アルゴリズム 33－7
（1993．5．28）

分散共有メモリに対するオンラインアルゴリズム

古賀 久志
東京大学理学部情報科学科今井研究室

ローカルメモリを持った複数のプロセッサーからなる分散共有メモリシステムにおいては、各ページはメモリアクセス要求列に対してコストが低くなるように、適宜移動または複写されて適当なプロセッサーに配置される必要がある。本研究では、こうした低いコストのページ配置を実現するオンラインアルゴリズムをコンペティティブ比（最適オフラインアルゴリズムとのコスト比）の面から考える。

特に本研究では、ページ移動問題に対しての確率的アルゴリズムをページ複写問題用に改良して木やリングのネットワークに対して既存の決定的アルゴリズムよりコンペティティブ比の面で強い結果が得た。

On-line Algorithms for Distributed Shared Memory

Hisashi Koga
Department of Information Science, Faculty of Science, University of Tokyo

In a distributed shared memory system consisting of multiple processors, each of which has its own local memory, each page needs to be located at a proper processor by migration or replication so as to make the total cost for memory-access lower. In this paper, on-line algorithms attempting to implement this low-cost locating are considered in terms of competitiveness, the ratio of the cost of the on-line algorithms to that of the off-line optimal algorithm.

Especially in this paper, we show that the application of algorithms based on randomized algorithms for page migration problem to page replication problem can acquire more powerful result than existing deterministic algorithms about competitive ratio.

1 前書き

一般に分散共有メモリシステムは各々がローカルメモリを持った複数のプロセッサのネットワークとして構成される。そのような設計においては仮想記憶の概念を用いて一つの大きな仮想アドレス空間を実現することができる。

そうした分散共有メモリシステムにおいては、あるプロセッサ（q とする）がページ b のアドレス a にアクセス要求（以下リクエスト）を出す時、まず q は自分のローカルメモリの中にページ b が存在しているかどうかを調べる。もしあれば余計な転送コストなしに b にアクセスできる。一方、もしなければ q は b を持っているプロセッサ（p とする）を見つけアクセス要請を出さねばならない。要請を受けた p はアドレス a の値を q に返すわけだが、この時 p, q 間の距離に比例した転送コストがかかる。そこで、もし、q が何度もページ b にアクセスするようであれば単に a の値を返すだけでなく、b 全体を q に転送したほうがかえってかかるコストが低くなるかも知れない。なぜなら、一旦ページを q に持ってくればそれ以降 q は余計なコストなしに b へアクセス可能だからである。しかし、ページ全体を転送するにはページの大きさに比例した莫大なコストがかかるのでその間にトレードオフが生じる。

本論文では、1つのページについてページアクセスのリクエスト列に対してコストを低く押えるためにどのようにオンラインでページを動かしていけばいいかという問題に対するアルゴリズムを、コンペティティブ比（最適オフラインアルゴリズムとのコスト比）の面から考えてみた。

書き込み可能ページについての上記の問題をページ移動問題と呼び、読みだし専用ページに対しての上記の問題をページ複写問題という。これら2つの問題の差異は、書き込み可能ページは一貫性を保つためにただ1つのコピーしか存在してはいけないのに対し読みだし専用ページについては複数のコピーが同時に存在できるということである。

これまで決定的アルゴリズムでは、いかなるネットワークに対してもページ移動問題では3コンペティティブが下限であり、ページ複写問題については2コンペティティブが下限であることが知られている。また、実際に複写問題については木に対して2コンペティティブなアルゴリズムが存在する [2]。

一方、確率的アルゴリズムではページ移動問題において、一様グラフに対しては 2.28 コンペティティブなアルゴリズムが見つかっており、またコインフリッピングアルゴリズム（以下 CF）が任意のネットワークについても3コンペティティブであることが知られている等、決定的アルゴリズムより強い結果が得られている [4]。

そこで、本研究ではこうした確率的アルゴリズムをページ複写問題に適用して、木に対して約 1.71 コンペティティブなアルゴリズムを示した。さらに CF を改良すると、木に対して2コンペティティブ、リングに対して4コンペティティブとなることを示した。

2 準備

2.1 ページ複写問題のモデル化

本稿では、プロセッサのネットワークを無向連結グラフによりモデル化する。グラフのノードはプロセッサに対応し、枝は隣接プロセッサ間のリンクに対応する。枝の長さはリンクの長さを表す。さらに、δ_{ij} をノード i, j 間の最短パスの長さと定義する。

こうした設定の下で、特定のページ b に関するページ複写に着目する。ノード q からの b へのリクエストは、もし b が q のローカルメモリ中にあれば、0 のコストで処理される。しかし、もし b が q のローカルメモリ中になければ、リクエストは b を既に持っている q にもっとも近いノード p によって処理されるが、この時 p, q 間のパスの長さ（δ_{pq}）だけのコストがかかる。さらに、p がリクエストを処理した後、p は q にページ全体を複写することが許される。この時かかるコストは $r\delta_{pq}$ である。ただし、r はページサイズに比例する1より大きい定数。

以上がコストの定義であるが、後にリクエストが q にもっとも近いページをすでに持ったノードから最短パスに沿って処理されるとは限らないアルゴリズムを考える。その時も2つのノード間のパスの長さに比例するコストがかかるので、そのコストをうまく表現する距離関数 D を定義する必要がある。

本稿では、理想的に各ローカルメモリは無限の容量を持つものとする。この場合、1つのページに着目して解析すれば良い。

ページ複写問題とは、ページ b への有限長のリクエスト列を処理する際、最初はただ一つのノード s のみがページ b を持っているという条件で処理コストが低くなるようにどのプロセッサにページコピーを持たせればいいかをオンラインで決めていく問題である。

本研究では、解析の簡単のためにページ複写に対して以下の二つの仮定をした。

- 一度ページコピーを持ったノードは決してそのページをドロップすることがないものとする。
- ノードは自分に隣接したノードにしかページ複写はできない。

これら二つの仮定からページを持っているノードの集合は常にグラフの連結成分になる。

2.2 コンペティティブネスによる評価

オンラインアルゴリズムとは、将来のリクエストを知らないで現在のリクエストを処理するアルゴリズムである。本研究では、特に Sleator,Tarjan [3] によって導入されたコンペティティブネス (competitiveness) の点から解析を行なっている。この解析方法は、従来の最悪値評価よりも現実を良く反映しており、平均値評価よりも入力に確率的な分布を仮定していないという点で力強いといわれている。また、実際にコンペティティブネスによる評価で優れているアルゴリズムが平均値評価ですぐれているアルゴリズムよりも現実的に良くなるようなオンライン問題もあることが経験的に知られている。

コンペティティブネスの定義は以下のようになる。リクエスト列σを処理するのに要するアルゴリズムAのコストを$C_{\mathrm{A}}(\sigma)$と表すことにしよう。

決定的アルゴリズムAは任意のリクエスト列σに対して、定数bが存在し、以下のような条件が成立する時cコンペティティブであるという。

$$C_{\mathrm{A}(\sigma)} \leq c \cdot C_{\mathrm{OPT}}(\sigma) + b$$

ここで、OPTとはリクエスト列全体を前もって知っていることにより最小のコストでリクエスト列を処理するアルゴリズムのことである。しかし、オンライン環境ではOPTアルゴリズムは使用できない。

一方、確率的アルゴリズムBについては、アルゴリズムの確率的な振舞いについてコストの期待値を考え、決定的アルゴリズムと同じように定義する。つまり、任意のリクエスト列σに対して、定数bが存在し、以下のような条件が成立する時cコンペティティブであるという。

$$\mathrm{E}[C_{\mathrm{B}}(\sigma)] \leq c \cdot C_{\mathrm{OPT}}(\sigma) + b$$

以上がコンペティティブネスの定義であったが、ページ複写問題では最初にすべてのノードにページ複写するアルゴリズムを考えると、定数bに最初の複写のコストの総和を代入してしまえば、0コンペティティブになり問題が意味を持たなくなる。

そこで、本稿では以下のようにコンペティティブネスを再定義する。決定的アルゴリズムAは任意のリクエスト列σに対して、以下のような条件が成立する時cコンペティティブであるという。

$$C_{\mathrm{A}(\sigma)} \leq c \cdot C_{\mathrm{OPT}}(\sigma)$$

確率的アルゴリズムBは任意のリクエスト列σに対して、以下のような条件が成立する時cコンペティティブであるという。

$$\mathrm{E}[C_{\mathrm{B}}(\sigma)] \leq c \cdot C_{\mathrm{OPT}}(\sigma)$$

確率的なアルゴリズムでは最悪の場合を均すことにより決定的アルゴリズムより平均的に良い結果を得られる[1]。

3 カウンタを用いた確率的アルゴリズム

3.1 2ノードに対するアルゴリズム

2ノードはノードs,tからなっているものとし、sが最初にページを持っているノードとする。ここでは、次のようなアルゴリズムを考える。

> **Algorithm ランダムカウント1**
> ノードtにカウンタc_tを持たせる。c_tは、0から$l-1$までの間でランダムに初期化される(lはパラメータでrより大きい)。tからリクエストが生じる度にc_tがインクリメントされ、c_tがlになれば、tへページが複写される。

定理1 ランダムカウント1はμコンペティティブである。ただし$\mu = \max\left\{1+\frac{r}{l}, 1+\frac{l+1}{2r}\right\}$

証明 kをtからのリクエスト回数とおく。また、$\delta_{st}=1$とおいても一般性は失わない。
c_tがmに初期化されたとしよう$(0 \leq m \leq l-1)$。

- もし$(k+m) < l$なら、複写は起こらずかかるコストはkである。$\cdots$ケース(A)
- もし$(k+m) \geq l$なら、$(l-m)$番目のリクエスト後に複写が起こり$l+r-m$のコストがかかる。$\cdots$ケース(B)

これらの事実から、コストの期待値を求めるには以下の2つに場合分けを考えればよい。

- $k \geq l$: かならずケース(B)が起こり期待値は$\frac{1}{l}\sum_{m=0}^{l-1}(l+r-m) = r + \frac{l+1}{2}$
- $k < l$: $m < l-k$なら、ケース(A)が起こり、それ以外はケース(B)が起こる。ゆえに期待値は

$$\frac{1}{l}\sum_{m=0}^{l-k-1} k + \frac{1}{l}\sum_{m=l-k}^{l-1}(l+r-m) = \frac{-k^2+2k(l+r)+k}{2l}$$

一方最適オフラインアルゴリズムのコストは、$k \geq r$なら、最初にtに複写するのでこの時のコストはr。$k < r$なら、最後までtに複写しないのでコストはk。よって、ランダムカウント1のコンペティティブ比は以下のようになる。

	RANDOM COUNT / OPT
$k < r$	$\frac{-k+2(l+r)+1}{2l}$
$r \leq k < l$	$\frac{-k^2+2(l+r)k+k}{2lr}$
$k \geq l$	$1+\frac{l+1}{2r}$

$\frac{-k+2(l+r)+1}{2l}$は、$k<r$の時、$1+\frac{r}{l}$より小さく、また$\frac{-k^2+2(l+r)k+k}{2lr}$は$r \leq k < l$の時$1+\frac{l+1}{2r}$より小さいので定理1はいえたことになる。■

特に、$l = \frac{\sqrt{1+8r^2}-1}{2}$とおくと、

$$\mu = 1 + \frac{\sqrt{8r^2+1}+1}{4r}$$

となりこの値は$r \to \infty$のとき、$1+\frac{\sqrt{2}}{2} \simeq 1.71$に近付くので、$r$が大きければランダムカウント1は2コンペティティブ以下であることがいえる。

3.2 木への拡張

ここでは、上記のアルゴリズムをツリーへ拡張してみた。

> **Algorithm ランダムカウント2**
> 木の各ノードvはカウンタc_vを持っている。アルゴリズムの始動時にすべてのカウンタは0から$l-1$までの間のある同じ値にランダ

ムに初期化される（lはパラメータでrより大きい）。ページをまだ持っていないノードiがリクエストを出した時、iとiにもっとも近いすでにページを持ったノード間のパス上にあるすべてのノードのカウンタをインクリメントする。そして、カウンタがlになったノードにページが複写される。

ネットワークとして木を考えているので、任意の2ノード間のパスはただ一つしか存在しない。また、sからの距離が遠いノードほど対応するカウンタが小さくなる。従ってページを持っているノードはツリーの連結成分になる。

定理 2 ランダムカウント2はμコンペティティブである。ただし$\mu = \max\left\{1+\frac{r}{l}, 1+\frac{l+1}{2r}\right\}$

証明の方針 コストの定義から考えて全体のコストを枝に分割して考えることができる。つまり、ある枝をページ複写、リクエスト処理に使用されたパスが含んでいる時、その枝の長さに対応したコストがかかると考えられる。ここで、各枝に注目してみるとランダムカウント2はランダムカウント1と同じように動いているのがわかる。すなわち、末端のノードのカウンタはランダムに初期化され、カウンタがインクリメントされるたびにその枝の長さだけのコストがかかり、さらにカウンタがlになると複写が末端のノードになされる。各枝にかかるコストの総和が全体としてのアルゴリズムのコストであり、各枝についてはμコンペティティブなので、ランダムカウント2もμコンペティティブである。 ■

4 コインフリッピングアルゴリズム

Algorithm コインフリッピングアルゴリズム (Coin Flipping Algorithm)
コインフリッピングアルゴリズムとは、リクエストが生じるたびにリクエストを出したノードがまだページを持っていなければ、ページを既に持つ最も近いノードからそのノードに確率pでページ複写するというアルゴリズムである。しかし、複写は隣接したノードにしか許されないので、離れたノードへの複写はそのノードへのパス上のノードに順々に複写していくことによってなされる。

CFについては、Westbrook[4]がページ移動問題で$p = \frac{1}{2r}$としてやるといかなるネットワークについても3コンペティティブであることを示している。

4.1 木に対して

定理 3 CFは$p = \frac{1}{r}$のとき2コンペティティブである。

証明の方針 いかなるアルゴリズムA（最適オフラインアルゴリズムを含んだ）について$\mathrm{E}[C_{\mathrm{CF}}(\sigma)] \leq 2C_{\mathrm{A}}(\sigma)$となることを示す。

まず、リクエスト列σに対してAとCFを並列実行させ、AとCFの動作をマージしてみる。マージされてできたAとCFの動作の列は以下の2種類のイベントから構成される。

1. Aがページを複写する。
2. AとCFがリクエストを処理する。これはCFのページ複写を伴うこともある。

ここで、これら全てのイベントについて以下の不等式が成立するような非負のポテンシャルΦ（初期値は0）を見つけてやる（Δはイベントの結果としての数量の変化）。

$$2\Delta C_{\mathrm{A}} - \mathrm{E}[\Delta C_{\mathrm{CF}}] \geq \mathrm{E}[\Delta\Phi] \qquad (1)$$

この不等式を全てのイベントについて加えてやると、

$$2C_{\mathrm{A}} - \mathrm{E}[C_{\mathrm{CF}}] \geq \mathrm{E}[\Phi_{\mathrm{end}}] - \mathrm{E}[\Phi_{\mathrm{start}}]$$

となり、あとは$\Phi_{\mathrm{start}} = 0$かつ、$\Phi_{\mathrm{end}} \geq 0$であることから定理3がいえることになる。

定義 1 アルゴリズムBにおいてページを持っていないあるノード（dとしよう）に対する境界ノードとは、Bにおいてページを持っているパスs,d上のsからもっとも離れたノードのことである。ここでは、木を考えているのでs,d間のパスは一つしかなく、従って境界ノードも一意に定まる。

定理3の証明 Sをアルゴリズムのみでページを持っているノードの集合とする。そして、ポテンシャルを

$$\Phi = 2r\sum_{j\in S}\delta_{jp(j)}$$

とする。ここで$p(j)$とはsをツリーのルートと見た時のjの親ノードのことである。明らかに、$\Phi \geq 0$であり、Φの初期値は0である。

これから(1)が各イベントについて成立することを示す。

1. Aがページを複写する。複写元ノードをk、複写先ノードをjとする。この時$\Delta C_{\mathrm{A}} = r\delta_{kj}$、$\Delta C_{\mathrm{CF}} = 0$である。一方、ポテンシャルの変化は$j$が複写後に$S$に属するかで場合わけされる。

   $j \notin S$: $\Delta\Phi = 0$
   $j \in S$: 複写前に$k \in S$ なら$\Delta\Phi = 2r\delta_{kj}$、そうでない時は$\Delta\Phi \leq 2r\delta_{kj}$

   いずれの場合も(1)が成立するのがわかる。

2. AとCFがリクエストを処理する場合。dをリクエストを出しているノードとする。

   - dがすでにCFでページを持っている場合
     この時、$\Delta C_{\mathrm{CF}} = 0$、$\Delta C_{\mathrm{A}}$は少なくとも0より大きい。また、CFでは複写が起こらないので$\mathrm{E}[\Delta\Phi] = 0$、となり(1)は満足される。
   - dがまだCFでページを持っていない場合
     iをCFにおけるdの境界ノードとする。CFがリクエストを処理するのにδ_{id}かかり、その後にCFが複写する確率は$\frac{1}{r}$なので

     $$\mathrm{E}[\Delta C_{\mathrm{CF}}] = \delta_{id} + \frac{1}{r}r\delta_{id} = 2\delta_{id}$$

     Aについてはリクエスト発生時Aがどこまで複写しているかによって3つの場合にわかれる。最初の2つの場合jをAにおけるdの境界ノードとする。

Aがiまでまだ複写していない場合。
$\Delta C_{\mathrm{A}} = \delta_{jd}$であり、ポテンシャルはCFで複写が起こるかどうかに関係なく不変なので
$\mathrm{E}[\Delta\Phi] \leq 2\delta_{jd} - 2\delta_{id} = 2\delta_{ij}$
となり (1) 成立。

Aがページをiを越えて複写しているが、dまでは複写していない場合。$\Delta C_{\mathrm{A}} = \delta_{jd}$。一方、CFで複写が起こらない時、$\Delta\Phi = 0$であり、複写が起これば$\Delta\Phi = -2r\delta_{ij}$なので、
$\mathrm{E}[\Delta\Phi] = -\frac{1}{r} \cdot 2r\delta_{ij} = -2\delta_{ij}$ となり (1) 成立。

Aがページをdを越えて複写している場合。$\Delta C_{\mathrm{A}} = 0$。またCFで複写が起こらない時、$\Delta\Phi = 0$であり、複写が起これば$\Delta\Phi = -2r\delta_{id}$なので
$\mathrm{E}[\Delta\Phi] = \frac{1}{r} \cdot (-2r\delta_{id}) = -2\delta_{id}$ となり (1) 成立。

以上で証明終了 ■

4.2 リングについて

ここでは、CFを改造してリングに対して4コンペティティブであることを示す。しかしこのアルゴリズムはリクエストを出したノードにもっとも近いすでにページをもつノードによってリクエストが処理されるという仮定を満たしていない。

sを最初にページを持っているノードとする。ここで図1のようにsを最初にページを持っている2つのノードs, s' (s, s'間の最短パス$(\overline{ss'})$の長さは0) に置換しても状況は不変である。そこで、ここからはそのようなリングを考える。sから時計回りに複写が起こり、s'から反時計回りに複写が起こる。

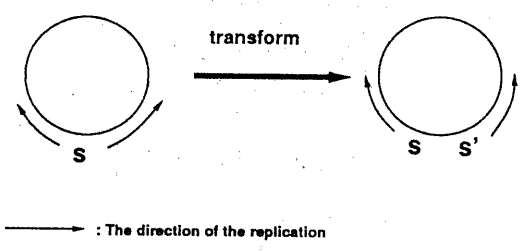

図 1: Transformation of G into G'

まず、コストを表す新しい距離関数Dを定義する。リングには任意の2ノード間に2つのパスが存在する。D_{ij}は以下のように定義される。

$$D_{ij} = \begin{cases} \text{if } i \neq j & \overline{ss'} \text{を含んでいない方の} \\ & \text{パス } ij \text{の長さ} \\ \text{if } i = j & 0 \end{cases}$$

この距離関数の定義は不自然ではない。なぜなら、s, s'は既にページを持っているので$\overline{ss'}$を含んだパスはページアクセス操作に使われることはないからである。

定義 2 (左境界点) アルゴリズムAのある時点での左境界点とは、Aでその時点までに時計回りにページを複写された末端のノードのことである。最初はどんなアルゴリズムでも左境界点はs。

定義 3 (右境界点) アルゴリズムAのある時点での右境界点とは、Aでその時点までに反時計回りにページを複写された末端のノードのことである。最初はどんなアルゴリズムでも右境界点はs'。

定義 4 (ノードカバー) アルゴリズムAのノードカバーとはL_{A}をAの左境界点、R_{A}をAの右境界点としたときの弧$L_{\mathrm{A}}sR_{\mathrm{A}}$のこと

Algorithm 変形されたCF(Transformed CF)
リクエストがあるノードdで出された時、dがすでにページを持っていればコスト0で処理される。もし、dにページがまだなければリクエストは$\frac{D_{ad}}{D_{ab}}$の確率でbに処理され、$\frac{D_{bd}}{D_{ab}}$の確率でaによって処理される。リクエスト処理後にリクエストを処理したノードからdに$\frac{1}{r}$の確率でページ複写がなされる。その際、通常のCFのようにリクエストを処理したノードからd間のパス上の全てのノードにもページは複写される。ただし、a, bはそれぞれリクエストが生じた時の左境界点、右境界点である。

定理 4 TCFは4コンペティティブである。

証明
木の場合と同じように非負のポテンシャルを定義し、いかなるアルゴリズムAについても、任意のリクエスト列σに対してAとTCFを並列実行させた時の動作をマージした動作列に含まれる2種類のイベントについて

$$4\Delta C_{\mathrm{A}} - \mathrm{E}[\Delta C_{\mathrm{TCF}}] \geq \mathrm{E}[\Delta\Phi] \qquad (2)$$

となることを示す。
ポテンシャルを以下のように定義する。

$$\Phi = 4r \times \text{length of } (\text{Aのノードカバー} \bigcap (\text{TCFのノードカバー})^{c})$$

明らかに、$\Phi \geq 0$でΦの初期値は0である。
これから先は (2) が各イベントについて成立することを示す。

1. A がページを複写する。複写元ノードを k、複写先ノードを j とする。この時、$\Delta C_{\mathrm{A}} = r\delta_{kj}$ で $\Delta C_{\mathrm{TCF}} = 0$ である。ポテンシャルの変化は複写前に k, j が TCF のノードカバーに含まれているかによって場合わけされるが、ポテンシャルの変化の期待値が最大になるのは（$k, j \notin$ TCF のノードカバー）の時で、この時 $\mathrm{E}[\Delta\Phi] = 4r\delta_{kj}$ となる。一方、$4\Delta C_{\mathrm{A}} - \mathrm{E}[\Delta C_{\mathrm{TCF}}] = 4r\delta_{kj}$ なので、(2) 成立。

2. A と CF がリクエストを処理する場合。d をリクエストを出したノードとする。a, b をそれぞれリクエスト発生時の TCF の左境界点、右境界点とし、x, y をそれぞれリクエスト発生時の A の左境界点、右境界点とする。

   - d が TCF のノードカバーにリクエストの出される前に含まれている場合。$\Delta C_{\mathrm{TCF}} = 0$ で、ΔC_{A} は 0 より大きい。TCF では複写は起こらないので $\mathrm{E}[\Delta\Phi] = 0$ となり (2) 成立
   - d が TCF のノードカバーにリクエストの出される前に含まれていない場合
     最初に、以下のようなノードの順序を定義する。

$$m < n \quad \text{if} \quad D_{sm} < D_{sn}$$

例えば、図 2 (a) では $x < a < b < y$ となる。明らかに、$a < b$、$x < y$ なので

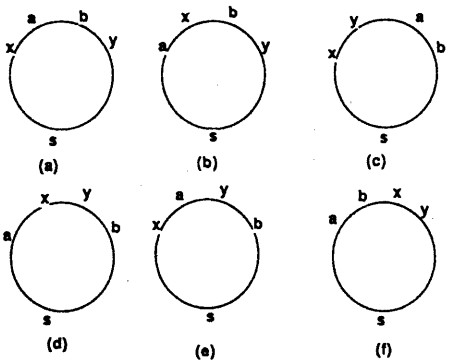

図 2: Six cases about the locations of a,b,x,and y

a, b, x, y の順序は 6 通りある（図 2）。しかし、(e),(f) は (b),(c) を左右反転させたものなので考慮する必要はない。また、d は TCF のノードカバーに含まれていないので常に $a < d < b$。以下は (a),(b),(c),(d) の 4 通りの場合について考える。しかし、すべての場合について

$$\begin{aligned} & \mathrm{E}[\Delta C_{\mathrm{TCF}}] \\ = \ & \frac{D_{bd}}{D_{ab}}(D_{ad} + \frac{1}{r}rD_{ad}) \\ & + \frac{D_{ad}}{D_{ab}}(D_{bd} + \frac{1}{r}rD_{bd}) \\ = \ & 4\frac{D_{ad}D_{bd}}{D_{ab}} \end{aligned}$$

ここでは紙面の都合上、ケース (a), ケース (b) の場合についてのみ証明しておく。

ケース (a) では、
$\Delta C_{\mathrm{A}} \geq \min(D_{xd}, D_{yd})$、$\mathrm{E}[\Delta\Phi] = 0$ だが、

$$\begin{aligned} \mathrm{E}[\Delta C_{\mathrm{TCF}}] &= 4\frac{D_{ad}D_{bd}}{D_{ab}} \\ &\leq 4D_{bd} \\ &\leq 4D_{yd} \end{aligned}$$

$$\begin{aligned} \mathrm{E}[\Delta C_{\mathrm{TCF}}] &= 4\frac{D_{ad}D_{bd}}{D_{ab}} \\ &\leq 4D_{ad} \\ &\leq 4D_{xd} \end{aligned}$$

となって、(2) 成立。

ケース (b) では d の位置について 2 通りの場合がある。
もし、$a < d \leq x$ なら、A は d にページを持っているので $\Delta C_{\mathrm{A}} = 0$、また、ポテンシャルの変化はリクエストが a によって処理される時、$\frac{1}{r}(-4rD_{ad})$ であり、b によって処理されるとき $\frac{1}{r}(-4rD_{xd})$ であるので、結局
$\mathrm{E}[\Delta\Phi] = -4\frac{D_{ad}D_{bd}}{D_{ab}} - 4\frac{D_{xd}D_{ad}}{D_{ab}}$。
ここで、

$$\begin{aligned} & 4\Delta C_{\mathrm{A}} - \mathrm{E}[\Delta C_{\mathrm{TCF}}] \\ = \ & 0 - 4\frac{D_{ad}D_{bd}}{D_{ab}} \\ \geq \ & -4\frac{D_{ad}D_{bd}}{D_{ab}} - 4\frac{D_{xd}D_{ad}}{D_{ab}} \end{aligned}$$

となって (2) 成立。

また、$x < d < b$ なら、$\Delta C_{\mathrm{A}} \geq \min(D_{xd}, D_{yd})$ である。ポテンシャルはリクエストが TCF で a によって処理される時のみ変化し、その期待値は $-4\frac{D_{bd}D_{ax}}{D_{ab}}$。したがって、

$$\mathrm{E}[\Delta C_{\mathrm{TCF}}] + \mathrm{E}[\Delta\Phi] = 4\frac{D_{bd}D_{xd}}{D_{ab}}$$

となる。ここで、

$$4D_{xd} - 4\frac{D_{bd}D_{xd}}{D_{ab}} = 4D_{xd}(1 - \frac{D_{bd}}{D_{ab}}) \geq 0$$

$$4D_{yd} - 4\frac{D_{bd}D_{xd}}{D_{ab}} \geq 4D_{yd} - 4D_{bd} = 4D_{by} \geq 0$$

となるので (2) が成り立っているのがわかる。

5 結論

本研究では、Westbrook [4] の確率的オンラインアルゴリズムをページ複写問題に適用した。特に本研究ではオンラインアルゴリズムのコンペティティブネスに着目してみた。その結果、木に対しては決定性アルゴリズムの下限である2コンペティティブを破る事ができた。また、コインフリッピングアルゴリズムが木に対して2コンペティティブになることを示した。さらに、コインフリッピングアルゴリズムを改良することによりリングに対して4コンペティティブのアルゴリズムが存在することを示した。これは、決定性アルゴリズムでは木という2点間のパスが1本しかない特殊な場合にしか結果が出ていないのと比較すると、より一般のグラフに近付いたという点で確率的なアルゴリズムの力強さを実証しているといえるだろう。今後の課題としては、コストの定義をより妥当なものにすることやや複写の方法についての人為的な制限を取り払ったより現実に近いモデルを考えてみるということが挙げられる。

6 謝辞

この1年間、私の研究を支えてくれた今井助教授並びに、今井研究室の皆さんに感謝します。

参考文献

[1] S. Ben-David et al. "On the power of randomization in on-line algorithms". In *Proceedings of the 22th ACM Symposium on Theory of Computing*, pages 379–386, 1990.

[2] D.L. Black and D.D. Sleator. "Comptitive algorithms for replication and migration problem". Technical Report CMU-CS-89-201, Department of Computer Science, Carnegie Mellon University, 1989.

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

[4] J. Westbrook. "Randomized algorithm for multiprocessor page migration". In *Proceedings of a DIMACS Workshop*, pages 135–149, 1991.