数理解析研究所講究録
992巻 1997年 229-235

オンライン先読みページングゲームにおける最適戦略の設計と解析

九州大学大学院システム情報科学研究科　山家 明男† (Akio Yanbe)
九州大学大学院システム情報科学研究科　櫻井 幸一† (Kouichi Sakurai)

概要: 仮想記憶システムのモデルとしてよく知られているページング問題において，ページングアルゴリズムが要求を先読みできるモデルを扱う．この先読みページング問題を先読みページングアルゴリズムと要求入力側との二人ゲームとして考え，ゲームの最適値および最適なアルゴリズムを求める．

1 はじめに

ページング問題とは，k ページの高速メモリと $n(> k)$ ページの低速メインメモリを持つ仮想記憶システムをモデル化したものである．このシステムでは要求ページの列を供給しなければならないが，要求ページが高速メモリに入っていることで供給が満たされることになる．高速メモリに入ってないページが要求されることをページフォールトといい，これが起こったら高速メモリにあるページを一つはずしてこの要求ページを高速メモリにロードしなければならない．ページフォールトが起こった時に，どのページを高速メモリから取り除くか決定するアルゴリズムのことをページングアルゴリズムという．ページングアルゴリズムの効率の良さはページフォールトの起こった回数と関連付けて測られる．

このページング問題に対するアルゴリズムを考える設定としてオンラインとオフラインというものがある．オンラインアルゴリズムは未来の要求に関する情報を使わないという設定で設計されるが，オフラインアルゴリズムは未来の要求をすべて使ってもよいという違いがある．

Belady [Bel66] はページング問題に対する最適オフラインアルゴリズムを与えている．このアルゴリズムはOPT または MIN と呼ばれ，ページフォールトが起こった時に将来最も長い間要求されないページを入れ換えるというもので，ページフォールト数が最も少なくてすむアルゴリズムである．しかし，オフラインアルゴリズムは実際のシステム上では実現不可能であるため，ページング問題のアルゴリズムはオンライン設定で考えるのが普通である．

ページング問題では，どんなオンラインアルゴリズムを使っても要求列のすべての要求にページフォールトを起こすような要求列が存在している [Bel66, MGST70]．これによって，最悪時のページフォールト数によってオンラインアルゴリズムの性能を測っても差がでないことがわかる．このことから，オンラインページング問題では絶対的なコストの大きさでアルゴリズムの性能を測るのは意味がないということで，他のアルゴリズムとのコストの比による解析が行われるようになった．

オンラインページング問題の競合比的解析では，オンラインアルゴリズムの効率の良さはオフライン最適アルゴリズムとの性能比 (competitive ratio) によって算出される．オンラインアルゴリズム A の競合比が h であるとは，全ての要求列 ρ に対して，$C_A(\rho) \leq h \cdot C_{opt}(\rho) + a$ が成り立つことである．ここで $C_{opt}(\rho)$ は最適オフラインアルゴリズムで要求列 ρ を処理した時のページフォールト数，$C_A(\rho)$ は A で ρ を処理した時のページフォールト数，a は初期状態に依存する定数である．

Sleator と Tarjan [ST85] は，それまでに知られていたオンラインアルゴリズムの中で LRU(Least Recently Used) と FIFO(First-In, First-Out) は競合比が k になることを示した．さらに，どんなオンラインアルゴリズムも競合比を k より小さくすることはできないことを示した．

†〒812-81 福岡市東区箱崎 6-10-1 九州大学大学院システム情報科学研究科情報工学専攻 TEL.092-641-4050 FAX.092-632-5204
{yanbe, sakurai}@csce.kyushu-u.ac.jp

オンラインとオフラインという設定は未来の情報に関してまったくわからないかすべて知っているかという両極端である．そこで，ある有限の長さだけ (l とする) 将来の情報を知ることができるという中間的な設定，つまり先読みの設定を考えることはごく自然な流れである．我々が本論文で考える先読みモデルは，先読み数が l の場合，長さ $l+1$ の先読みキューを用意し，その中に現在の要求 1 つとその後に入ってくる l 個の要求を入れておく．そして，先読みアルゴリズムはこの $l+1$ 個の要求列を見て処理を行うことになる．この最も単純なモデルは弱先読みモデルと呼ばれている．直観的に，先読みアルゴリズムは未来の情報をいくらかでも手に入れられるので，オンラインアルゴリズムよりも性能が良くなると思われるが，残念ながら，競合比的解析ではどんな先読みアルゴリズムも競合比は k より小さくならない (改善されない) ことが示されている [BB94, KP94]．つまり，先読み設定用に特別にアルゴリズムを考えなくても LRU,FIFO を使えば競合比は最適となるわけである．

このことから，競合比的解析ではオンラインアルゴリズムと先読みアルゴリズムに差がでないことがわかる．その後，先読みページング問題に対して他の先読みモデルを考えて競合比的解析がいくつか行われている．

しかし，ここで要求列の長さに対する最悪時のページフォールト数の割り合い (最悪ページフォールト率) を使ってアルゴリズムの性能を評価するようなゲーム的解析を行えば，オンラインアルゴリズムと先読みアルゴリズムとに差が出てくるのではないかと考えた．我々のゲーム的解析の結果としては，まず，先読み数 $l < n-k$ の場合にはオンラインアルゴリズムと差が出ないことがわかった．つまり，要求列のすべての要求に対してページフォールトが起こるような要求列が存在していることを示した．

次に，我々はオフラインアルゴリズムを参考に先読みアルゴリズム A_l を提案し，これを使うと，先読み数 $l \geq n-k$ の場合には最悪ページフォールト率は必ず改善されることを示した．つまり，長さ N の要求列に対してフォールト数は必ず N より少ない回数ですむ．この場合，LRU,FIFO は先読み分の要求を使わないので先読み数 $l \geq n-k$ の先読み設定ではゲーム的最適ではない．また，さらに先読み数 $l = n-k$ の場合には，A_l はゲーム的に最適戦略であり，A_l を使うと最悪の場合のページフォールト率は $l/(l+1)$ となることを示した．$l > n-k$ の場合については，$k = 1, 2$ の場合の最悪ページフォールト率を求めることができた．

なお，先読みモデルには他にいくつか考えられており，先読みキューの中に読み込まれているデータによって以下のような種類がある．なお，これらには競合比的解析しか行われていないので，今後ゲーム的解析を行う余地があると思われる．

Young [You91] の資源限定先読みモデルでは，先読みキューの中には現在の要求と，将来 l 回ページフォールトを起こすような最長の未来の要求列が読み込まれる．このモデルに対しては，競合比 $\max\{2\frac{k}{l+1}, 2\}$ の先読みアルゴリズムが示された．しかし，この先読みの方法はキューの長さが固定ではないので現実的ではない．

Albers [Alb93] の強先読みモデルでは，先読みキューの長さは $l+1$ で，現在の要求も含めて $l+1$ 個の異なる要求が読み込まれている．このモデルに対しては，先読み数 $0 \leq l \leq k-2$ において競合比 $k-l$ の先読みアルゴリズムを示し，さらに $k-l$ より小さくできないことも示している．

Breslauer [Bre96] の自然先読みモデルでは，長さ $l+1$ の先読みキューの中には，現在高速メモリの中に入ってないページで異なる要求が最大 $l+1$ 個読み込まれている．このモデルに対しては，競合比 $\frac{k+l}{l+1}$ の先読みアルゴリズムを示し，これが最適競合であることも示している．

2 定義

2.1 先読みページング問題

先読みページング問題は(n,k,l)で与えられる．ここで，nは低速メインメモリの全ページ数，kは高速メモリに入れられるページ数，lは先読み数である．このページングのシステムに対して，要求列$\rho = r_1,r_2,r_3,\cdots,r_N$が出される．各$r_i$は全$n$ページのうちの1つのページを表す．もし要求ページが高速メモリ内にあれば，ページングアルゴリズムはなにもしなくてよい．つまりこの要求に対してはコストはかからない．しかし，もし要求ページが高速メモリに入っていなければ(これをページフォールトという)，ページングアルゴリズムはこのページをメインメモリから高速メモリにロードすると同時に高速メモリにあるk個のページのうち一つを取り除かなければならず，このときにコストが1かかる．つまり，ページングアルゴリズムは各時点でどのkページを高速メモリに入れておくかということを決定するためのアルゴリズムである．なお，本論文では決定性のものだけを考える．要求ページr_iを処理した後の高速メモリの状態を，高速メモリ内にあるページの集合S_iで表すことにする．初期状態S_0では高速メモリには任意のkページが入っているものとする．

2.2 先読みモデル

本論文で考える先読みモデルは，先読み数がlの場合，長さ$l+1$の先読みキューを用意し，その中に現在の要求1つとその後に入ってくるl個の要求を入れておく．先読みアルゴリズムではページフォールトが起こってどのページを高速メモリからはずすか決める際に，このキュー内にある$l+1$個のページを見ることができる．つまり，l先読みアルゴリズムをA_lとすると，i番目の要求ページr_iを処理するときにページフォールトが起こった場合，高速メモリから取り出すページxは

$$A_l(r_i,r_{i+1},\cdots,r_{i+l}) \to x \in S_{i-1}$$

のように決められる．

2.3 最悪ページフォールト率

l先読みページングアルゴリズムA_lの要求列ρに対するページフォールト数を$PF_{A_l}(\rho)$で表す．要求列の長さをNとし，A_lを使った時の最悪時のページフォールト率を次のように定義する．

$$\max_{\rho} \lim_{N\to\infty} \frac{PF_{A_l}(\rho)}{N}$$

本論文では特に，要求列を生成するアルゴリズムを考え，システム側と要求側のゲームとしてこの問題を考えるので，要求列生成アルゴリズムをΦとし，Φの生成する要求列を$\rho(\Phi)$と書いて，Φによる最悪時のページフォールト率を

$$\min_{A_l} \lim_{N\to\infty} \frac{PF_{A_l}(\rho(\Phi))}{N}$$

と定義すると，いわゆるゲームの値は

$$\max_{\rho(\Phi)} \min_{A_l} \lim_{N\to\infty} \frac{PF_{A_l}(\rho(\Phi))}{N}$$

と定められる．これを先読みページング問題の最悪ページフォールト率と呼び，システム側(A_l)はこの値を小さくし，要求側(Φ)は大きくすることが目的となる．この値が決定性のA_l,Φを使って決まることは文献[YS96]で示している．(つまり，ゲーム理論の言葉でいえば，このゲームには純粋最適戦略が存在するということ．)

3 先読みページング問題の最適戦略

3.1 先読み数 $l < n-k$ の場合の最悪ページフォールト率

先読みを行わないページング問題では，最悪時のページフォールト数に関して以下のような定理が示されている．これで，先読みを行わない場合はどんなオンラインアルゴリズムも最悪ページフォールト率は1であることがわかる．

定理 3.1 ([Bel66, MGST70]) ページング問題では，どんなオンラインアルゴリズムに対しても長さ N の要求列で N 回ページフォールトが起こるようなものが存在している．

直観的には，先読みができれば未来の情報をいくらかでも手に入れられるので，それだけページフォールト数を減らすことができると思われるが，先読み数が少ない場合には最悪の場合のページフォールト数は先読みしない場合と変わらないことを示した．

定理 3.2 先読みページング問題で先読み数 $l < n-k$ の場合，どんな l 先読みアルゴリズムに対しても，長さ N で N 回ページフォールトが起こるような要求列が存在する．

証明: 次のような入力列を生成するアルゴリズムを考える．まず，最初の要求 $r_1 r_2 \cdots r_{l+1}$ をすべて異なるページで，かつ初期状態 S_0 に入っていないページで構成する．これで，キューには $r_1 r_2 \cdots r_{l+1}$ が入る．この後，ページングアルゴリズムが要求をひとつ処理するたびに，次のように新しく要求を入力することにする．まず，次の要求ページ r_1 に対してページフォールトが起こるので，この時に高速メモリから取り除いたページを x_1 とする．これを次の要求ページ r_{l+2} とする．一般的に書くと，要求ページ r_i を高速メモリに入れるために高速メモリから取り除いたページを x_i とする．これを次の要求ページ r_{l+i+1} とする．

このアルゴリズムによって生成される要求列は，ページングアルゴリズムが見ることのできる未来の $l+1$ 個の要求のうち，どの要求ページも高速メモリに入っていないという状況を作り上げることができ，よってページフォールトが常に起こることになる． □

つぎに，先読み数 $l \geq n-k$ では先読みアルゴリズムをうまく作ることによって，オンラインアルゴリズムと比べて最悪ページフォールト率が改善されることを示した．つまり，最悪ページフォールト率は1より小さくできるということである．

定理 3.3 先読みページング問題で先読み数 $l \geq n-k$ の場合，l 先読みアルゴリズムを使うと，長さ N のどんな要求列に対してもページフォールト数を N 回未満にすることができる．

証明: これは，先読みアルゴリズムがキューにある $l+1$ 個のページを見て，キュー内のページに高速メモリに入っているページがあれば，そのページが要求されるまで高速メモリから出さなければいいし，キュー内のページで高速メモリに入っているページがなければ，$l+1 \geq n-k+1$ より同じページが二回以上現れていることになるので，一回目に現れた時に高速メモリにそのページを読み込んでから二回目に現れるまで，高速メモリから出さなければよい．こうすれば，要求列のどの要求にもページフォールトが起こるということはあり得ない． □

そこで，$l \geq n-k$ の場合，最悪ページフォールト率がどのくらい小さくなるのか解析を行っていくことにする．まず，要求列生成アルゴリズム Φ を考え，Φ がおこす最悪ページフォールト率の下限を調べる．

- 要求列生成アルゴリズム Φ

  最初にキューの中に入る要求列 $r_1 \cdots r_{l+1}$ は，高速メモリに入ってない (S_0 にない) ページをできるだけ多く含むように作る．そして要求ページ r_i の処理後，キューには $r_{i+1}, \cdots, r_{i+l}$ がこの順で入って

いる．ここで，次の要求ページ r_{i+l+1} は次のように決める．もし，高速メモリに入ってなくて，かつキューにも入ってないページがあれば，(つまり，S_i に入ってなくて，かつ $\{r_{i+1},\cdots,r_{i+l}\}$ に入ってないページがあれば，) そのページを r_{i+l+1} とする．そんなページがなければ，r_i を r_{i+l+1} とする．

この Φ の最悪ページフォールト率の下限について以下の補題を示す．

補題 3.4 Φ の生成する要求列の最悪ページフォールト率の下限は少なくとも $\frac{n-k}{l+1}$ である．

証明: 先読み数 $l \geq n-k$ の場合，Φ はキューの中に常に高速メモリに入ってない $n-k$ 個の異なるページを入れておくので，要求を $l+1$ 個処理する間に少なくとも $n-k$ 回はページフォールトを起こす． □ ここで述べた Φ の設計および下限の解析はまだ甘く，改善の余地は残されていると思われる．

今度は逆に，先読みページングアルゴリズム A_l を考える．最適オフラインページングアルゴリズム (OPT) の動きは，ページフォールトが起こった時に将来最も長い間要求の来ないページを高速メモリから除くというものである [Bel66]．よって，これを参考にして自然な形で以下のような先読みページングアルゴリズムを考える．

- ページフォールトが起こった時に，高速メモリ内のページのうちキューに入っていないページを取り除く．このようなページが存在しない場合，キューにある頂点のうち一番長い間要求されないページを取り除く．

この A_l は定理 3.3の証明で書いた先読みアルゴリズムの動きを含んでいるので，必ず最悪ページフォールト率は 1 よりも小さくなる．また，先読み数 l が要求列の長さ N より大きい時には，最適オフラインページングアルゴリズム OPT と全く同じ動きをする．

この A_l の最悪ページフォールト率について，以下の補題を示す．

補題 3.5 先読み数 $l = n-k$ の場合に A_l が起こす最悪ページフォールト率の上限は $\frac{l}{l+1}$ である．

証明: これは要求列から連続した $l+1$ 個 $(r_i, r_{i+1}, \cdots, r_{i+l})$ をどんなに取り出してきても，A_l が $l+1$ 回連続してページフォールトを起こすことはあり得ないことを示す．まず，キューに $r_i, r_{i+1}, \cdots, r_{i+l}$ が並んでいて，A_l が r_i の処理を行おうとしている時，高速メモリに入っているページがこのキューの中に入っている場合と入っていない場合の二つの場合分けを行う．

高速メモリに入っているページがキューの中に入っている場合，そのページのうち最初に現れるページを r_{i+j} とする．A_l を使うと，r_{i+j} が処理されるまでページ r_{i+j} は高速メモリから取り除かれることはないので，r_{i+j} を処理する時にページフォールトは起こらない．よって，この場合 $l+1$ 回連続でページフォールトが起こることはない．

高速メモリに入っているページがキューの中に入っていない場合，$l \geq n-k$ より，キューの中に同じページが少なくとも二回現れることになる．このうち一度現れてからもう一度現れるまでの間が一番短いものを r_{i+j} と r_{i+j+h} とする．すると，r_{i+j} が高速メモリに読み込まれてから次に r_{i+j+h} を処理するまで，このページは高速メモリから取り除かれることはない．よって，この場合も $l+1$ 回連続でページフォールトが起こることはない．

ゆえに，要求列から連続した $l+1$ 個をどんなに取り出しても，そのすべてにページフォールトを起こすことはないので，最悪ページフォールト率は $\frac{l}{l+1}$ 以下となる． □

よって，上記の補題 3.4，3.5より先読み数 $l = n-k$ の場合の先読みページング問題の最悪ページフォールト率について，以下の定理がいえる．

定理 3.6 A_l と Φ は $l = n-k$ のときのゲーム理論的に最適なアルゴリズムであり，最悪ページフォールト率は $\frac{l}{l+1}$ である．

3.2 $k = 1, 2$ の場合の最悪ページフォールト率

k の値によって先読みページング問題の最悪ページフォールト率を考えてみると，以下の結果が分かる．

定理 3.7 $k = 1$ の場合には，任意の先読み数の先読みアルゴリズムに対して，長さ N で N 回ページフォールトが起こるような要求列が存在する．

つまり，この場合の最悪ページフォールト率は 1 であるということで，これは明らかである．

定理 3.8 $k = 2$ の場合には，最悪ページフォールト率は $l < n-2$ のとき 1，$l \geq n-2$ のとき $\frac{n-2}{n-1}$ となる．

証明: $l \leq n-2$ のときは定理 3.2，3.6より明らかなので，$l > n-2$ の場合を考える．

まず A_l による上限であるが，任意の時点 i でキューに入っている要求列を $r_i, r_{i+1}, \cdots, r_{i+n-2} \cdots, r_{i+l}$ とする ($k = 2$ より $l > n-2$)．このとき，この要求列の最初から $n-1$ 個のページ (つまり $r_i, r_{i+1}, \cdots, r_{i+n-2}$) を A_l で処理したときにページフォールトは $n-2$ 回以内になることを示す．高速メモリ内のページを $S_{i-1} = \{a, b\}$ とする．この $r_i, r_{i+1}, \cdots, r_{i+n-2}$ の中に，a または b が存在する場合にはそこでページフォールトが 1 回起こらないで済む．a も b も存在しない場合には $r_i, r_{i+1}, \cdots, r_{i+n-2}$ の中に同じページが 2 回以上現れていることになるので，そこで 1 回ページフォールトが起こらないで済む．よって，最悪ページフォールト率の上限は $\frac{n-2}{n-1}$ となる．

逆に下限の方であるが，全ページの列 $1, 2, \cdots, n$ が連続して現れるような要求の入力を考える．ここから任意に取り出した連続した $n-1$ 個のページ列 $r_i, r_{i+1}, \cdots, r_{i+n-2}$ と，高速メモリ内のページ $S_{i-1} = \{a, b\}$ に対して，a と b のうちどちらか一方だけしか $r_i, r_{i+1}, \cdots, r_{i+n-2}$ に入っていないので，どんな l 先読みアルゴリズムも少なくとも $n-2$ 回のページフォールトを起こす．よって，最悪ページフォールトの下限は $\frac{n-2}{n-1}$ となる． □

4 今後の課題

今後さらに，$l > n-k$，$k \geq 3$ の場合について最悪ページフォールト率の解析を詳しく行い，先読みページング問題のゲームの値，および最適な A_l，Φ を求めていく予定である．

また，他の先読みモデル (資源限定先読み，強先読み，自然先読み) については，競合比的立場から先読みアルゴリズムが示されているが，これらのアルゴリズムがそれぞれのモデルにおいてゲーム的に最適であるかどうか検討することも今後の重要な課題であるといえる．

参考文献

[Alb93] Albers, S., "The influence of lookahead in competitive paging algorithms," In *Proc. 1st European Symposium on Algorithms* **LNCS726** (1993) 1–12.

[BB94] Ben-David, S. and Borodin, A., "A new measure for the study of on-line algorithms," *Algorithmica* **11** (1994) 73–91.

[Bel66] Belady, L.A., "A study of replacement algorithms for virtual storage computers," *IBM Systems Journal* **5** (1966) 78–101.

[Bre96] Breslauer, D., "On competitive on-line paging with lookahead," In *Proc. 13rd Symp. on Theoretical Aspects of Computer Science* (1996) 593–603.

[KP94] Koutsoupias, E. and Papadimitriou, C.H., "Beyond competitive analysis," In *Proc. 35th IEEE Symp. on Foundations of Computer Science* (1994) 394–400.

[MGST70] Mattison, R.L., Gecsei, J., Slutz, D.R., and Traiger, I.L., "Evaluation techniques for storage hierarchies," *IBM Systems Journal* **9(2)** (1970).

[MMS90] Manasse, M.S., McGeoch, L.A., and Sleator, D.D., "Competitive algorithms for server problems," *J. Algorithms* **11** (1990) 208–230; also in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing* (1988) 322-333.

[ST85] Sleator, D.D. and Tarjan, R.E., "Amortized efficiency of list update and paging rules," *Communications of the ACM* **28** (1985) 202–208.

[You91] Young, N., "Competitive paging and dual-guided on-line weighted caching and matching algorithms," *PhD thesis, Dept. of Computer Science, Princeton University* (1991).

[YS96] 山家 明男, 櫻井 幸一, "ランダムな要求に対する k-server 問題の計算複雑性," 電子情報通信学会コンピュテーション研究会 COMP96-25 (1996).