロックの物理的に次のブロックが空きブロックかをチェックして連続化を図っても我々のセンターでは 0.8 以上であった．利用者ごとのデータセット数は少ないため，p_0, p_2 はほとんど 1.0 であった．

べられることとDSPEにはリンク・アドレスがないというOSカタログの構造を利用して，第1レベルのブロックを必要個数だけ連続して確保する方法である．

まず，何も登録されていないOSカタログに実際に使用される第1レベルの単純名よりも昇順で後になるような第1レベルのみからなるデータセット名（単純名）を必要個数だけシステム・ユティリティで登録する．これをダミー・エントリと呼ぶ．その後，通常の利用において第1レベルのエントリが登録されるごとに，そのエントリはダミー・エントリより前に登録され，ダミー・エントリが後へシフトされる．ダミー・エントリの最初の登録により，あらかじめ確保された連続のブロックに，実際に検索される第1レベルの単純名が格納されうる．すなわち，必要な範囲において，$p_1=0$となる．

一方，ブロック数D_1が増大するため (11)の値が大きくなるが，第1レベルのエントリの登録数が増大する適当な間隔ごとにダミー・エントリを後方から消去すれば，増大を抑えることができる．

この方法は，第1レベルの単純名に対する制限はきわめて緩いため，一般のシステムに十分かつ容易に適用できる．

（ii） エントリ・セーブ法

会話型処理ではデータセットを数多く扱い，かつ，ほとんどがカタログを前提としている．しかも，第1レベルの単純名は，通常ログオン時の利用者識別名である．

エントリ・セーブ法は，ログオン時にOSカタログへアクセスし，第1レベルのエントリ（12バイト）をメモリ上にログオフまで保存する方法である．

エントリの位置は他利用者によるOSカタログへの登録等によって変ることもあるが，エントリ中のリンク・アドレスは変化しない．したがって，セーブされたエントリの内容は保証される．

一方，OSカタログへのアクセス時に，第1レベル

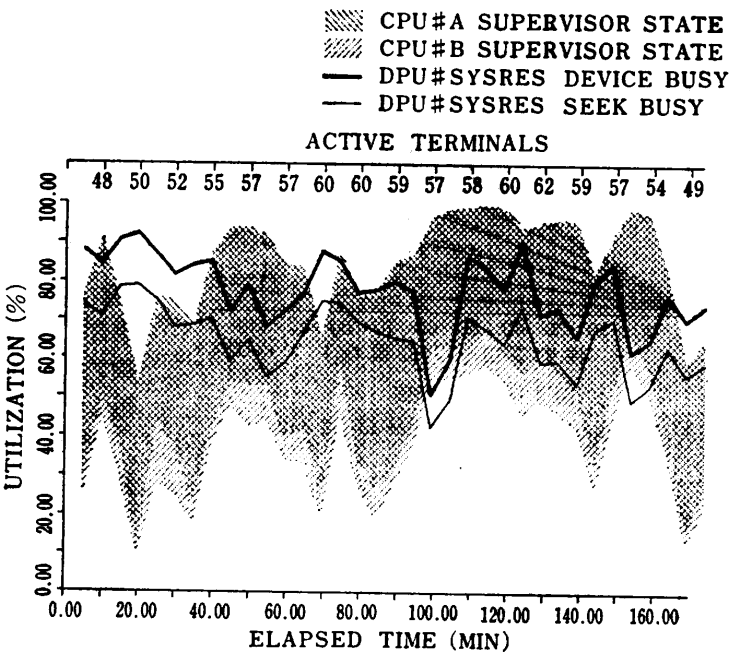

図 5(a) CPUとDASDの使用率（改善前）

Fig. 5(a) Utilization of CPU's and DASD before improving OS catalogue access.

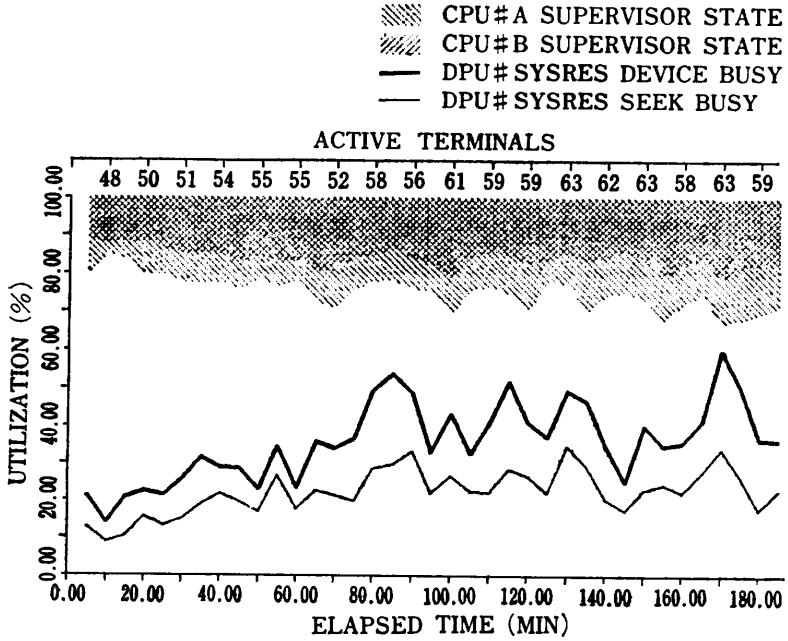

図 5(b) CPUとDASDの使用率（改善後）

Fig. 5(b) Utilization of CPU's and DASD after improving OS catalogue access.

の単純名（利用者識別名）とセーブされたエントリの単純名が一致すれば，第1レベルの検索は不要で，

$$T_1(x_1, p_1)=0,$$

となり，式(5)，(10)は

$$T(x_1, p_1)=2(s+r+W) \quad (13)$$

$$V_2(y_1, p_1)=4(s+r+W)+E \quad (14)$$

となる．

第1レベルの検索はログオン時の1回は必要であるが，以後のアクセスにおいては，この方法により第1レベルの検索が省略できる．

エントリ・セーブ法は，ある第1レベルのエントリ(利用者識別名）から論理的に続く第2レベル以降の(利用者固有の独立した）ブロックが構成されるというOSカタログの特徴を利用しており，多数の利用者によるシステムの共同利用に適した方法である．

5. おわりに

我々の計算機システムでは，アクティブなTSS端末数とバッチ・ジョブの増加に伴い，応答時間の著しい増大とバッチ・ジョブのスループットの低下を経験した．その時に，ハードウェアおよびソフトウェア・モニタによる測定で，OSカタログのDASDの異常に高い使用率が観測され，これが処理能力を低下させる主原因であることが判明した．

ここでは，既存のカタログ管理をほとんど変更しないで改善する方法を検討した．しかし，OSカタログの構造と多重仮想空間の特徴を発揮させ，第1レベルのエントリを1つの空間上に展開し，あるDASDにそのバックアップを行えば，常にエントリ・セーブ法的になるため，アクセス時間を画期的に短縮できる．しかし，カタログ管理をかなり変更する必要があるため，ここではインプリメントしなかった．

現在では，個人用のデータセットを使用する利用者は1,600余人に増大し，一方，1人当り平均9個強保存している．このような状況では，ブロック・リザーブ法とエントリ・セーブ法による効率改善の効果は顕著に見られる．

なお，ブロック・リザーブ法は我々のセンターにより，エントリ・セーブ法は富士通(株)によりシステムに適用された．最後に，参考までに，改善前と後の運用システムにおけるCPUとSYSRES(OSカタログ用のDASD)の使用率を図5に示す*．図5(a)ではSYSRESがほかのDASDに較べて極端に高い使用率であった．図5(b)ではSYSRESの使用率は低下し，しかも1.5～1.7倍高速のCPUに置換えられているために，実質的な改善の効果はより大きいと考えられる．しかし，運用中の状況であるために必ずしも同一ワークロードではないが，ほかの日もほぼ同様の状況であった．

この研究は京都大学大型計算機センターの開発計画の下で行われた．開発協力者として日頃熱心にご討論を載いた同センター飯田記子助手，ご協力を載いた同センター今城一夫氏および富士通の関係者各位，ご助言を賜った京都大学工学部萩原宏教授に心から感謝の意を表します．

* CPUのスーパバイザ・モードによる使用部分（斜線部）の下方はプロブレム・モードによる使用を示す．

参考文献

1) 富士通：OSIV/F 4 解説，富士通マニュアル，64 SG-1001.

(昭和55年3月17日受付)
(昭和55年9月18日採録)