1

キャッシュのモデル化とその応用

田中　淳裕

キャッシュは，データの再参照を高速に行うために情報通信分野のいたるところで用いられている最適化技術の一種である．簡単なモデル化の結果として，キャッシュのヒット率（キャッシュ中に存在するデータが再度参照される確率）がデータへの参照間隔によって特徴付けられることを示し，参照間隔の推定という立場で，一般によく知られている置き換えアルゴリズムの考察を行う．さらに，キャッシュミスが起きた時の参照間隔分布と，分割キャッシュモデルとを組み合わせることによって，キャッシュのヒット率が改善された例を紹介する．

キーワード：キャッシュ，参照間隔，置き換えアルゴリズム

1. はじめに

キャッシュは高速に動作する記憶装置であり，動作速度が比較的遅い記憶装置の前段階に配置して，データの参照結果や検索結果を一時的に保存するために用いられる．同一データに対する検索・参照が再度起きたときには，キャッシュ中に残っているデータを利用すれば再検索・再参照を高速に行うことができる．高速動作を特徴とするキャッシュの記憶容量は小さくならざるを得ず，そのため，一杯になったキャッシュの中から何を捨て，何を残すかを決定する数多くのキャッシュ管理アルゴリズムが提案されてきている．

キャッシュの応用範囲は広い．計算機におけるCPUキャッシュや，OSが管理するファイルシステムのキャッシュ，あるいはデータベースの検索速度を改善するためのキャッシュ，さらにはネットワーク上に散在するWebオブジェクトの複製をまとめて手近な場所に管理するためのキャッシュサーバ（プロキシサーバ）など，情報通信分野のいたるところでキャッシュが用いられている．

World Wide Webの爆発的な普及に伴い，計算機分野でのキャッシュの研究をより拡張・加速する形で，多くのキャッシュアルゴリズムが提案されている．特に，Webが対象とするデータには多種多様なものが混ざっており，例えば，数バイトのアイコンから数百メガバイトを超えるような画像データ，あるいは1ヶ月に数度しか参照されることのないようなものから，1日に数百万アクセスを超えるようなものまでを，統一して管理できるアルゴリズムが求められている．

一方では，それらのアルゴリズムがなぜ効率的に働くのかを解析するための数理的な評価モデルの整備が遅れている．

本稿では，キャッシュのヒット率（あるものが参照されたとき，それが以前にも参照されており，かつキャッシュ中に存在する確率）を導く単純なモデルを紹介し，そのヒット率が，参照間隔によって特徴付けられることを示す．また，参照間隔の推定問題という立場で既存のアルゴリズムの特徴を簡単に整理する．さらには，一つの参照列が，異なる特徴を持つ複数の参照列の重ね合わせとして構成されている場合に対応する分割キャッシュを取り上げ，実例を通して全体性能がどのように改善されるかを解説する．なお，以下では参照される「もの」をオブジェクトxとし，（離散）時刻tにて参照されたオブジェクトをx_tとし，参照列を$\cdots x_t x_{t+1}\cdots$などと記述する．

2. キャッシュのモデルとアルゴリズム

2.1 ワーキングセットモデルと参照間隔

キャッシュは一度参照されたオブジェクトが再度参照されるときに初めて効力を発揮する．そのため，あるオブジェクトが参照されてから，次にそのオブジェクトが参照されるまでの時間間隔に基づく制御が基本となっている．この時間間隔のことを参照間隔と呼び，あるオブジェクトxの参照間隔D_xは次のように定義される．

$$D_x := \min\{s \geq 0 | x_t = x \text{ and } x_{t-s} = x\} \quad (1)$$

これは，いわゆる再帰時間の定義に相当する．

古典的なモデルであるワーキングセット（Working Set）法[1]では，時間ウィンドウと呼ばれる定数

たなか　あつひろ
NECシステムプラットフォーム研究所
〒211-8666 川崎市下沼部1753

Tに対して，時刻$t-(T-1)$から時刻tまでに参照したオブジェクトの部分列$x_{t-(T-1)}\cdots x_t$を考え，その中に時刻$t+1$での参照オブジェクトx_{t+1}が含まれるときをキャッシュヒット，含まれないときをミスと定義している．このとき，ヒット率は

$$\sum_x \pi_x \cdot Pr\{D_x \leq T\} \tag{2}$$

で与えられる．ここで，π_xはオブジェクトxを参照する定常状態確率とする．

式(2)に，ある種の性質のよいマルコフ連鎖において得られている，「極限確率π_xは平均再帰時間$E[D_x]$の逆数に等しい」という結果を適用すると，ヒット率は次のように書き表すことができる．

$$\sum_x \frac{Pr\{D_x \leq T\}}{E[D_x]} \tag{3}$$

2.2 代表的なアルゴリズムの再解釈

前節の結果に従うと，キャッシュ中から廃棄対象オブジェクトを選択するときには，$E[D_x]$と$Pr\{D_x \leq T\}$の大きさに基づいて判断すればよいことがわかる．このような戦略と，代表的なキャッシュアルゴリズムであるLRU（Least Recently Used）とLFU（Least Frequently Used）との戦略が参照間隔の推定という形で結びついていることを，以下では簡単に説明する．

LRUは，実現の容易さから幅広く用いられている手法であり，オブジェクトを管理するスタックの最上位に最近参照されたものを配置し，スタックの最下位に最近最も使われていない（Least Recently Used）ものを配置する形でオブジェクトが管理されている．スタックの最下位に位置するオブジェクトが削除対象である．スタックの上位にあるオブジェクトほど近い将来においてもより参照されやすいであろうという直感的な推測に基づくアルゴリズムである．

LRUは，現在時刻tとオブジェクトxを最後（直近）に参照した時刻$Ref(x)$との差分$\Delta(x) := t - Ref(x)$を考え，$\Delta(x)$が小さいオブジェクトxに高い優先順位を与えた管理アルゴリズムと定義することもできる．ここで，オブジェクトがキャッシュ中に滞在する平均滞在時間T'を考えると，LRUアルゴリズムは，$Pr\{\Delta(x) \leq T'\}$が大きいものをキャッシュ中に残し，小さなものを置き換え対象としている．すなわち，D_xの推定値として$\Delta(x)$を用いたアルゴリズムと解釈することができる．

LRUにおいては，前回参照時刻$Ref(x)$より以前の状況は考慮されていない．あくまでも短期的な振る舞いに着目し，長期的な振る舞いから算出される$E[D_x]$は完全に無視されている．当然，前回の参照時刻$Ref(x)$だけではなく，その前，さらにその前の参照時刻を利用したキャッシュアルゴリズムも考えられており，LRU-kとして知られている[4]．

一方のLFUは，参照頻度の高いものほど再度参照されやすいであろうという推測に基づくアルゴリズムである．キャッシュ中で，最も使われる頻度が小さい（Least Frequently Used）オブジェクトが削除対象である．ある期間における参照頻度がπ_xに相当するとみなすと，LFUは，ある程度長期的な振る舞いを観測することで得られる$E[D_x]$のみを利用したアルゴリズムとみなすことができる．

このように，LRUは参照列が持つ短期的な特徴を，一方のLFUは長期的な特徴を考慮したアルゴリズムということができる．当然，LRUとLFUとの両方の特徴を兼ね備えたアルゴリズムが存在することも容易に想像がつき，LRFUというアルゴリズムが既に提案されている[3]．

2.3 異なるアクセスパターンへの対応

現実の参照列は，異なる種類のオブジェクトに対する参照が入り混じったもので構成されている．例えば，Webの参照系列は，数バイトのアイコンから数百メガバイトを超えるような画像データへの参照，あるいは参照頻度が全く異なるWebオブジェクトへの参照などから構成されている．

このような場合に，オブジェクトの属性（種類：テキスト/画像/音声，サイズの大小，参照頻度など）に基づき，それぞれを個別にキャッシュ管理する分割キャッシュが知られている．StoneらはCPUキャッシュを命令用とデータ用とに分割し，その際の最適な分割条件を導いている[5]．

オブジェクトの参照頻度別にキャッシュの管理方式を分けるタイプのアルゴリズムとして2Q[2]とMultiQ[7]とが知られている．これらはいずれもファイルシステムやデータベースのバッファ管理用に開発されたものである．

2Qでは一つのキャッシュ領域を，1度しか参照されていないオブジェクトをFIFOで管理する領域Q_1と，複数回参照されたオブジェクトをLRUで管理する領域Q_2とに分けて，管理する．これは，1度しか参照されないようなオブジェクトへの参照が短期間に大量に発生した場合に，何度も参照されているオブジェクトがキャッシュから削除されてしまうというLRUの問題点を解決するために提案されたアルゴリ

ズムである．MultiQではオブジェクトの参照回数に応じて三つ以上の管理領域を設定している．

この二つのアルゴリズムは，LRUの欠点を克服してはいるものの，各領域（Q_1とQ_2）のサイズを動的に調整するための手法が欠けており，事前評価を行って調整可能パラメータの値を決定する作業や，ヒューリスティクスとが必要とされていた．

3. 分割キャッシュとその応用

Webオブジェクトのキャッシュ管理において，ミスが発生した時の参照間隔（IIPO[1]）に着目し，分割キャッシュの最適化条件を適用することよって，ヒット率を向上させた例[6]を紹介する．この例では，2Qアルゴリズムの基本的な構造を踏襲した上で，Q_1とQ_2のサイズを動的に最適値に保つ2Q-Optアルゴリズムを提案している．

分割キャッシュにおいては，各領域Q_jを微小量増加させた時のヒット率の改善率を算出し，最も改善率の小さな領域のサイズを減少させ，逆に最も改善率の大きな領域のサイズを増加させるということを行って各領域のサイズを最適値に保つ．2Q-Optにおいては改善率を算出するために，キャッシュミスが起きたときのIIPOの分布を用いて改善率を推測している．

キャッシュミスが発生したときに，そのオブジェクトのIIPOの値がもし小さければ，仮にそのキャッシュのサイズをもう少し大きくしたときにそのキャッシュミスは起きなかったであろうという推測が成り立つ．逆に大きな値であれば，キャッシュサイズを大きくしたところで，相変わらずキャッシュミスが起きていたことと推測できる．

したがって，2Q-Optでは，ある一定期間においてIIPOを各領域ごとに集計し，IIPOの分布が原点付近に集中している方により多くのキャッシュ領域を与えるという戦略をとっている．

図1に2Q-OptとLRUのヒット率の比較結果を示す．ワークロードとしてはWebプロキシサーバのベンチマークとして広く用いられているWeb Polygraph[2]を用いた．キャッシュサイズが大きいときには，置き換え方式による効果の差はほとんど現れないが，キャッシュサイズの小さな領域で，相対比約20%のキャッシュヒット率の向上が得られている．

このような効果が得られた原因は，図2と図3とを見比べることで理解することができる．図2は，LRUアルゴリズムを用いた場合のIIPOの分布を，参照回数が1回のオブジェクトと，複数回参照された

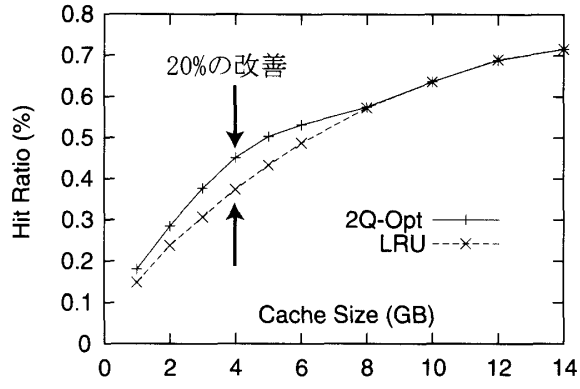
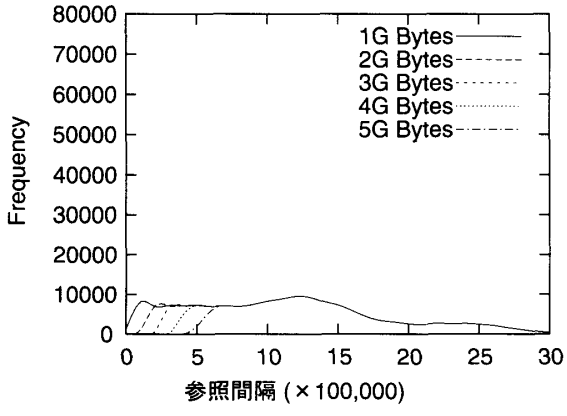
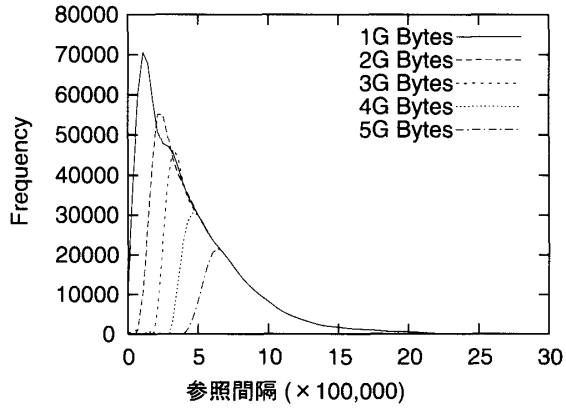


図1　2Q-OptとLRUのヒット率の比較



(a)　参照回数＝1のオブジェクト（LRU）

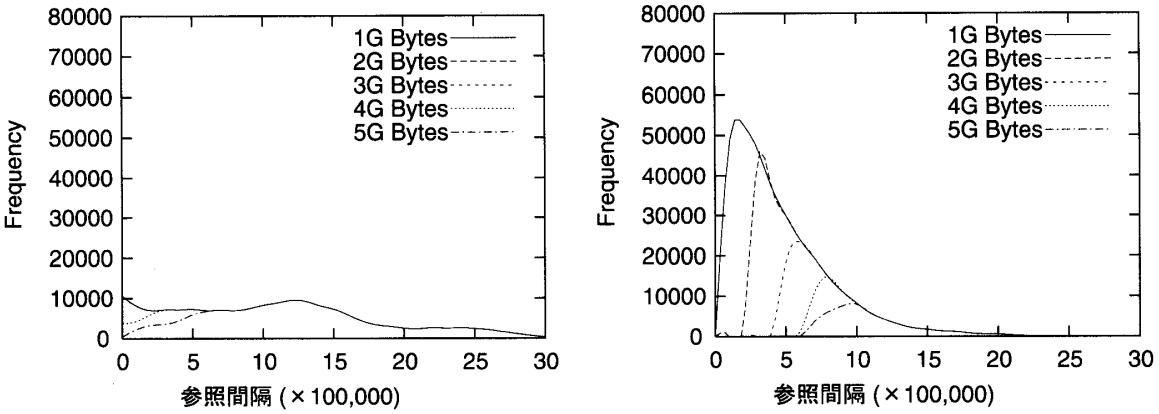


(b)　参照回数≥2のオブジェクト（LRU）

図2　IIPOの分布（LRU）

[1] Interreference Interval for Purged Objects：一度参照されたものの既にキャッシュから削除されてしまったオブジェクトに対して定義される参照間隔の意．

[2] http://www.web-polygraph.org/



(a) 参照回数=1のオブジェクト (2Q-Opt)　　(b) 参照回数≥2のオブジェクト (2Q-Opt)

図3 IIPOの分布 (2Q-Opt)

オブジェクトに分けて，キャッシュサイズを1GBから5GBに変化させてグラフ表示したものである．図3は2Q-Optを用いた場合の同様の結果を示している．

1度しか参照されていないオブジェクトと複数回参照されたオブジェクトのIIPOの分布を比べると，明らかに複数回参照されたオブジェクトの方が原点付近に集中している．したがって，キャッシュサイズを大きくするときには，Q_1ではなくQ_2のサイズを大きくすることで，全体のキャッシュヒット率が大きく改善されることが期待できる．

LRUはオブジェクトの参照回数を区別しないタイプのアルゴリズムなので，キャッシュサイズの増分が，複数回参照のオブジェクトに対しても，1度しか参照されていないオブジェクトに対しても平等に利用されてしまう．一方で2Q-Optの場合は，IIPOの分布の特徴を判断し，Q_2により多くのキャッシュ領域を割り当てるような制御を行う．キャッシュミス時に値の小さなIIPOを与えるオブジェクトが，キャッシュ領域の拡大によってキャッシュヒットとなるからである．

4. おわりに

本稿では，情報通信分野のいたるところで用いられている最適化技術の一つであるキャッシュを取り上げ，簡単なモデル化の結果より参照間隔がキャッシュのヒット率を大きく左右することを示し，参照間隔の推定という立場で一般的に知られているキャッシュアルゴリズムの再解釈を試みた．さらに，キャッシュミスが発生したときの参照間隔分布に着目し，分割キャッシュモデルを適用することで，異なるアクセスパターンを内在するWebの参照系列に対するキャッシュヒット率が向上する例を紹介した．

バッファを用いたシステムの効率的な運用という意味では，キャッシュも待ち行列もお互いに似ている．また，システムに対する客の到着間隔とサーバの能力によって特性が決定される待ち行列と，「もの」が参照される間隔と「もの」を保管する領域の大きさによって性能が決定付けられるキャッシュとの間には，何らかの関係があるようにも思える．今後の研究により，両者の関連が解明されることを望む．

参考文献

[1] P. J. Denning and S. C. Schwartz: Properties of the working-set model, *CACM*, 15 (3), pp. 191-198, 1972.

[2] T. Johnson and D. Shasha: 2Q: A low overhead high performance buffer management replacement algorithm, *Proc. 20th VLDB*, pp. 439-450, 1994.

[3] Lee et al.: On the Existence of a Spectrum of Policies that Subsumes the Least Recently Used (LRU) and Least Frequently Used (LFU) Policies, *Proc. SIGMETRICS'99*, pp. 134-143, 1999.

[4] E. J. O'Neil et al.: The LRU-K Page Replacement Algorithm For Database Disk Buffering, *Proc. SIGMOD'93*, pp. 297-306, 1993.

[5] H. S. Stone et al.: Optimal partitioning of cache memory, *IEEE Trans. Computers*, 41 (9), pp. 1054-1068, 1992.

[6] A. Tanaka and K. Tatsukawa.: Interreference Interval for Purged Objects: A New Metric for Design and Analysis of Web Caching Algorithms, *Proc. IPCCC 2003*, pp. 549-554, 2003.

[7] Y. Zhou et al.: The multi-queue replacement algorithm for second level buffer caches, *Proc. 2001 USENIX Technical Conference*, pp. 91-104, 2001.