バーチャル・メモリ方式の計算機のメモリ・ハイアラルキー†

西　本　哲　則††

Abstract

The performance of a time sharing system having paging mechanism was analyzed. From the data of paging pattern, it is found that the swapping probability is approximated by an exponential function of the number of pages which one task holds in main memory.

On the basis of the above mentioned experimental results and the theory of cyclic queue applied to the time sharing system, the ratios of CPU idle time and operating system time are expressed by the following parameters; the capacity of the main memory, the paye swap time between two memories mentioned above, the number of running tasks and average execution time per instruction.

1. はじめに

タイム・シェアリング・システムは，電子計算機を複数個のユーザーが使用することによって，電子計算機の使用効率を増大することを一つの目的としている．

しかし，現実には，電子計算機の遊んでいる時間（アイドル・タイム）とオペレーテング・システムを走っている時間をあわせた時間いわゆるオーバーヘッドが増大して効率が悪くなり，目的と逆の状態になっていた．たとえば，文献1)に報告されているようにタスクが全然ない時間すなわちユーザーが1人も計算機本体を使用していない時間を除いたうちでタスク上を走っている時間の割合が 8～15%，オペレーティング・システムを走っている時間の割合が 35～45%，残りの 30～40% が磁気ドラムからの転送待ちのためのアイドル・タイムである．

バーチャル・メモリ方式のコンピュータ・システムにおいて，並列して走るタスクが多いほど，ミッシング・ページ・フォールト間の時間が短くなることに起因してオーバヘッドが大きくなる．

この報告は，以上のメカニズムをモデルにより明らかにすると同時にオーバーヘッドを減少するためにはバーチャル・メモリ方式の計算機のシステム構成をどのようにすべきか，特に主記憶装置の容量と，中央処理装置のスピード，補助記憶装置（磁気ドラム，磁気デスク）のスピード，タスクの数をどのように選べばよいかを明らかにしたものである．

2. タイム・シェアリングシステムのモデル化

タイム・シェアリング・システムのオーバーヘッドは大きく3つに分けられる．

（1） 中央処理装置のアイドル・タイムのうちで主記憶装置とドラムなどの補助記憶装置との間のデータ転送のために生じるアイドル・タイム．

（2） オペレーティング・システム上を走っている時間．

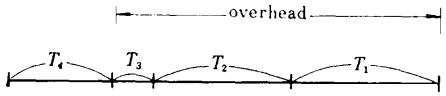
（3） すべてのタスクが端末入出力装置にあるか，タスクが0であり中央処理装置にあるか，タスクが0であり中央処理装置がなにもすることがないためのアイドル時間．

（3）は中央処理装置，補助記憶装置のスピード，あるいは主記憶装置の容量などに関係するよりむしろタイム・シェアリング・システムのおかれている環境に依存するものであると考えられるからこの報告では考慮しない．そして以下タスク数というときには端末入出力装置にあるものは除外し，中央処理装置待ちか，

† Memory Hierarchies of Virtual Memory Computer by Tetsunori Nishimoto (Central Research Lalroratory, Hitachi Ltd. (present Panafacom Ltd.))

†† 日立製作所(株)中央研究所，現在はパナファコム(株)に勤務

補助記憶装置待ちのタスクだけを対象とすることにする．(1)，(2)のオーバーヘッドについて解析するのであるが，(1)と(2)を同時に取扱おうとすると複雑になるので，ここでは(1)のオーバーヘッドを考えているときには(2)のオーバーヘッドを 0，(2)のオーバーヘッドを考えているときには(1)のオーバーヘッドを 0 と仮定する (Fig. 1)．この仮定ではオーバーヘッドが大きいときには，他方を考慮した場合と比較して少し誤差がでてくる．しかし，オーバーヘッドを測定したものをみると[1] 測定した時点によってかなりゆらぎがあり，現実のオーバーヘッドは正確にはとらえないことと，オーバーヘッドが小さいようにシステムを構成すべきであるからオーバーヘッドの大きいところは，それほど正確に推定する必要がないという 2 点から上記の仮定で十分であると思われる．(1)，(2)をともに小さくすれば(1)と(2)を同時に扱ったものも（他方への影響が小さくなるために）小さくすることができると考えてよいだろう．



T_1 : CPU idle time
T_2 : time running on programs of processing page fault and trap drum channel
T_3 : time running on operating system except $T2$
T_4 : time running on tasks

$$OH1 = \frac{T_3}{T_1 + T_4}$$

$$OH2 = \frac{T_2}{T_2 + T_4}$$

Fig. 1 Definition of overhead

$OH1$ をオペレーティング・システム上を走る時間(すなわち(2)) を無視したときの中央処理装置のアイドル・タイム（すなわち(1)) の全時間に占める割合とする．$OH2$ をアイドル・タイム(すなわち(1)) を無視したときのミッシング・ページ・フォールト(主記憶装置に必要なページがないためにおこる割込) の処理のプログラム上を走る時間と，トラップ・ドラム・チャネル（1 ページを補助記憶装置と主記憶装置とのあいだで転送し終ったときにチャネルから起こす割込) の処理プログラム上を走る時間の和の全時間に対する割合とする．

ただし，オペレーティング・システム上を走る時間のうちの大部分を占めるものは，補助記憶装置と主記憶装置とのあいだでページの入れかえをするためにプログラム上を走る時間である．文献1)によるとその割合は，オペレーティング・システム上を走る時間のうちの約 70% (即ち Fig. 1 で $T_2/(T_2+T_3) \fallingdotseq 0.7$) にもなる．残りのスーパーバイザーコイルなどの処理時間はタスクが走るステップ数に比例して大きくなり，主記憶装置の容量と直接関係しないからここではオーバーヘッドと考えないこととする．

もし必要であれば，$OH1$，$OH2$ を求めるときにはタスクの走る時間の中に含めておいて，あとでそれを一定の割合で分割すればよい．$OH1$ を算出するのに，タイム・シェアリング・システムを次のようにモデル化する．Fig. 2 はモデルとした電子計算機の記憶装置のメモリー・ハイアラルキーをあらわしたものである．主記憶装置へは情報がページと呼ばれる一定の単位ごとに補助記憶装置，たとえばディスク装置あるいはドラム記憶装置から読み込まれる．もしも主記憶装置にないページがアクセスされるとミッシング・ページ・フォールトが起こり，補助記憶装置から必要なページが読み込まれる．

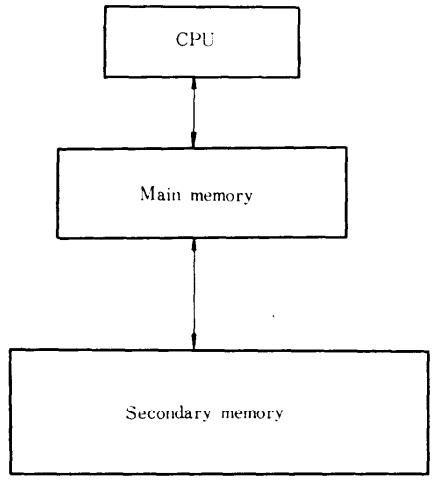


Fig. 2 Memory hierarchy

Fig. 3 はタスクの動きをモデル化したものである．このモデルのなかにあるすべてのタスクの数を N であらわす．N 個のタスクの各々は CPU と書かれているボックスにはいり，タスクがプログラム上を走っているうちに主記憶装置上にないページを要求したとき，補助記憶装置から主記憶装置へのページの転送を必要とするため，queue 21, queue 22, …, queue 2M のいずれかに移動する．ただし M は補助記憶装置で

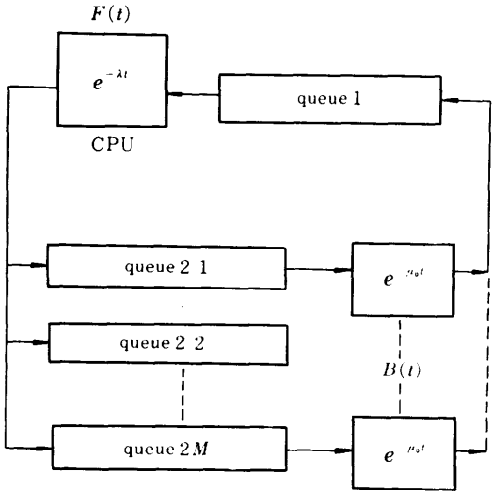


Fig. 3 Block diagram

独立に動作可能なものの数である．補助記憶装置から主記憶装置へページの転送が終了するとそのタスクは queue 1 にならぶ．そして queue 1 上でそれ以前にならんでいるタスクがなくなると，中央処理装置でタスクは実行される．タスクが queue 1 から，中央処理装置に移った時刻からミッシング・ページ・フォールトを起こす時刻までの時間の分布関数は平均が $1/\lambda$ の指数分布であると仮定する．また1台の補助記憶装置から主記憶装置へページを転送する時間の分布関数 $B(t)$ も平均が $1/\mu_0$ の指数分布であると仮定する．この仮定も前記オーバーヘッド(1),(2)の場合と同様な議論で十分である．

中央処理装置から queue 21, queue 22, …, queue $2M$ へタスクが移動するときには M 個の queue へは同確率で移るものとする．

$OH1$ は以上述べたモデルによれば次のようにあらわされる[4),6)]．

$$OH1=\frac{\binom{N+M-1}{N}\left(\frac{\lambda}{\mu}\right)^N}{\sum_{l=0}^{N}\binom{l+M-1}{l}\left(\frac{\lambda}{\mu}\right)^l} \qquad (1)$$

ただし $\mu\equiv M\cdot\mu_0$

式(1)は従来から知られていたのであるが，従来のこのような解析ではタスクの数 N と主記憶装置の容量 C に関係なく一定であるとした．そして主記憶装置を使用していることをモデルのなかでどのように表現するかが問題であった．そこで λ を主記憶装置の容量とタスクの数の関数だとすることによって主記憶装置の容量を考慮することにした．λ の関数形をシミュレーションでだし，それにより解析してみた．中央処理装置の平均命令実行時間を g，1命令あたりの平均的な記憶装置への参照回数を α とし，主記憶装置の容量を C ページ，タスクの数を N としたとき，記憶装置への1回の参照に対して主記憶装置に参照されたアドレスを含むページがない確率を $E(C,N)$ とすると，

$$\lambda=\frac{\alpha}{g}(C,N) \qquad (3)$$

となる．

実際にはアーキテクチャーが変化した場合にはページ・サイズも変化するが，アーキテクチャーは一定であるとしておく，また極端にプログラム・サイズが小さくて1ページにおさまる場合はミッシング・ページ・フォールトは起こらないことになるが，このようなケースは除き，1ページよりも十分大きなプログラム，サイズを有する場合を対象とする．プログラム・サイズがどの程度より大きくなったらミッシング・ページ・フォールトの確率はプログラム・サイズに依存しなくなるかについてはこの報告の目的とはずれているために別に報告する予定である．

1つのタスクの使用可能なページは C/N となるから

$$E(C,N)=E\left(\frac{C}{N},1\right) \qquad (4)$$

と考えてよいだろう．

ここで $E(C/N,1)$ の関数形がどのようになるかを知る必要がある．主記憶装置の容量を x ページ，ミッシング・ページ・フォールトの確率を y とする．主記憶装置の容量が $x+dx$ になったときミッシング・ページ・フォールトが $y+dy$ となったとする (Fig. 4 参照)．そのときミッシング・ページ・フォールトの減少分は x のなかにページがなくて dx ページのなかにある確率になるから

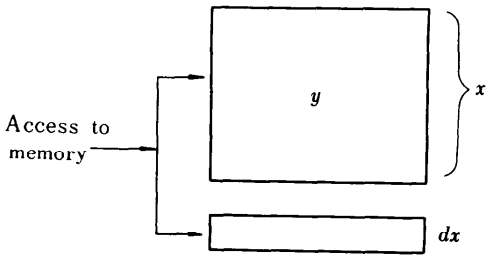


Fig. 4 Memory capacity and missing page fault

$dy=-(Bdx)$.

ただし，B は主記憶装置1単位に1ページが存在する確率である．B が定数と仮定すると上式を解けば

$$y=A\cdot e^{-Bx}$$

となり $E(x,1)$ は x の指数関数となる．

そこで HITAC 5020 のアドレス・パターン（バーチャル・メモリ方式計算数として使用した場合の）をトレースした磁気テープから x を変化させて $E(x,1)$ を（タスク数が1のときについて）求めてみた．

その結果が Fig. 5 である．Fig. 5 は横軸に主記憶装置のページ数，縦軸には $E(x,1)$ をとり縦軸のみを対数グラフで表現したものである．

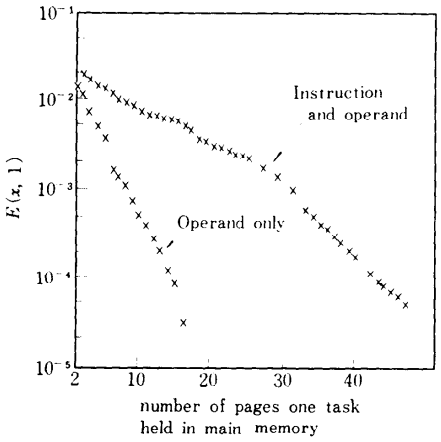


Fig. 5 Memory capacity (page) and swapping probability

対象としたアドレス・パターンは PL 1 W のコンパイルのフェーズ1であり，すべての記憶装置へのアクセスのアドレス・パターンに対して $E(x,1)$ を求めたものと，オペランドの主記憶装置へのアクセスのアドレス・パターンに対して $E(x,1)$ を求めたものの2つがプロットしてある．リプレースメント・アルゴリズムは least recently used[5] である．ただし，このときのページ・サイズ PS は256語である．このグラフから点はほぼ直線上にのっていることがわかる．

したがって $E(x,1)$ は次のような関数形をしているとしてよい．

$$E(x,1)=A\cdot e^{-Bx} \qquad (5)$$

式(4),(5)から

$$E(C,N)=A\cdot e^{-B\frac{C}{N}} \qquad (6)$$

となる．

式(3),(6)から

$$\lambda=\frac{\alpha}{g}\cdot A\cdot e^{-B\frac{C}{N}} \qquad (7)$$

となる．

1ページが補助記憶装置（1台）から主記憶装置へ転送される時間を T とすると μ_0 は T を用いて

$$\mu_0=\frac{1}{T} \qquad (8)$$

とあらわされる．

式(1)は Z を

$$Z\equiv\frac{\lambda}{\mu} \qquad (9)$$

と定義することによって

$$OH1=\frac{\binom{N+M-1}{N}Z^N}{\sum_{l=1}^{N}\binom{l+M-1}{l}Z^l} \qquad (10)$$

とあらわせる．

式(7),(8),(9)から

$$T=\frac{M\cdot g}{\alpha}\cdot\frac{Z}{A}\cdot e^{B\frac{C}{N}} \qquad (11)$$

となる．式 (11) から Z は C,T,N,M の関数であり，式 (10) からわかるように $OH1$ は Z,N,M の関数となっている．しかし逆に式 (10) を Z について解いて Z を $OH1$, N, M の関数として求め，これを式(11)に代入すると $OH1$ がある一定の値のときの C,T,N,M の関係がわかる．このことから，オーバーヘッドとタスクの数を一定にしたときはページを転送する時間 T は式(11)から主記憶装置のページ数 C の指数関数で与えられるもので良いことがわかる．

次に $OH2$ を求めるために T_{OS} を以下のように定義する．ミッシング・ページ・フォールトの処理プログラムとトラップ・ドラム・チャネルのプログラムをそれぞれ1回走るステップ数の和を T_{OS} とする．

タスクが queue 1 から中央処理装置に移ってからマッピング・フォールトが起こるまでの平均時間は $1/\lambda$，すなわち $\frac{g}{\alpha\cdot A}\cdot e^{B\frac{C}{N}}$ である．

したがって

$$OH2=\frac{g\cdot T_{OS}}{\frac{g}{\alpha\cdot A}\cdot e^{B\frac{C}{N}}+g\cdot T_{OS}}$$

$$=\frac{1}{\frac{1}{\alpha\cdot T_{OS}\cdot A}e^{B\frac{C}{N}}+1} \qquad (12)$$

となる．

式(12)を T_{os} に関して解けば

$$T_{os}=\frac{OH2}{\alpha\cdot A\cdot(1-OH2)}\cdot e^{B\frac{C}{N}} \qquad (13)$$

となる.

3. 結果とその検討

Fig. 6 は第2章の(1)式で λ をタスク数 N および主記憶装置のページ数 C に関係なく一定であるとし $M=1$ のときのタスク数 N とオーバーヘッド $OH1$ の関係を示したものである.

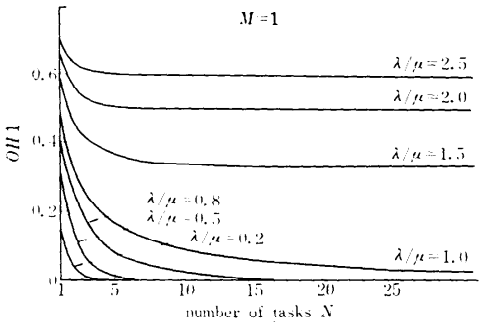


Fig. 6 Relation between the number of tasks and $OH1$ with λ assumed to be constant

これからわかることは, タスクの数が増大すると $OH1$ が減少しているがこれは λ をタスクの数 N に関係なく一定としたためで, 実際は N が増大すると経験上明らかなように $OH1$ が増大する. 前章の Fig. 5 の点を近似する直線を求めるために $y=A\cdot\exp(-Bx)$ の形をした関数でもっとも Fig. 5 に近い関数を多項式回帰で求めた. それによると, すべての主記憶装置へのアクセスのアドレス・パータンをトレースしたものに対する $E(x,1)$ の A,B はそれぞれ次のようになった.

$$A=0.0416 \qquad (14)$$

$$B=0.131 \qquad (15)$$

この数値を使用したタスクの数 N と主記憶装置のページ数 C による λ への影響を考慮したときのオーバーヘッドすなわち (10), (11) 式による $OH1$ を以下に示す.

Fig. 7 は横軸にタスクの数 N をとり, 縦軸に $OH1$ をとり $C=256$ で $\alpha/M\cdot T/g$ をパラメータとしたグラフである.

Fig. 8 は $OH1$ が 0.05 になったときのタスクの数の等高線をグラフにしたものである. すなわちこの

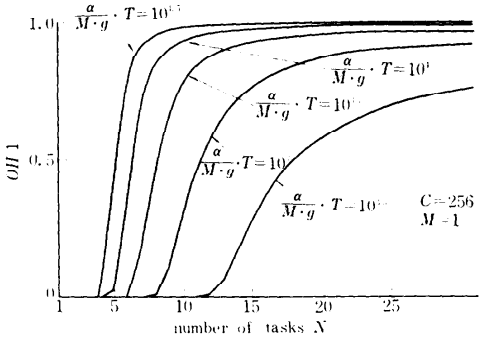


Fig. 7 Relation between the number of tasks and $OH1$

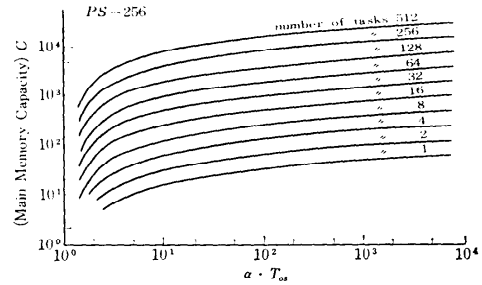


Fig. 8 Main memory capacity and relative secodndary memory speed at $OH1=0.05$

図は $OH1$ が悪化しないタスクの数の上限の数の等高線である. 横軸は $\alpha/M\cdot T/g$ すなわち1ページを補助記憶装置に転送する時間を中央処理装置のスピードで割ったものを示したものであり, 縦軸は C すなわち主記憶装置の容量をページを単位としてあらわしたものである. (1ページ=256 W)

ただし厳密にはモニターなどの常駐部をのぞいたものを縦軸にとってある. Fig. 9 は縦軸に $OH2$, 横軸にタスクの数をとったグラフで $C=256$ の場合である. $OH2$ もタスクの数が多くなると, あるところから急に増大することがわかる. Fig. 10 は縦軸に主記憶装置の容量 C をとり横軸に $\alpha\cdot T_{os}$ をとったものであり, そのときの $OH2$ が 0.05 となるときのタスクの数の等高線をあらわしたものである.

なお, 応答時間 R とオーバーヘッドの関係を簡単に示してみる. 1個のユーザーが端末で入力して端末へ応答の返ってくるまでに使用する中央処理装置の時間を T_R とする. N 個タスクがあり時分割で N 個のタスクを均等に実行しているとすれば $N\cdot T_R$ だけ中央処理装置の時間を消費した後で端末へ帰ってく

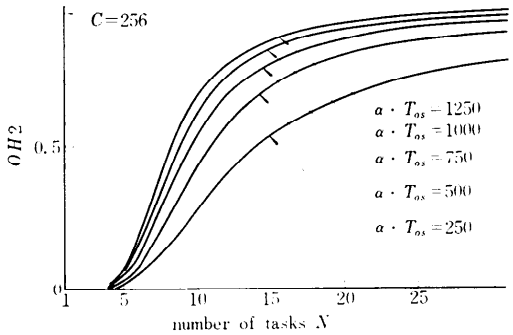


Fig. 9 Relation between the number of tasks and $OH2$

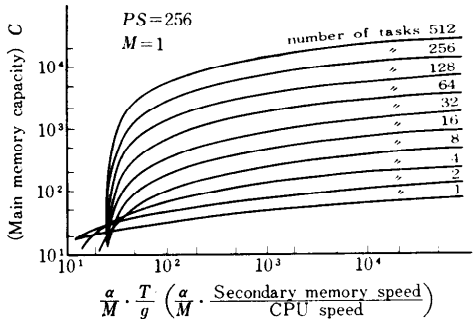


Fig. 10 Main memory capacity and T_{OS} at $OH2=0.05$

る．そしてオーバーヘッドの部分を考慮すると R は次のようにあらわされる．

$$R\fallingdotseq\frac{T_R\cdot N}{(1-OH1)(1-OH2)}. \tag{16}$$

式(16)から次のことがわかる．

Fig. 8 で示される最適タスク数と Fig. 10 から読みとったタスク数の小さい方よりもタスク数 N が小さいときには $OH1, OH2$ はほとんど 0 であるから応答時間はタスク数 N に比例するが，N がそれよりも大きいときには $OH1, OH2$ のどちらかが急激に 1 に近づいて応答時間 R は急激に増加する．

スループット TH は次のように表わされる．

$$TH\fallingdotseq\frac{(1-OH1)\cdot(1-OH2)}{T_R}. \tag{17}$$

具体例として 65 kW の主記憶装置をもちページ・サイズが 256 W であり，オペレーティング・システムが 25 kW 程度であり，1 命令実行時間が約 10 μs，補助記憶装置として 1 ページ転送するのに 20 ms 程度の磁気ドラムを持つタイム・ジェアリング・システムでは $C=160$, $\alpha=1$, $M=1$ として $\alpha/M\cdot T/g$ は約 2×10^3 となる．

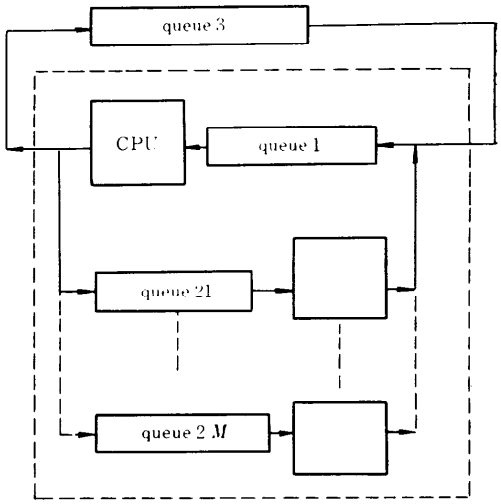


Fig. 11 A method to reduce overhead

Fig. 8 から $OH1$ を小さくするときの最大タスクは約 4 くらいとなる．また T_{OS} は約 1000 であるから[1] $OH2$ を小さくするためには Fig. 10 からタスク数を 4 ないし 3 以下にしなければいけない．したがって $OH1, OH2$ をともに小さくするためにはタスクの数を 3 以内にしなければいけない．

タスクの数が Fig. 8, Fig. 10 に示されるよりも大きくなると $OH1, OH2$ は急に悪化する．このようなことを避けるために 1 つの方法として Fig. 11 に示すように第 3 の queue 3 を設けて点線のなかには一定の数（Fig. 8, Fig. 10 で示される）よりも小さな数のタスクしか存在しないようにし，それ以外のタスクは queue 3 におとすようにする方法が考えられる．ここで queue 3 にあるタスクは強制的に補助記憶装置に待機させ，主記憶装置には割当てないようにしておく．点線のなかのタスクと queue 3 のタスクは適当な時間間隔ごとにいれかえるようにする．このようにして実効的なタスクの数を制限することによってオーバーヘッドを 0.05 以下に減らすことができる．

4. おわりに

バーチャル・メモリ方式の電子計算機をつかったタイム・シェアリング・システムのオーバーヘッドを 2 つに分け，中央処理製置のアイドル・タイムに関する $OH1$，オペレーティング・システムを走る時間に関するオーバーヘッドのうちの大部分を占めるミッシング・ページ・フォールトの処理のプログラムとトラッ

プ・ドラム・チャネルの処理のプログラム上を走るためのオーバーヘッド $OH2$ についてミッシング・ページ・フォールトの起こる確率を1つのタスクの占める主記憶装置の容量の指数関数で近似したこと，各タスクに均等に主記憶装置のページが分配されることを仮定したことによって解析した．それにより次のことがわかった．

多重走行をするタスクの数が増加すると1つのタスクに割当てられる主記憶が減少し補助記憶装置とのページの転送が増大しオーバーヘッドが増加する．

オーバーヘッドの大きなシステムにおいては CPU のスピードを速くしてもスループットはほとんど向上しない．このようなシステムでは補助記憶装置のスピードを速くするか主記憶の容量を増加するか，あるいはタスクの数を小さくすることによってオーバーヘッドを減少させ，その結果スループットを向上することができる．

付録 1

記号の説明

$F(t)$：主記憶装置に必要とするページがなくて補助記憶装置へページを要求するまでの時間の分布関数．

$B(t)$：1台の補助記憶装置がページを転送し終るまでの時間の分布関数．

λ：$F(t)$ が指数分布をしているとしたとき，その平均の逆数．すなわちミッシング・ページ・フォールトが起こる時間間隔の平均の逆数．

M：接続されている補助記憶装置の数．

λ_0：λ/M

μ_0：$B(t)$ が指数分布をするとしたときのその平均の逆数．すなわち1ページを補助記憶装置（1台）から主記憶装置へ転送する時間の逆数．

μ：$\mu_0 \cdot M$.

q：1命令を実行する時間．

α：1命令当り記憶装置を参照する回数．

C：主記憶装置のページ数．

$OH1$：オーバーヘッド1．すなわち中央処理装置のアイドル・タイムの全時間に対する比．(Fig. 1 参照)

T：1ページが主記憶装置から補助記憶装置（1台）へ転送される時間の平均すなわち $B(t)$ の期待値．

N：端末で入出力待ちのタスクを除いて中央処理装置で実行待ちまたは実行中のタスクの総数．いいかえると Fig. 3 内にあるタスクの総数．

R：応答時間．

T_R：1個のユーザーが応答するまでに使用する中央処理装置の時間．

PS：ページ・サイズ．

$P(n_1, n_2, \cdots, n_M)$：queue 21, queue 22, $\cdots$, queue 2M に $n_1, n_2, \cdots, n_M$ 個のタスクのある確率．

$P(n)$：queue 21, queue 22, $\cdots$, queue 2M に合計 n 個のタスクのある確率．

$E(C, N)$：主記憶装置の容量を C ページ，タスクの数を N としたときの記憶装置への1回の参照に対するその番地を含むページが主記憶装置にない確率．

T_{OS}：マッピング・フォールトとトラップ・ドラム・チャネルをそれぞれ1回起こしたときシステム・プログラム上を走るステップ数の和．

$OH2$：中央処理装置のアイドル・タイムを無視したときのオペレーティング・システムを走る時間の全時間に対する比 (Fig. 1).

TH：スループットすなわち単位時間に受けつけるタスクの数．

トラップ・ドラム・チャネル：1ページの情報をドラムから主記憶装置に転送したときにチャネルから起こす割込．

参考文献

1) 益田，他3：セグメンテーション機構を有するタイム・シェアリング・システムの解析：信学論，Vol. 54-C, No. 9, pp. 843～849 (昭 46-9).
2) P. H. Seaman and R. C. Roucy: Simulating Operating System: IBM Systems Journal, Vol. 8, No. 4, pp. 264～279, 1969.
3) 三上，他4：コンピューター・システム性能評価シミュレータ PACSS：情報処理，pp. 14～25，(昭和 46-1).
4) 田中：循環待ち行列モデルを用いた多重プログラミングシステムの解析：信学論，Vol. 53-C, No. 2, pp. 57～64 (昭 45-1).
5) 益田，他：ページング・マシンにおけるスワッピング・アルゴリズムの比較とプログラムの動作解析：情報処理，Vol. 13, No. 2, 1972, pp. 81～88.
6) 待ち行列研究会編，応用待ち行列事典（広川書店）.
7) 西本：ページング方式の計算機のメモリー・ヒェラルキー：昭和 47 年度電気関係学会東北支部連合大会予稿集，pp. 131.
8) H. Kobayashi: Some Recent Progress in Analytic Studies of System Performance: First USA-Japan Computer Conference Proceedings, Oct. 1972, pp. 130～138.

(昭和48年4月27日受付)
(昭和48年8月16日再受付)