ページングアルゴリズムの性能に関する実験的および理論的解析†

中所武司†† 林 利弘†††

ソフトウェアの信頼性向上のためには，プログラムの段階的詳細化法，データ抽象化技法などが有効である．しかし，これらをサポートする言語処理系は，ライブラリ化された他モジュール情報を必要とするため，コンパイル時に参照すべきデータが多くなる．

そこで，新たに開発された言語 SPL の処理系を対象として，これらのデータ参照時のページング動作解析を行い，処理効率の向上をはかった．

そして，実データの解析結果から，データ参照時のページング動作の特徴として，ページサイズが小さいほどフォールト率が低くなること，手続き部ほどではないが明らかな局所参照性があること，汎用ページングアルゴリズム間では，LRU, FINUFO, FIVE, FIFO の順に良い性能を示すことなどを明らかにした．

理論的解析では，ページ参照系列に局所参照性がある場合，LRU と FIFO の性能差はフォールト率の低い所ほど大きいこと，LRU と簡易 LRU 方式 (FINUFO と FIVE) の性能差は履歴情報の収集期間の長さとの関連が深く，LRU フラグのリセットのタイミングが重要であることなどを導いた．

1. まえがき

ソフトウェアの生産性および信頼性の向上はソフトウェア工学の主要課題であり，そのためのプログラミング法として，プログラムをモジュールの階層構造として表現するための研究がなされている．例えば，Dijkstra の階層構造設計[1]，Wirth 等の段階的詳細化法[2),3)]，Parnas のモジュール分割法[4)]，Myers の複合設計[5)]，Liskov のデータ抽象化法[6)] などである．

このようなモジュール化技法を実用化するためにはそれに適したプログラミング言語を開発することが望ましいが，その場合，処理系はライブラリ化された他モジュール情報を必要とするため，コンパイル時に参照すべきデータ（以後テーブルと称す）が多くなる．そこで，このテーブルを補助メモリに置き，必要に応じて主メモリにロードするようなページング処理が必要である．

ところが，ページング処理に関する研究[7)~13)]は，これまで仮想記憶システムの分野で主に行われてきたが，その成果をモジュール化プログラムの処理方式に適用するには2つの問題があった．すなわち，第1には，これまでの実データに基づくページング動作解析の多くはプログラムの手続き部を対象としており，データ参照に対する特性が明らかでないこと，第2には，実用的なページングアルゴリズム間の性能比較が主に実測データによって行われており，未だ十分には解明されていないことである．

本論文はこれらの問題に1つの解を与えることを目的とする．第1の問題については，新たに開発された，プログラムのモジュール化，階層化機能を有する言語 SPL[14)] (Software Production Language) の処理系[15)]が参照するテーブルのページング動作解析を行い，最適ページサイズ，局所参照性，各種ページングアルゴリズムの適用効果を調べることにより，データ参照時の特性を明らかにする．

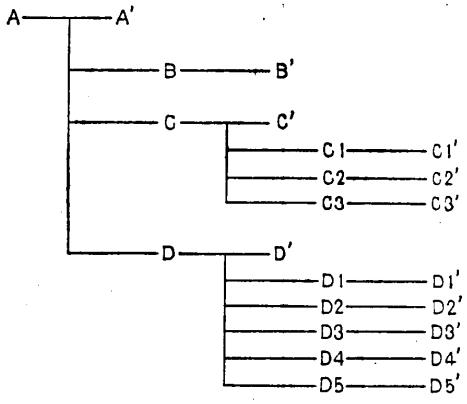
第2の問題については，各種ページングアルゴリズムの性能差に関する実験結果の一般性を示すために，LRU (Least Recently Used) 方式と FIFO (First In First Out) 方式の性能差の理論式の導出，簡易 LRU 方式間の性能差に対する理論的根拠の付与を行った．

2. モジュール化プログラムの処理方式

モジュラープログラミング言語としての SPL の特徴は，段階的詳細化法によってトップダウンに設計した各レベルのプログラムをそのままモジュールとして作成できることと，各モジュールを宣言部と手続き部に分離して作成し，宣言（環境）モジュールについてはその下位モジュールからの参照を許していることである．図1はこのモジュール化機能を用いて開発した

† Experimental and Theoretical Performance Analyses of Paging Algorithms by TAKESHI CHUSHO (Systems Development Laboratory, Hitachi, Ltd.) and TOSHIHIRO HAYASHI (Omika Works, Hitachi, Ltd.).

†† (株)日立製作所システム開発研究所
††† (株)日立製作所大みか工場



X: Environment module
X': Process module

図 1 階層構造のプログラムの例
Fig. 1 Example of hierarchically-modularized program.

SPL プログラムのモジュール構造例である.

このように階層構造で表現されたプログラムの各モジュールを分離コンパイルする場合は，設計時と同じように上位のモジュールからコンパイルを行い，解析処理後の中間結果をライブラリ化しておき，関連モジュールがそろった段階でオブジェクトの生成処理が行われる．そのために処理系で用いるテーブルが大きく，その大半は補助メモリに置かれる．そこで，このテーブルの参照時に生じる主—補助メモリ間のデータ転送回数の多少が処理性能に与える影響は大きい．次章でこのページング動作解析を行う.

3. ページング動作解析

3.1 解析方法

ページング動作解析は，まず図1の階層構造プログラム（約600ステップ）のコンパイル時のテーブル参照アドレス系列を補助メモリに収集した後，その解析プログラムを用いて行った．なお，これらのプログラムの開発過程は文献16) に詳しい.

参照アドレス系列は次のジョブ単位に収集した.

- 環境モジュール解析ステップ (ENV)
- 処理モジュール解析ステップ (PROC)
- オブジェクト生成ステップ (GEN)

これらの収集データに対する解析プログラムは以下の機能を備えた.

- 各ページ参照頻度の測定
- LRU スタックの各深さの参照頻度の測定
- 各種ページングアルゴリズムの適用と性能比較

ここで，評価対象とした5種類のページングアルゴリズムについて述べておく.

(1) LRU 方式[8),9)]

ページフォールト時には主メモリバッファ内の最も長い間参照されていないページを追い出す.

(2) FINUFO 方式 (First In Not Used First Out)

主メモリバッファ内の各ページに LRU フラグ1ビットを設け，参照されるとオンにする．フォールト時には，バッファを円環状につながったものとみなし，最後にロードされたページの次からサーチし，最初に出会ったフラグがオフのページを追い出す．この時，途中のフラグがオンのものはオフにしてスキップする．これは Multics[11)], OS 7[12)] などで採用されている.

(3) FIFO 方式

フォールト時には，バッファ内の最初にロードされたページを追い出す.

(4) FIVE 方式

LRU フラグとして5ビット設け，参照されると左端ビットをオンにする．フォールト時にはこのフラグを0～31の整数値と見て最小値のページを追い出すと共に，すべてのフラグを1ビット右にシフトする.

(5) 1部ページ常駐方式

例えば参照頻度の上位のページなど，1部のページを主メモリに常駐化してページング対象から除く.

3.2 ページサイズ

ページサイズが128，256，512バイトの場合について，LRU 方式によるフォールト率を測定すると，図2のようにページサイズは小さい方が良いという結果が得られた.

3.3 局所参照性

LRU 方式の場合，主メモリバッファ内にロードされているページの間でどのページがより最近に参照さ

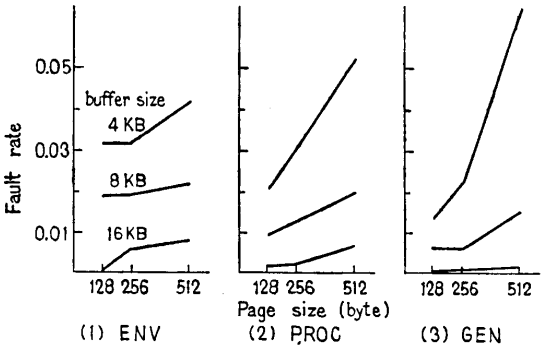


図 2 ページサイズとフォールト率の関係
Fig. 2 Relations between page size and fault rate.

表 1 LRUスタックおよび各ページの参照頻度分布
Table 1 (1) Reference frequency of each LRU stack distance (SD) and (2) reference frequency of each page, in first 30,000 data of PROC-job with 256 bytes/page.

(1) SD	Freq.	Rate (%)	(2) Order	Freq.	Rate (%)
1	18,164	60.5	1	4,903	16.3
2	5,360	17.9	2	2,271	7.6
3	1,267	4.2	3	2,148	7.2
4	1,263	4.2	4	2,055	6.9
5	1,034	3.4	5	1,616	5.4
6	447	1.5	6	1,527	5.1
7	347	1.2	7	1,036	3.5
8	236	0.8	8	783	2.6
9	212	0.7	9	682	2.3
10	127	0.4	10	572	1.9

れたかという情報を保存するためにスタックが用いられる．このスタックはより最近に参照されたページほど上位になるように処理される．

そこで，スタックの上位のページほど頻繁に参照される場合は LRU 方式が最適である．この性質は局所参照性と言われ，スタックの深さ i のページの参照確率を $L(i)$ とすると，

$$L(i)>L(j),\ i<j \qquad (1)$$

と表現される．

実測データは，**表1**(1)の例でスタックの先頭ページの参照が全体の 60% を越えるなど，明らかな局所参照性を示し，スタックの深さに従って参照頻度は急速に減少している．

3.4 静的参照頻度

1部のページをページングの対象から除き，主メモリに常駐させる方式が研究[13]されているが，常駐ページの選択が難しい問題である．本解析でも，各ページの参照頻度が表1(2)の例のように偏っていたので，参照頻度の上位のものを常駐にする方式を適用してみたが，特に良い結果は得られなかった．その理由は，表1の(1)と(2)の比較からわかるように，LRU スタック内の局所参照性が静的参照頻度の偏りよりもかなり大きいためと思われる．

3.5 ページングアルゴリズムの性能比較

テーブル参照アドレス系列の実測データに**3.1**で述べたアルゴリズムを適用してフォールト率を調べた結果，汎用アルゴリズム間では，LRU, FINUFO, FIVE, FIFO の順に良い性能を示した．その1部を**図3**に示す．

この結果から，LRU と比較した時の FIFO, FINUFO, FIVE のフォールト増加率を求めると**図4**になる．この図の縦軸は，図3の各測定点における

$$g(f_{LRU})=\frac{\text{比較対象方式のフォールト率}}{\text{LRU方式のフォールト率}}-1 \qquad (2)$$

を示す．これらの結果から次のことが確められる．

(1) FIFO は明らかに他方式より悪く，その度合はフォールト率の低い所ほど大きい．

(2) FINUFO は比較的安定した性能を示しており，大半は LRU に比べて 10% 以下の悪化にとどまっている．

(3) FIVE も LRU と比較して 10% 以下の悪化のものが多いが，フォールト率の低い所では FINUFO より劣る．

これらの性能差については次章で理論的に解析する．

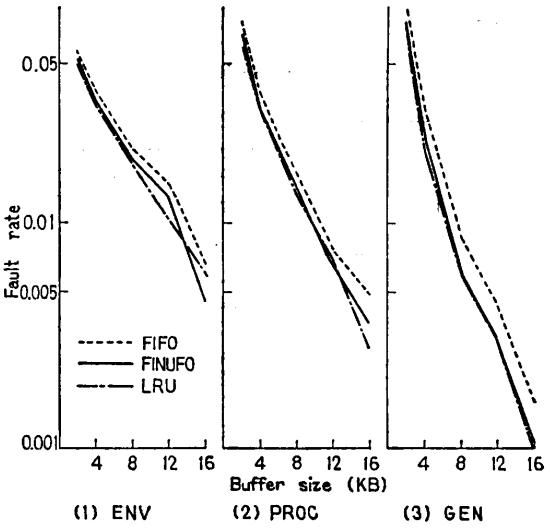


図 3 ページングアルゴリズムの性能
Fig. 3 Performance of paging argorithms (page size 256 bytes).

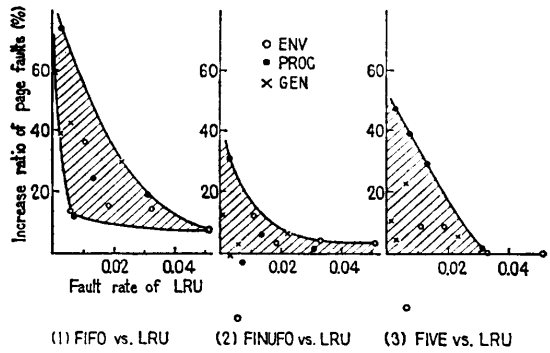


図 4 LRU 方式に比した他方式の性能低下度
Fig. 4 Increase ratio of page faults with FIFO, FINUFO and FIVE in comparison to LRU (page size 256 bytes).

4. 理論的解析

4.1 最適ページサイズ

3.2 の実測結果ではページサイズは小さい方が良いという結果が得られた．その理由は，ロケーション x のデータ参照直後のロケーション y のデータ参照確率を $p(y|x)$ とすると，コンパイラのテーブル参照では

$$p(x|x) \gg p(x \pm d_1|x) > p(x \pm d_2|x), \quad 0 < d_1 \ll d_2, \qquad (3)$$

すなわち，同一データの参照確率がその近傍の参照確率よりかなり高いためと考えられる．一方，手続き部の場合は近傍の参照確率が高いのでページサイズはある程度大きいものが良い．一般に仮想記憶システムでは 2～4 k バイトのものが多い．

このように，ページサイズを小さくするとフォールト率は減少するが，それに反比例してアドレス変換テーブルが大きくなるため，実際の最適ページサイズは次のように決まる．まず，参照データエリア（仮想記憶領域）の大きさ D，ページサイズ s の時，アドレス変換テーブルの大きさは，1 エントリ長を a として，$a(D/s)$ となる．これは使用可能な主メモリ容量 C の 1 部が用いられるため，バッファサイズを B とすると，

$$B + a(D/s) = C \qquad (4)$$

となる．一方，1 回のページ参照による主—補助メモリ間の入出力時間 t は，補助メモリの平均アクセス時間 u，転送速度 v の時，1 ページの入出力時間 $u + s/v$ とページ内容の書換え率 w とフォールト率 f から，

$$t(s, B) = (1+w)(u + s/v) f(s, B) \qquad (5)$$

で表わされる．したがって，最適ページサイズは(4)の制約条件下で(5)の t を最小にする s である．

本解析では，関数 $f(s, B)$ として図 2 のような実測値を用いて(5)を求めた結果，最適ページサイズは 256 バイトであることがわかった．

4.2 バッファサイズとフォールト率の関係

図 3 の実験結果からもわかるように，フォールト率はバッファサイズの増加と共に急速に減少している．これは **3.3** で述べたようにデータの局所参照性が強いためである．この実験結果を詳しく調べると，バッファサイズ b とフォールト率 f の関係は，

$$\begin{cases} f \geq f_0 \text{ のとき}, & f(b) = Ab^{-k} \qquad (6) \\ f < f_0 \text{ のとき}, & f(b) = Be^{-\alpha b} \qquad (7) \end{cases}$$

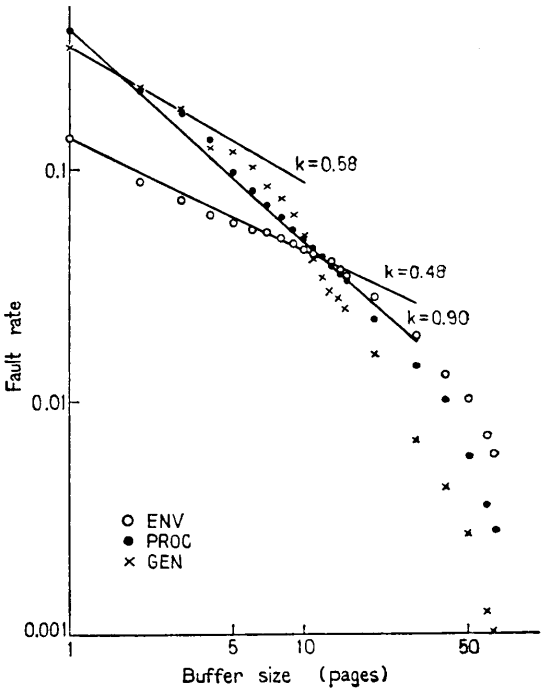


図 5 LRU 方式でのフォールト率とバッファサイズの関係
Fig. 5 Performance of LRU (256 bytes/page).

の形になっている．定数 (A, k, B, α, f_0) の値は，図 3，**図 5** などから

ENV : (0.14, 0.48, 0.058, 0.035, 0.04),
PROC : (0.40, 0.90, 0.068, 0.050, 0.03),
GEN : (0.34, 0.58, 0.065, 0.066, 0.18)　(8)

であった．

これまで Belady[7] らいくつかの研究で，一般に主メモリの少ない所では(6)が成立つという報告がなされているが，本実験でも同様であった．しかし，k の値は $k<1$ で，一般値 $(k \approx 2)$ より小さい．これは，データの局所参照性が手続きの場合ほどは強くないためである．

4.3 LRU と FIFO の性能差

ページング動作に関する理論的解析[8),9)] はいくつか行われているが，LRU と FIFO の性能差を解析したものは見当らない．その理由の 1 つは，(1)の局所参照性を有するデータに FIFO を適用した場合の扱いが難しいためである．一方，実システムによる実験例[10)]としては，FINUFO と FIFO のフォールト率の比が 1 : 2.2[11)] あるいは 1 : 1.3[12)] であったという報告などがある．

本節では，性能差の理論式を求める代りに，その上界式と下界式を導くことにより，両者の性能差が図 4

(1)のようにフォールト率の低い所で大きくなることを示す.

今,同じページ参照列に対してLRUおよびFIFOによるページング処理をしているとして,ある時点でのバッファ上のページ集合 G_{LRU}, G_{FIFO} の間に,

$$G_{FIFO}=G_{LRU}-\{u_1,\cdots,u_k\}+\{v_1,\cdots,v_k\} \quad (9)$$

の関係があるとする.ここでページフォールトが生じたとすると,追い出すページは,LRUではバッファページ数を b としてLRUスタックの b 番目のページであるが,FIFOでは最初にロードされたページでありそれを q とする.この時,LRUおよびFIFOのフォールト率 f_{LRU}, f_{FIFO} の差 d は,各々のバッファ内のページの参照確率の和の差に等しいので,ページ w の参照確率を $P(w)$ とすると,

$$\begin{aligned} d&=f_{FIFO}-f_{LRU}\\ &=\sum_{i=1}^{k}p(u_i)-\sum_{i=1}^{k}p(v_i)+p(q)-L(b+1)\end{aligned} \quad (10)$$

となる.$L(b+1)$ は,LRUで今追い出されたばかりの,スタックの $b+1$ 番目のページの参照確率である.q はLRUスタック内の位置とは無関係に選ばれることから,G_{FIFO} 内のページの平均参照確率を持つと考えられるので,

$$p(q)=\frac{1}{b}\left\{\sum_{i=2}^{b+1}L(i)-\sum_{i=1}^{k}p(u_i)+\sum_{i=1}^{k}p(v_i)\right\} \quad (11)$$

となる.一方,(1)より,

$$p(u_i)>p(v_j),\ 1\le i,\ j\le k \quad (12)$$

であるので,(10),(11),(12)から,d の下界式

$$d\ge\frac{1}{b}\sum_{i=2}^{b+1}L(i)-L(b+1) \quad (13)$$

が導かれる.

次に上界を求める.FIFOにおいて,あるページ x がロードされ,b 回のページフォールト後に追い出されるまでにこのページが参照される回数を n_x,全参照回数を n とすると,

$$f_{FIFO}=b/n, \quad (14)$$

$$p(x)=n_x/n, \quad (15)$$

となる.ここでページ x に注目すると,1回のページフォールトに対し n_x 回はフォールトが生じないため,

$$f_{FIFO}=1/(n_x+1) \quad (16)$$

と考えてよい.そこで,(14),(15),(16)から

$$f_{FIFO}=1-bp(x) \quad (17)$$

となる.これは,バッファ内のページの平均参照確率を $p(x)$ とした時のフォールト率を意味している.ここで $p(x)$ の下界を求めるために,ページ x がバッファ内に存在する間にLRUスタックのどこまで下がり得るかを考える.まず最悪の場合として,最初に x がロードされた時にバッファ内にあった他のページはすべて,追出される前に参照されたとすると,これらのページが x の上にくるので $(b-1)$ 段下がる.さらに $(b-1)$ 回のフォールトで $(b-1)$ 段下がる.結局,最大 $2(b-1)$ 段下がる.そこで,$P(x)$ はこの場合の平均参照確率以上と考えられるので,

$$P(x)\ge\frac{1}{2b-1}\sum_{i=1}^{2b-1}L(i) \quad (18)$$

となる.一方,f_{LRU} は次式で表わされる.

$$f_{LRU}=1-\sum_{i=1}^{b}L(i). \quad (19)$$

そこで,LRUとFIFOのフォールト率の差の上界式は,(17),(18),(19)から次式になる.

$$d\le\sum_{i=1}^{b}L(i)-\frac{b}{2b-1}\sum_{i=1}^{2b-1}L(i). \quad (20)$$

これらの上下界式を(2)に適用するとLRUとFIFOの性能差 g の上下界式は共にフォールト率の低い所で単調減少(付録参照)になることが導かれる.これは,フォールト率の低い所ほど相対的性能差が大きくなっている図4(1)の実験結果を裏づけている.

4.4 簡易LRU方式間の性能差

LRUはページ参照系列に局所参照性がある場合に適した方式であるが,スタック処理に時間を要するためあまり実用的でない.そこで実際にはFINUFOやFIVEなどの簡易方式が採用されている.これらの簡易方式は,ページ参照系列の履歴情報の保存量を限定することにより実行効率を良くしたものであるが,この限定は,履歴情報を収集する期間の短縮と情報量の削減という形で行われる.そこで本節では,これらの保存量の限定度合と簡易方式間の性能差に関する実験結果との関係について解析する.

まず,保存する情報量についてみると,LRUでは,主記憶バッファ内のすべてのページについて,どれがより最近に参照されたものかという順序が記憶されているが,FINUFOでは,ある一定期間に参照されたか否かの情報だけ記憶している.また,FIVEでは,最も最近に生じた5回のフォールトの各々の間で参照されたか否かの情報を記憶している.したがって,情報量はLRU, FIVE, FINUFOの順に多い.

次に保存する情報の収集期間 T を求める．T はページ参照回数で表現するのが直観的で良いが，ここでは比較し易さのためにその間に生じたフォールト数で表現する．T はフォールト率の関数になるので，参照回数の代りにそれにフォールト率を乗じたフォールト数を用いても一般性は失われない．

まず LRU の場合，ある時点での LRU スタックの状態は，主記憶バッファページ数を b として，b 回前のフォールト以後のページ参照系列によって決まる．なお，このことは，それ以前に参照されてその後は参照されていないページは既にバッファ内に存在しないこと，および最も最近の b 回のページ参照がすべてフォールトの場合，b 回前のフォールトでロードされたページは深さ b のスタックの底に記憶されていることなどから証明できるが詳細は略す．したがって，

$$T_{LRU}=b \tag{21}$$

となる．

次に，FINUFO の参照フラグは，**3.1**（2）項で述べた周期的なスキャンの間に参照されたものだけ記憶している．そして，その周期の間に生じるフォールト数はバッファ内のページでフラグがオフのものの数に等しいので，そのオフのフラグの割合を r とすると，

$$T_{FINUFO}=rb \tag{22}$$

となる．r は，定常状態ではオンからオフに変わるフラグとオフからオンに変わるフラグの数が等しいことから，次式を解いて求める．

$$\frac{b(1-r)}{br}f \leq r(1-f). \tag{23}$$

左辺は，フォールト時のスキャンでオフのフラグに出会うまでにオンからオフに変えられるフラグ数にフォールト率を乗じたもの，右辺は，参照されてオンにするフラグがその前にオフである確率 δ を r で近似し，それにフォールトの生じない確率を乗じたものである．なお，δ は実際には局所参照性のために r より小さいため不等式になる．結局，(21)，(22)，(23)から，

$$T_{FINUFO} \geq T_{LRU}\frac{\sqrt{f(4-3f)}-f}{2(1-f)}. \tag{24}$$

最後に FIVE の場合は，**3.1**（4）項で述べたように最も最近に生じた5回のフォールトの各々の間で参照されたか否かの情報を記憶しているので，

$$T_{FIVE}=5 \tag{25}$$

となる．

以上で3方式の履歴情報収集期間の一般式が導かれたので，相互比較を行う．T_{LRU} と T_{FINUFO} はデータの局所参照性に依存するので，今回の実験式(6)，(7)に(8)の ENV のパラメータを適用すると，**図6**が得られる．なお，FINUFO の場合は不等式なので，同じ ENV の場合の実測値も求めておいた．

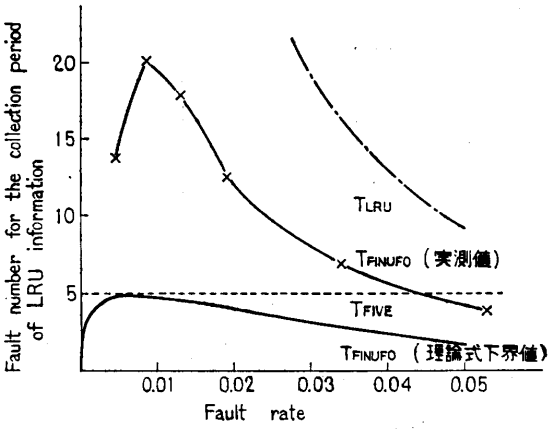


図 6　各アルゴリズムの履歴情報収集期間
Fig. 6 The collection period of LRU information on LRU, FINUFO and FIVE.

この図において，LRU と他の簡易方式の履歴情報収集期間の差はフォールト率の小さいほど大きく，図4(2)，(3)の実験結果と合致している．一方，FINUFO と FIVE を比べると，LRU フラグとして前者が1ビット，後者が5ビット用いているにもかかわらず，フォールト率の小さい所で履歴情報収集期間は前者が上回る傾向にあり，図4の実験結果と合っている．これは，簡易 LRU 方式の性能は，単に履歴情報保持のための記憶容量だけでなく，LRU フラグのリセットのタイミングに大きく影響されることを示している．

5. むすび

新たに開発された，プログラムのモジュール化，階層化機能を有する言語 SPL の処理系を対象として，テーブル参照アドレス系列のページング動作解析を行い，その結果の一般性を示すために，ページングアルゴリズムの性能について理論解析を行った．

実験結果から，テーブル参照時のページング動作の特徴として，ページサイズが小さいほどフォールト率が低くなること，手続き部を対象とした場合ほどではないが，明らかな局所参照性があること，汎用ページングアルゴリズム間では，LRU，FINUFO，FIVE，FIFO の順に良い性能を示すことなどを明らかにし

た.

理論的解析では，ページ参照系列に局所参照性がある場合，LRUとFIFOの性能差はフォールト率の低い所ほど大きいこと，LRUと簡易LRU方式(FINUFOとFIVE)の性能差は履歴情報の収集期間に対応しており，LRUフラグのリセットのタイミングが重要であることなどを導いた.

手続き部を対象としたページング動作解析はいくつか行われているが，データを対象にしたものは少ない．今回はコンパイラのテーブル参照を採上げたが，今後，データベース等に対象を広げて，データ参照時の特性を明らかにしていく必要がある.

最後に本研究の機会を与えていただいた日立製作所大みか工場高井兵庫副技師長ならびに日頃御指導頂く同社システム開発研究所中田育男博士に深謝します.

参考文献

1) Dijkstra, E. W.: The Structure of the "THE"—Multiprogramming System, Comm. ACM, Vol. 11, No. 5, pp. 341-346 (1968).
2) Dijkstra, E. W.: Notes on Structured Programming, Structured Programming, Academic Press, pp. 1-82 (1972).
3) Wirth, N.: Program Development by Stepwise Refinement, Comm. ACM, Vol. 14, No. 4, pp. 221-227 (1971).
4) Parnas, D. L.: On the Criteria to be Used in Decomposing Systems into Modules, Comm. ACM, Vol. 15, No. 12, pp. 1053-1058 (1972).
5) Myers G. J.: Reliable Software Through Composite Design, p. 159, Petrocelli/Charter, New York (1975).
6) Liskov, B. and Snyder, A.: Abstraction Mechanism in CLU, Comm. ACM, Vol. 20, No. 8, pp. 564-576 (1977).
7) Belady, L. A. and Kuehner, C. J.: Dynamic Space-Sharing in Computer Systems, Comm. ACM, Vol. 12, No. 5, pp. 282-288 (1969).
8) Mattson, R. L. et al.: Evaluation Techniques for Storage Hierarchies, IBM Sys. J., Vol. 9, No. 2, pp. 78-117 (1970).
9) Coffman, E. G. and Denning, P. J.: Operating System Theory, p. 331, Prentice-Hall, Englewood Cliffs (1973).
10) 益田 他2：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, pp. 81-88 (1972).
11) 池田：コンピュータユーティリティの構造，p. 345，昭晃堂，東京 (1974).
12) 諸方 他2：OS7メモリスケジューリング方式，情報処理学会第14回大会予稿集，pp. 363-364 (1973).
13) Yoshizawa, Y. and Kondo, M.: Effect of Partially Preloaded Program on Paged Virtual Memory, Journal of Information Processing, Vol. 1, No. 2, pp. 92-97 (1978).
14) 林 他3：制御用トップダウン・ストラクチャードプログラミング言語—SPL—，日立評論，Vol. 60, No. 3, pp. 59-64 (1978).
15) 中所 他4：制御用ストラクチャドプログラミング言語 SPL の処理系の特徴，情報処理学会第17回大会予稿集，pp. 5-6 (1976).
16) Chusho, T. and Hayashi, T.: Two-Stage Programming: Interactive Optimization after Structured Programming, Proc. the 3rd UJCC, pp. 171-175 (1978).

付　録

ここでは**4.3**で述べたLRUとFIFOの性能差関数gの上，下界関数g_U, g_Lがf_{LRU}（以下fと略す）の小さい所で単調減少であることを示す．まず簡単のため，

$$L(i)=\int_{i-1}^{i} l(x)dx,\quad i=1,2,\cdots \tag{26}$$

なる関数$l(x)$を導入する．$l(x)$は(1)の$L(i)$の性質から，$0\leq u<v$の範囲で次の性質を有する.

$$0<l(v)<\frac{1}{v-u}\int_u^v l(x)dx<l(u)\leq 1. \tag{27}$$

そこで，(2)，(19)，(20)などからg_Uとその導関数は，

$$g_U(f)=\frac{\int_0^b l(x)dx-\frac{b}{2b-1}\int_0^{2b-1} l(x)dx}{1-\int_0^b l(x)dx}, \tag{28}$$

$$\left(1-\int_0^b l(x)dx\right)^2\frac{dg_U}{db}$$
$$=l(b)-\frac{1-\int_0^b l(x)dx}{2b-1}\left\{bl(2b-1)-\frac{\int_0^{2b-1} l(x)dx}{2b-1}\right\}-\frac{b}{2b-1}l(b)\int_0^{2b-1} l(x)dx, \tag{29}$$

となる．ここで(27)から，

(29)式右辺
$$>l(b)-\frac{bl(2b-1)-l(2b-1)}{2b-1}-\frac{bl(b)}{2b-1}$$
$$=\frac{b-1}{2b-1}\{l(b)-l(2b-1)\}>0. \tag{30}$$

一方，(19)から$db/df<0$なので，結局，

$$dg_U/df=(dg_U/db)(db/df)<0, \tag{31}$$

となり，g_U は単調減少関数であることが示された．

次に，(2)，(13)，(19) などから g_L とその導関数は，

$$g_L(f)=\frac{\frac{1}{b}\int_1^{b+1} l(x)dx-L(b+1)}{1-\int_0^b l(x)dx}, \tag{32}$$

$$\begin{aligned}&\left(1-\int_0^b l(x)dx\right)^2\frac{dg_L}{db}\\&=\left\{\frac{1}{b}\int_1^{b+1} l(x)dx-l(b+1)\right\}\\&\quad\times\left\{l(b)-\frac{1}{b}\left(1-\int_0^b l(x)dx\right)\right\}\\&\quad+\{l(b)-l(b+1)\}\left\{1-\int_0^b l(x)dx-l(b)\right\}\\&\quad+l(b)\{l(b)-L(b+1)\}\end{aligned} \tag{33}$$

となる．(27) から (33) の右辺の第1項第1因子，第2項第1因子および第3項は正なので，(33) の右辺が正であるためには，第1項第2因子と第2項第2因子が正であれば十分である．すなわち，

$$1-\int_0^b l(x)dx>l(b)>\frac{1}{b}\left(1-\int_0^b l(x)dx\right). \tag{34}$$

この条件は，本実験の場合は，b がある程度大きい，$b>10\sim30$ のところ，すなわち，f が小さい，$f<0.09\sim0.01$ のところで成立する．この場合は

$$dg_L/df=(dg_L/db)(db/df)<0, \tag{35}$$

となるので，結局，g_L は f の小さい所で単調減少関数となる．

(昭和53年8月25日受付)
(昭和54年6月21日採録)