WWWトラフィック特性に基づくキャッシュ方式の提案

西川記史[†1] 細川貴史[†2] 森 靖英[†3]
吉田健一[†4] 辻 洋[†5]

ここ数年インターネットを中心とした広域ネットワークが注目を集めている．なかでも，WWWはインターネット上の主要アプリケーションとしての地位を確立した感があり，情報化社会の利便性を象徴するアプリケーションとして期待を集めている．しかし急速な需要の増加にネットワーク・インフラの整備が追い付かず問題ともなっている．このようなネットワーク・トラフィックの増加に対してキャッシュを用意するのは伝統的な対応方法であるが，CPUのメモリアクセス高速化に効果のあった手法を性質の異なるWWWトラフィックに単純に応用するのでは，十分な効果をあげられない．本研究では，初めにWWWトラフィックの特徴について分析し，現在キャッシュ方式の前提とされている低速・大容量の記憶装置の使用が適切でないことを述べる．次に，高速・小容量の記憶装置を前提とした分散キャッシュ技術を提案し，提案技術の評価実験の結果について報告する．

A Proposal for Caching Strategy Based on WWW Traffic Characteristics

Norifumi Nishikawa,[†1] Takafumi Hosokawa,[†2] Yasuhide Mori,[†3]
Kenichi Yoshida[†4] and Hiroshi Tsuji[†5]

The WWW (World Wide Web) service has recently become a widely used network service which symbolizes the benefits of a networked society. By using the WWW, the user can access various types of information easily and quickly. However, a rapid growth in demand sometimes causes a heavy network overload, and the resulting slow-response spoils the benefits of the WWW. Statistics clearly indicate the potential overload in the backbone of the Japanese wide area networks. Thus, the demand for better operating methods of WWW has been increasing. In this paper, we analyze WWW traffic pattern and propose a distributed cache system which is adapted to the WWW traffic patterns. The experimental results show the advantage of our approach over conventional approaches.

1. はじめに

ここ数年インターネットを中心とした広域ネットワークが着目を集めている．なかでも，WWWはインターネット上の主要アプリケーションとしての地位を確立した感があるが，急速な需要の増加にネットワーク・インフラの整備が追い付かず問題となっている．たとえば数年前までネットワーク・トラフィックの統計にはなかったWWWのトラフィックは，現在日本国内のネットワーク・トラフィックの主要な部分を占めるにまで急増しており，需要の増加に応じるために回線容量を増加しても，その翌日から急速にWWWトラフィックが増加し，増加分を占有してしまう状況にある．特に国内から米国へのアクセス量は多く，不足気味の海外回線の容量を圧迫する最大の要因となっている．

このような状況に対して，WWWのトラフィック内部に重複の多いことを利用したキャッシュ・サービスの利用を推進する動きもある．代表的なキャッシングプロキシーとしてSquid[17]やCERNサーバ[16]，Delegateサーバ[20]等が広く用いられている．

一方，キャッシュ技術自身の研究もさかんに行われている．特にキャッシュが満杯になった場合にどのページをキャッシュから捨てるかのキャッシュ置換え方式の研究はキャッシュヒット率改善の手段として広く研究

†1 日立製作所ソフトウェア事業部
Software Division, Hitachi, Ltd.
†2 郵政省電気通信局電気通信事業部高度通信網振興課
Telecommunications Business Department, Ministry of Posts and Telecommunications, Telecommunications Bureau, Advanced Network Division
†3 日立製作所中央研究所
Central Research Laboratory, Hitachi, Ltd.
†4 日立製作所システム開発研究所
Systems Development Laboratory, Hitachi, Ltd.
†5 日立製作所コンシューマネットビジネス推進本部
e-Service & Business Development, Hitachi, Ltd.

されており，CPU のメモリアクセス高速化に効果のあった LRU（Least Recently Used）に，さらにページサイズやアクセス頻度を考慮した方式[1),2)]，応答時間（レイテンシ）向上を目的としてデータ取得に要した時間を考慮した方式[3),4)]等が提案されている．

また，より大規模なネットワークのトラフィック削減や負荷分散を目的として複数のキャッシュをネットワーク上に配置し協調動作させる階層型キャッシュ[5)]や分散型キャッシュ[6)]，ICP（Internet Caching Protocol）[9)]の研究，アクセスの高速化に特化した先読みキャッシュ[14),15)]の研究等もさかんに行われている．

一方，WWW トラフィックの特性に関する研究もさかんである．特に WWW トラフィックが Zipf の法則に従うことが報告されている[8),12)] ことは注目に値する．これは，i 番目のアクセス頻度の WWW ページのアクセス回数は $(1/i)^k$（k はアクセスパターンを決定するパラメータ）に比例するという法則である．この法則は，WWW アクセスパターンは，ごく少数の WWW ページのみが高頻度で再利用され，その他の大部分のページはほとんど再利用される可能性のないページであることを意味している．このページの再利用に関する特性は，キャッシュシステム設計上重要であると考えられるが，上記研究事例ではこの性質に関して十分検討が行われておらず，これがキャッシュの効果を削減している 1 つの要因と考えられる．

そこで，本論文では，まず WWW トラフィックが Zipf の法則に従うという経験則をベースに，その特徴について分析し，分析結果に基づき望ましいキャッシュ技術および構成法について述べる．また，本キャッシュ技術を分散型キャッシュに適用するための方法について検討し，あわせてシミュレーションによる実験結果についても報告する．

2. WWW トラフィック分析

本章では，まず WWW トラフィックの特徴について分析し，その結果を用いて「どの程度キャッシュ用に計算機資源を用意すれば，どの程度のヒット率が期待できるか」等，従来は明確な基準なしに進めていた WWW キャッシングプロキシーの設備計画方法について数量的な解析を加える．

2.1 WWW トラフィック特性

数理言語学の分野では，「Zipf の法則」と呼ばれる法則が知られている[7)]．この法則は，f を単語の使用度数，r を使用度数の大きい方から振った順位，C，k を定数としたときに，

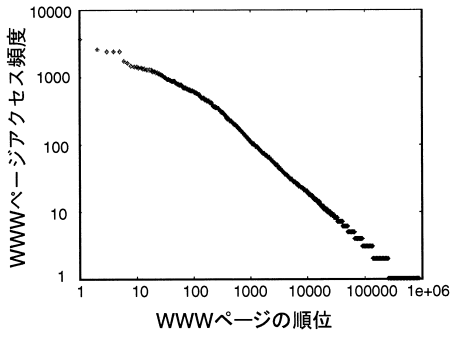


図 1 WWW のアクセス回数分布 (1)
Fig. 1 WWW access frequency distribution (1).

$$fr^k = C$$

が成り立つという経験則である．

興味深いことに，この法則が WWW ページのアクセス回数分布にもあてはまることが報告されている[8),12)]．**図 1** は，著者らが社内で利用している WWW プロキシーサーバにて記録したアクセス履歴 16 日分（約 225 万アクセス）を分析し，データごとのアクセス回数をアクセス回数順に示したものである．縦軸をデータごとのアクセス回数 f，横軸をそのデータのアクセス回数に基づく順位 r とした両対数グラフ上に，データが直線上に集まっているのが読み取れる．

ここで，アクセス回数の分布が単語の使用頻度と同様に「Zipf の法則」に従うことは，現象として興味深いだけでなく，キャッシュ・システム構築上でも重要であることに注意されたい．**図 2** は，同じアクセス履歴をもとに，WWW ページをアクセス回数別に分類したときに，アクセス回数別のデータが全体の何パーセントの割合を占めるかを示したものである．このうち，アクセス回数が 1 回で，まったくキャッシュすることに意味のないデータは 63%であり，95%のデータはキャッシュ効果の少ないアクセス回数 10 回未満のデータ☆である．

LRU アルゴリズムを用いたキャッシュ技術は，CPU のメモリーアクセス高速化等にその有効性が広く認識されているが，図 1 に示したように，メモリーアクセス高速化の前提となっているデータの局所参照性は，WWW ページの参照特性には見られない．

2.2 キャッシュ方式の性能比較

世の中に存在する WWW ページ関して定数 k，C

☆ 解析に用いたアクセス履歴は 16 日間の履歴であり，アクセス回数 10 回以下とは 1 日あたりのアクセス回数が 1 回に満たないことを意味する．

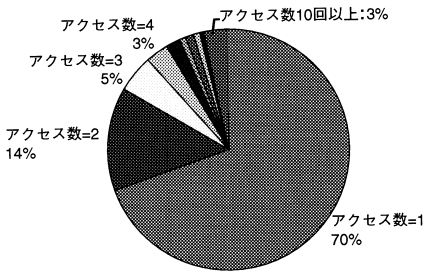


図 2 WWW のアクセス回数分布 (2)
Fig. 2 WWW access frequency distribution (2).

が分かれば，Zipf の法則により全 WWW ページの種類 R_t は出現回数 1 回の WWW ページの順位と推定できる．すなわち，

$$R_t = C^{\frac{1}{k}}$$

k はある程度の期間アクセス履歴を観測すれば計測できる．世の中すべての WWW ページの C の計測は困難であるが,「あるサイトが 1 週間の間に関係する全 WWW ページ」の C であれば，同じく履歴から計測できる．すなわち，観測から求めた k とアクセス総数 A を使い，

$$\begin{aligned} A &= \int_0^{R_t} Cr^{-k}dr \\ &= \frac{C^{\frac{1}{k}}}{1-k} \end{aligned}$$

を満たす C を求めればよい．この C の値の観測期間は，実際にキャッシュシステムを運用するときに何週間以内のデータに対してどの程度のヒット率を達成するか目標を定め，その目標に基づき設定すればよい．

これらの関係式と値を用いて積分計算を行えば，種々のキャッシュ方式とキャッシュ用記憶容量ごとに，その性能（キャッシュのヒット率）が推定できる．

たとえば，なんらかの方法でデータのアクセス回数を推定できれば，アクセス頻度上位のものだけキャッシュに蓄えることが可能となる（以下頻度方式と呼ぶ）．その場合のアクセス回数 r 位のデータがキャッシュに記憶されている確率は，アクセス回数 r 位までのデータがキャッシュ容量 S 未満であれば 1，以上であれば 0 となり，頻度方式（キャッシュ容量 S）のヒット率 $H_{Freq}(S)$ は，

$$\begin{aligned} H_{Freq}(S) &= \int_0^S P(r)dr \\ &= C^{\frac{k-1}{k}} S^{1-k} \end{aligned}$$

となる．また LRU アルゴリズムを用いたキャッシュ方式（以下 LRU 方式と呼ぶ）のヒット率 $H_{LRU}(S)$ は，アクセス回数 r 位のデータがキャッシュに記憶されている確率 $P(r)$ から推定できる．具体的には，キャッシュが一杯になる回数 D だけ，過去にそのデータが現れなかった確率 $(1-P(r))^D$ を考え，

$$\begin{aligned} &H_{LRU}(S) \\ &= \int_0^{R_t} P(r) \times (1-(1-P(r))^D)dr \\ &= \int_0^{R_t} P(r)dr - \int_0^{R_t} P(r) \times (1-P(r))^D dr \\ &= 1 - \int_0^{R_t} P(r) \times (1-P(r))^D dr \\ &= 1 - \frac{((1-k)C^{1-\frac{1}{k}})^{\frac{1}{k}}}{k} \\ &\quad \times \int_{(1-k)C^{-\frac{1}{k}}}^{\infty} t^{-\frac{1}{k}}(1-t)^{\frac{S}{1-k}} dt \end{aligned}$$

ここで，

$$\begin{aligned} D &= \frac{S}{1-k} \\ t &= P(r) \end{aligned}$$

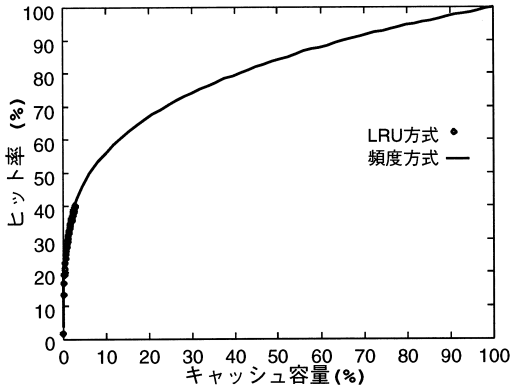
以上の考え方で，方式ごとのヒット率を推定し，グラフにまとめたものが**図 3** (a) である．図で横軸はキャッシュ容量が総データ量の何%であるかを，縦軸はキャッシュのヒット率を表している．ここで計算に必要な定数 k は前述のアクセス履歴より求めた値 0.75 を用いた．C はアクセス履歴の観測期間の長さにより変わるので，前述の履歴の値に近い 20000 を用いた．また，LRU 方式のヒット率を表す式は，第 2 項がいわゆる不完全ベータ関数の形をしており，これ以上式変形できなかったため Mathmatica の特殊関数ライブラリを使い，数値誤差上の問題が少ないヒット率 40%のところまで値を求めた．

図 3 (b) は図 3 (a) のキャッシュ容量 5%までを拡大したものであるが，これからから明らかなように，LRU 方式はキャッシュ容量が小さいときに頻度方式よりヒット率が下がるが，キャッシュ容量が総データ量の 5%以上であれば頻度方式を比較的良く近似している．

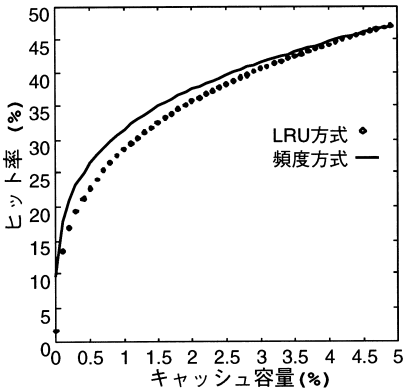
2.3 メモリ資源に関する考察

前節の結果を用いると「どの程度キャッシュ用に計算機資源を用意すれば，どの程度のヒット率が期待できるか」等が計算可能になり，従来は明確な基準なしに進めていた WWW キャッシングプロキシーの設備計画を，コストと効果を推定したうえで行うことが可能となる．

たとえば，頻度方式・LRU 方式ともにある容量以上



(a) ヒット率の変化



(b) キャッシュ容量5%までの拡大図

図3 方式ごとのヒット率の変化
Fig. 3 Comparison of hit rates.

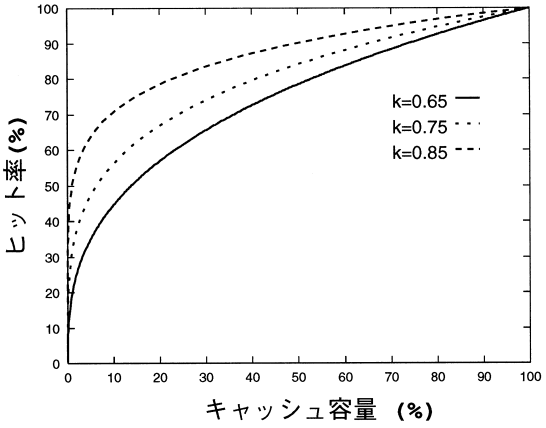


図4 k によるヒット率の変化
Fig. 4 Variation of hit rate by k.

表1 WWW キャッシュ用記憶容量
Table 1 Size of WWW cache storage.

回線速度	1週間の総データ量	全データ保持に必要な記憶容量	ヒット率 40%に必要な記憶容量
128 kbps	9.7 Gbyte	2.4 Gbyte	62 Mbyte
1.5 Mbps	113.4 Gbyte	28.4 Gbyte	725 Mbyte
45 Mbps	3.4 Tbyte	850 Gbyte	21.8 Gbyte
1 Gbps	75.6 Tbyte	18.9 Tbyte	484 Gbyte

にキャッシュ容量を増やしても，ヒット率はあまり増加しないことには注意を要する．**図4** に頻度方式を仮定して，k が変わった場合の記憶容量とヒット率の変化を示す．この図から，たとえば現有設備の主記憶容量から達成できるヒット率を求めることができる．また新規に設備更新を行う場合，どの程度の期間のデータを保持したいかから c を定めた後，記憶装置のコストと，キャッシュにより削減される回線使用料のつりあう点を求めれば，新規に導入すべき計算機の構成を，コストを客観的評価基準として求めることができる．

正確な議論には記憶装置のコストと回線使用料のデータが必要であるが，著者らは図から直観的に判断し，どの程度の期間のデータを保持したいかを決めた後は，その期間のデータ種類 R_t の 5〜10%程度（すなわち曲線の傾きからキャッシュの効果があまり向上しなくなるデータ容量）を目標にシステムに必要な記憶装置の容量を用意するのが得策と考えている．

さらに，頻度方式を前提として前述の式を使い，目標ヒット率と k およびある期間の総データ通信量（アクセス総数 A）から必要なキャッシュ容量を求めると，

$$CacheSize = (1-k)h^{\frac{1}{1-k}}A$$

となる．**表1** にサイトに必要な WWW キャッシュ用記憶容量を，$k = 0.75$・目標ヒット率 40%と仮定して，サイトに接続した回線速度ごとにまとめたものを示す．

表1 に従えば，キャッシングプロキシーに必要な記憶装置の容量は，実際に運用されているキャッシングプロキシーシステム[13]よりかなり小さい．これは，解析の間違いというより，現行のキャッシングプロキシーシステムの実装上の問題と考える．

具体的には初期に開発された WWW サーバでプロキシーとしても動作する CERN のサーバ[16]や，HARVEST[5]プロジェクトで開発された高効率キャッシングプロキシーシステムと，その系譜に属する多くの高効率プロキシー（たとえば，文献 17) や 18)）が，URL データに付随する内容の有効期限に関する情報を利用したキャッシュ管理アルゴリズム（いわゆる TTL に基づくキャッシュ置換え方式）を採用していることに問題があると思われる．

この方式は URL ごとに有効期限を指定し，有効期

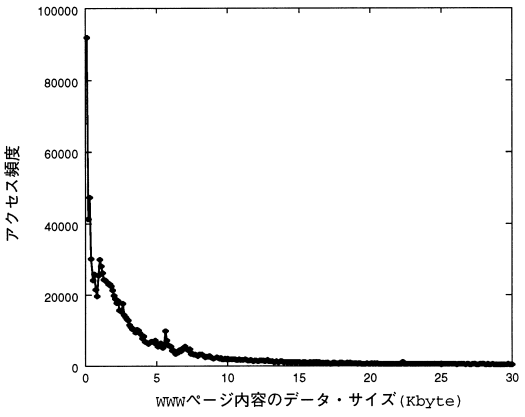


図 5 WWW ページ内容のデータサイズ分布
Fig. 5 Size of WWW pages.

限の長い URL を優先的にキャッシュに記憶する．しかし，一般に URL の有効期限はアクセス頻度とは独立であることから，アクセス頻度の低い URL までキャッシュ領域に記憶することになり，キャッシュ領域の不必要な増大と不必要なディスクアクセスによる性能低下をもたらしていると考えられる．

表 1 によれば目標ヒット率 40%の場合の頻度方式における必要記憶容量は，T1 回線（1.5 Mbps）においてパソコンの主記憶としても搭載可能な 1 Gbyte 以下であり，ハードデスクを必要としていない．現在の WWW プロキシーで主流の URL の有効期限を利用したキャッシュ管理は，データの再送を極力回避するために考案された管理方針であるが，WWW アクセスの統計的な性質を考慮すると，通常の LRU 方式に対して利点は少ないと考える．

2.4 ディスク I/O に関する考察

図 5 に前述のアクセス履歴より求めた WWW ページ内容のデータ・サイズについて調べた結果を示す．WWW ページ内容の平均データ・サイズは 10 Kbyte 弱であったが，これは極端にサイズの大きな少数の WWW ページに影響されている．実際には，データ・サイズ 5 Kbyte 以下のページが全体の 73%を占め，3 Kbyte 以下のページに絞っても全体の 60%を占めていた．

このことから，実際にキャッシングプロキシーのアーキテクチャ，特にディスク I/O に関する性能ネックを考える場合，WWW ページ内容の平均データサイズが小さいことを前提とした設計上の考察が重要と思われる．

表 2 に WWW ページ内容の平均データサイズを 3 Kbyte と仮定して，秒あたりのアクセス回数と，キャッシュ用記憶装置としてディスク装置を用いた場合に必要なディスク装置の数を示す．ここでディスク装置の台数は，1 台あたりのディスク装置がデータを入出力できる回数を，シークに必要な時間 8 ミリ秒とデータ転送に必要な時間 1~2 ミリ秒から，秒あたり 100 回（実効 I/O 性能 1 Mbyte/秒=平均 WWW ページサイズ 10 Kbyte∗100）であると推定して計算した．WWW ページ・アクセスのランダム性を考えると，WWW ページ 1 個につき 1 回シークが起きるという仮定は，OS の付加情報（ディレクトリ情報等）の I/O を無視しており楽観的すぎる．実際にはこの 2~3 倍の台数が必要になると予想される．

表 2 アクセス速度
Table 2 Access speed.

No.	回線速度	1 秒あたりのアクセス数	必要なディスク装置の台数
1.	128 kbps	5.3	1
2.	1.5 Mbps	62.5	1
3.	45 Mbps	1.9 K	19
4.	1 Gbps	41.7 K	417

表から明らかなように，T3 以上の回線速度（45 Mbps 以上）を持つサイトにおいては，必要なディスクの個数が多すぎる．実際には，回線速度が 5 Mbps を超えたあたりから，ディスク I/O のパフォーマンスによるボトルネックが発生すると考える．

3. 階層型分散キャッシュシステムの提案

前章のディスク I/O に関する考察から，回線速度が速くなるに従いディスク I/O の性能がパフォーマンスボトルネックとなることが分かる．特に大容量回線上に設置されるキャッシングプロキシーではディスク I/O のボトルネックを避けることが必須である．また，前章の記憶容量に関する考察から，頻度方式によればディスクではなく主記憶装置にキャッシュを記憶することは可能と考えられる．本章では上記考察に基づき，頻度方式と適切な分散方式と組み合わせ，主記憶をキャッシュストレージとして用いことでディスク I/O ボトルネックを解消したキャッシュシステムを提案する．

3.1 基本アイデア

本論文で提案する階層型分散キャッシュの特徴は，以下のとおりである：

頻度に基づくキャッシュコンテンツ選択： 提案方式の第 1 の特徴は，TTL ではなく頻度に基づくキャッシュコンテンツの選択である．将来高頻度でアクセスされるであろう URL のみを選択的にキャッシュする頻度方式の利点は先の議論のとおりである．し

かし，実際には将来の URL のアクセス数を正確に予測することは不可能である．そこで，我々は過去の URL アクセス頻度は将来の URL アクセス頻度を比較的よく近似していると考え☆，過去によくアクセスされたサイトの URL のみの選択的記憶と，頻度方式を比較的よく近似できる LRU キャッシュ置換え方式とを組み合わせた方式（以下疑似頻度方式と称す）を主たるキャッシュ戦略とした．具体的には，プロキシー上で観測されたアクセス履歴に基づき，過去一定期間におけるサイトのアクセス頻度を，将来のサイトのアクセス頻度として用いる．ここで，個々の URL ではなくサイト単位に集計を行うのは，データ量を削減するうえで重要である．

階層型分散キャッシュ構成： 提案方式の第 2 の特徴は，階層型分散キャッシュ構成である．大規模な組織におけるネットワーク構成は階層型を採用することが多く，複数のキャッシングプロキシーが階層型に使用される傾向がある．一般に，階層型キャッシュ構成では，上位層のキャッシュで高いヒット率を達成するのは困難である[5),8)]．これは，下位キャッシュの影響により，上位キャッシュに流れ込むトラフィックに含まれる重複が削減されるためである．提案手法で用いる頻度方式は，階層構成の各キャッシュごとに高アクセス頻度のサイトの URL（再利用される可能性の高い URL）を選択的にキャッシュする．このため，上位層のキャッシュでも従来方式と比較して高いヒット率を達成することが期待できる．また，主記憶をキャッシュストレージに用いるためには，複数のキャッシュ間で記憶領域を共有する必要がある．そこで，提案手法では，高アクセス頻度サイトの URL を，複数のキャッシュで分散共有することまで考慮した，分散キャッシュ構成も採り入れた．

自動キャッシュ分散制御： 提案手法の第 3 の特徴は，動的に変化するトラフィック特性に自動的に追従する自動キャッシュ分散制御である．WWW アクセスパターンは時々刻々変化するため，これに応じてキャッシュ対象とするサイトを定期的に更新（チューニング）する必要があるが，これを人手で行うのは多大な労力を要する．提案手法では，このチューニングの自動化方法として，プロキシーのアクセス履歴を定期的に解析し，アクセス頻度上位サイトの情報を定期的に更新する機能を採り入れた．このとき，キャッシュ対象サイト変更のヒット率への影響を抑えるため，分散キャッシュ間では各キャッシュの受け持ちサイトの移動が最小限になるようにアクセス頻度上位サイトの情報を更新する方式とした．

☆ ログの定性的な観察に基づく．4 章の実験結果はその考えが正しいことをサポートしていると考える．

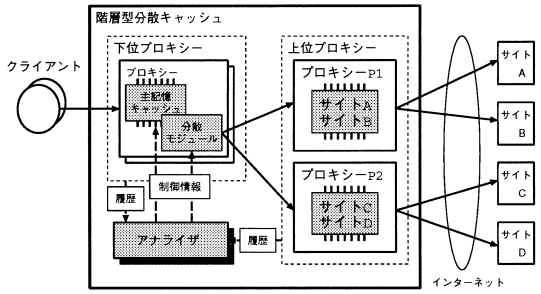


図 6 分散キャッシュの構成
Fig. 6 System configuration of distributed caching proxies.

3.2 システム構成

図 6 に本論文で提案する階層型分散キャッシュシステムの構成を示す．提案システムは，2 階層の階層型キャッシュ構成であり，各キャッシュはそれぞれ疑似頻度方式を用いる．上位プロキシーは記憶領域の共有と負荷分散を目的として分散キャッシュ構成とし，さらに上位プロキシーへの既存プロキシーの使用も考慮して，下位プロキシーからのリクエストを制御する分散方式とした．また，個々のプロキシーの設定を一元的に行えるようにすることは，システムの運用上重要であると考え，アナライザによる一元管理を行う構成とした．キャッシュシステム全体のコンフィグレーションは，アナライザが生成する制御情報により行う．

アナライザ アナライザは，各プロキシーからアクセス履歴を収集し，分散キャッシュ制御に必要な制御情報を生成する．アナライザが生成する制御情報は，個々のプロキシーにおけるキャッシュ対象を指定する選択制御情報（高アクセス頻度のサイトを記述）と，選択制御情報にさらに分散情報（高アクセス頻度のサイトをどのプロキシーでキャッシュするか）を付加した分散制御情報の 2 種類である．また，WWW トラフィックのアクセスパターンは日々変化するため，上記制御情報も定期的に再調整する必要がある．アナライザは，プロキシー上で観測されるアクセス履歴に基づき，高アクセス頻度のサイトの抽出を行い選択制御情報を生成する．提案システムは分散キャッシュ構成のため，さらにアクセス履歴に含まれる頻度情報および前回生成した分散制御情報値を参照して，以下のポリシーに基づき分散制御情報を生成する：

(1) 各プロキシーごとの過去のアクセス数を等し

```
/* ログの取得及びソート */
各キャッシュよりログを取得;
サイト毎にアクセス頻度を集計，ソート;

/* 初期化 */
serverの負荷及び記憶サイトの初期化;

/* 既割当サイトの再割り付け */
for 上位N%サイト* {
  server = サイトが割り当てられているキャッシュ;
  if server != null {
    server.記憶サイト.append(サイト);
    server.負荷 += サイト.アクセス数;
  }
}

/* 新規サイトの割り付け */
for 上位N%サイトのうち割り付けられなかったサイト* {
  server = 最小負荷のサーバ;
  server.記憶サイト.append(サイト);
  server.負荷 += サイト.アクセス数;
}
                    *:高アクセス頻度にソートされている
```

図7 分散制御情報生成アルゴリズム
Fig. 7 Algorithm for generating control information.

くすることにより各上位プロキシーの負荷を均等に制御，

(2) あるプロキシーにすでにサイトが割り当てられている場合，当該サイトは同一プロキシーに割り当てる．

具体的には，まず各上位プロキシーよりアクセス履歴を取得，サイトごとにアクセス頻度を集計・ソートし，アクセス頻度上位のサイトを抽出する．その後，前回上位プロキシーに割り当てられていたサイトかつアクセス頻度上位サイトであるサイトを同一の上位プロキシーに割り当て，新規にアクセス頻度上位となったサイトを，各上位プロキシーのアクセス数が均等になるように割り当てる．分散制御情報生成手順を**図7**に示す．

アナライザは，分散キャッシュシステムに1台存在すればよい．そのためアナライザの実装は，プロキシーとは独立のプログラム構成とし，各プロキシーからの履歴の取得および各プロキシー（下位プロキシーのみ）への制御情報の転送にはftpを用いる構成とした．

また，我々は文献8)で制御情報生成の異なる手法を示している．文献8)で示された手法は，より複雑なネットワーク構成への適用が可能であるが，本方式の第2の特徴は有していない．

分散モジュール 分散モジュールは，アナライザが生成した分散制御情報に基づき，ブラウザからのリクエストを適切な上位プロキシーに転送する．図6の例では，プロキシーP1はサイトAおよびサイトBを，プロキシーP2はサイトCおよびサイトDをそれぞれ担当し，たとえばブラウザがサイトAのURLを要求した場合は，上位プロキシーP1に当該リクエストを転送する☆．リクエストが高アクセス頻度でないサイトのURLの場合は，プロキシーで定義された転送先にリクエストを転送する．また，分散モジュールは，定期的に制御情報が更新されたかどうかを確認し，更新されていた場合は制御情報を再ロードする．

上位・下位プロキシー 上位・下位プロキシーともキャッシュストレージとして主記憶を，キャッシュ戦略として疑似頻度方式を採用したキャッシングプロキシーであり，高アクセスサイトのURLの選択的なキャッシュと，LRUによるキャッシュ置換えを行う．上位プロキシーは分散キャッシュを構成し，また，下位プロキシーには分散モジュールが存在し，上位プロキシー間の分散制御を行う．この結果，上位プロキシー間での記憶領域の共有と負荷分散を達成しつつ，頻度方式によるキャッシュ制御が可能となっている．

4. 実験結果

我々は，頻度方式の実装手段として提案した疑似頻度方式の効果を確認することを目的として，階層型ネットワークを用いたシミュレーションを行った．TTLに基づくキャッシュ置換え方式と比較した頻度方式の優位性は2章での議論より明らかであり，本章ではTTLに基づくキャッシュ置換え方式より効果のあるLRUキャッシュ置換え方式との比較を行った．

4.1 実験条件

本シミュレーションでは，大きな組織における階層型キャッシュシステムを仮定し，図6と同様の上位・下位キャッシュを使用した2階層型とした．下位プロキシーは各部門ごとに稼動しているキャッシングプロキシーを，上位プロキシーはファイアウォール上で稼動しているキャッシングプロキシーをそれぞれ表す．具体的には13台のクライアントを持つ3つの部門を想定し，3台の下位プロキシーと部門数+1台（4台）の上位プロキシー構成とした．また，実験に用いたアクセス履歴には，39個のリクエスト元が含まれている．WWWトラフィックを再現するために，これらのリクエスト元を，総リクエスト数が均等になるように各部門に分け，それらを各部門のクライアントのリクエストとした．実験には先の分析で使用したアクセス履歴のうち，10日分を使用した．

☆ 下位プロキシーに当該URLがキャッシュされていない場合．

本シミュレーションでは (1) 提案手法，および (2) ラウンドロビン型分散手法をそれぞれ階層型キャッシュシステムに適用し，ヒット率およびトラフィック削減効果を測定した．

提案手法の詳細　シミュレーションにおけるキャッシュ容量の選定においては，本階層型分散キャッシュのキャッシュストレージが主記憶であることを想定し，一般の PC に搭載可能な主記憶容量 64 Mbyte を個々のプロキシーのキャッシュ容量とした．また，クライアントキャッシュサイズは，一般のクライアント PC で使用されるであろうキャッシュ容量を想定して，8 Mbyte とした．各キャッシュにおける戦略は，クライアントキャッシュでは LRU を，下位・上位プロキシーのキャッシュは疑似頻度方式とした．上位プロキシーは分散キャッシュ構成とした．

下位プロキシーは，個々の下位プロキシーで観測されたアクセス履歴をもとに，アクセス頻度上位 10%のサイトの URL（すなわち，各部門ごとのアクセス上位 10%のサイトの URL）を選択的にキャッシュする．

また上位プロキシーの分散制御は，全上位プロキシーにおいて観測したアクセス履歴に基づき，アクセス頻度上位 10%のサイトの URL へのリクエストを，負荷が均等になるように上位プロキシー 4 台のうち 3 台に，それ以外のリクエスト（低アクセス頻度サイトへのリクエスト）を残りのもう 1 台のプロキシーに転送する．ここで，低アクセス頻度サイトへのリクエストを受け持つプロキシーはキャッシュを持たないが，これは当該リクエストは低アクセス頻度のサイトの URL のみであり，キャッシュしてもあまり効果がないと考えたためである．また，トラフィックの変動に追従するため，制御情報は過去 7 日間のアクセス履歴をもとに毎日計算する．なお，初日はアクセス情報が利用できないため，クライアントキャッシュのみが機能すると想定してシミュレーションを行った．

ラウンドロビン方式の詳細　従来のキャッシュ戦略と比較するため，我々は，ラウンドロビン方式による分散キャッシュのヒット率およびトラフィック削減率についても測定した．キャッシュ構成は先の頻度に基づく方式と同様，8 Mbyte のクライアントキャッシュおよび 64 Mbyte の上位・下位プロキシーキャッシュを用いている．ただし，本方式の場合は，上位プロキシーはすべて 64 MByte のキャッシュを持つ．キャッシュ戦略は，LRU によるキャッシュ置換えアルゴリズムのみであり，先の提案方式の場合と異なりすべての URL をキャッシュする．また，上位プロキシーへの分散は，下位プロキシーから上位プロキシーを順次呼び出すラウンドロビン方式とした．本ラウンドロビン方式で共有 LRU メモリを実現するのは困難であり，本シミュレーションでは上位レベルのキャッシュは共有されないと仮定した．

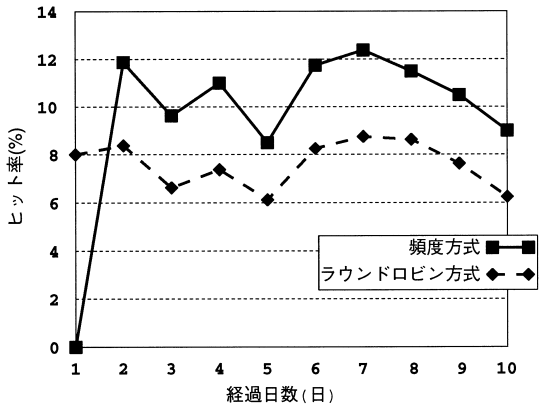


図 8　下位プロキシーキャッシュのヒット率
Fig. 8　Hit rates of lower level proxies.

4.2 実験結果

10 日間の期間に観測されたクライアントキャッシュのヒット率は，平均 16.3%であった．

次に，下位プロキシーキャッシュのヒット率の推移を**図 8** に示す．図に示すように，本方式によるヒット率（平均 10.6%）は，ラウンドロビン分散方式によるヒット率（平均 7.9%）より高い．これは，提案手法である疑似頻度方式の効果であると考えられる．

図 9 に上位プロキシーキャッシュにおけるヒット率の推移を示す．提案方式のヒット率（平均 8%）と比較して，ラウンドロビン方式のヒット率（平均 1.9%）は著しく低下している．この図は，提案方式の優位点を明らかに示していると考えられる．提案方式の初日のヒット率が 0%であるのは，初日は分散制御情報が存在せず，すべてのリクエストがキャッシュを持たないプロキシー P4 に転送されたためである．しかし，この現象はキャッシングプロキシーの定常運用では生じないため，特に問題にならないと考える．

図 10 は，階層型キャッシュシステム全体のヒット率の推移を示している．提案方式，およびラウンドロビン方式の平均ヒット率はそれぞれ 30.4%および 25.4%である．

以上の結果から，以下のことが結論付けられる：

(1)　WWW トラフィックの特徴に基づけば，現在キャッシュ方式の前提とされているハードディスクのような低速・大容量の記憶装置を使用しない，主

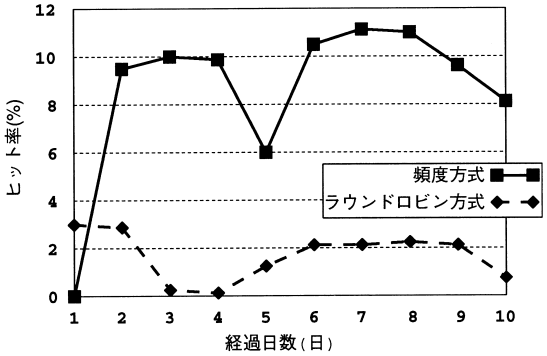


図 9 上位プロキシーキャッシュのヒット率
Fig. 9 Hit rates of upper level proxies.

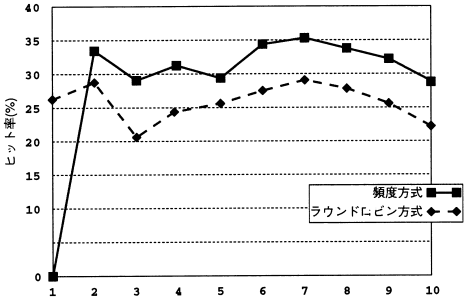


図 10 トラフィック削減率
Fig. 10 Rate of traffic reduction.

記憶のみで構成した分散キャッシュシステムが構成可能である．

ここで，比較の対象となっている LRU 方式のヒット率も，従来より広く用いられている TTL に基づくキャッシュ方式のヒット率より高いことに注意されたい．現状主として使われている TTL に基づくキャッシュ方式では，同程度のヒット率達成のため，1 桁程度上の記憶容量を必要としており[13]，低速なハードディスクをキャッシュ領域に使用している．

(2) 頻度に基づくキャッシュコンテンツ選択方式により，下位プロキシーキャッシュのヒット率は 2.7%改善された（7.9%から 10.6%へ）．この結果は，我々が提案した高アクセス頻度サイトの URL の選択的なキャッシュが，1 階層キャッシュ構造の場合にも効果があることを示している．

(3) 頻度に基づくキャッシュ分散方式により，上位プロキシーキャッシュのヒット率は 6.1%改善された（1.9%から 8.0%へ）．階層型キャッシュにおいて上位プロキシーへ流れ込むトラフィックは，下位プロキシーの影響により冗長性が非常に小さいという性質を持つ．上位プロキシーで明確なヒット率の差が現れたことは，我々が提案した頻度方式が，このような性質を持つトラフィック（特に階層型の上位キャッシュの戦略）に対して効果的な手法であることを示していると考える．

5. 考　　察

5.1 実験結果に関する考察と類似研究との比較

我々が提案した頻度方式と，従来から研究・利用されているキャッシュ方式[1)~4),16)~18)] との差異は，次の点である．すなわち，従来の手法がすべての URL を記憶し，キャッシュが満杯になった後どの URL を置き換えるかというキャッシュ置換え方式に着目しているのに対し，頻度方式は，どの URL を記憶し，どの URL を無視すべきかに着目している点である．具体的には頻度方式では，高アクセス頻度の URL のみを記憶することを提案している．

2 章で述べたように WWW トラフィックの特性は Zipf の法則に従う．これは，WWW トラフィックが CPU のメモリアクセス特性と比較して，一定リクエスト数内に非常に多くの種類のデータを含み，かつこれらの多くが 1 度しか利用されない，再利用可能性の低いデータであることを意味している．

我々は，上記の WWW トラフィックの特性およびキャッシュ方式の動作特性から，前章の実験における頻度方式の優位性の原因を「記憶領域の利用効率の差によるもの」と考えた．すなわち，キャッシュ置換え方式は，URL の再利用可能性とは無関係にすべての URL を記憶する．この結果，再利用可能性の低い URL が記憶領域の多くを占有し，無駄な記憶領域の消費が発生していると推察できる．一方，頻度方式では，再利用可能性の低い URL は最初から記憶しない．このため，キャッシュには再利用可能性の高い URL のみが存在することになり，記憶領域の有効活用が可能になっていると考えられる．特に階層型キャッシュの上位層におけるトラフィックは冗長性が削減されており再利用可能性の低いデータがより多数含まれている．したがって上位層ほど記憶領域の利用効率に差が生じ，これが前章の実験におけるヒット率の差の原因と考えられる．

キャッシュ記憶領域が有効活用されているかどうかは，キャッシュ中で再利用されなかった URL の数よりある程度推測できる．我々は，実験で用いたアクセス履歴 1 週間分のデータを用いて，頻度方式と LRU によるキャッシュ置換え方式のそれぞれについて，毎日の最後のアクセスが終了した時点で，キャッシュ中に存在するが一度も利用されなかった URL 数の比較を行った．具体的には，1 階層キャッシュ（キャッシュ容

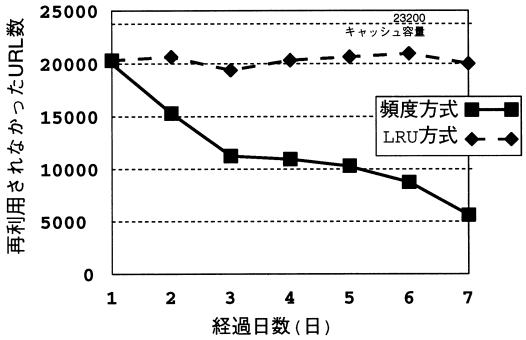


図 11 キャッシュ利用効率の比較
Fig. 11 Comparison for cache usage.

量は 1 週間の出現 URL 数の 1/10；23,200 個の URL を記憶可能）構成において頻度方式（過去 1 週間分のアクセス履歴におけるアクセス上位 10%の URL のみを選択的にキャッシュ）と LRU によるキャッシュ置換え方式について比較を行った．この結果を**図 11** に示す．

図 11 に示すように，記憶されたが一度も再利用されなかった URL 数は頻度方式では 1 日あたり平均約 11,000，最小 5,000（キャッシュ容量比それぞれ約 50%，20%）であるのに対し，LRU 方式は約 20,000（キャッシュ容量費約 80%）であり非常に多い．この結果からも，頻度方式は LRU 方式と比較してキャッシュの有効利用が可能であり，これがヒット率の差の主な要因と推察できる．

上記の議論に基づけば，LRU を基に改善されたキャッシュ置換え方式の効果についてある程度推察することが可能である．

たとえば，文献 1) では，LRU をもとに，キャッシュ置換え回数が最小になるように LRU リストの末尾に近くかつサイズの大きな URL を削除する方法，およびある閾値以上のサイズを有する URL をキャッシュしない方法が効果があることを示している．これらの方法は，キャッシュ容量（サイズ）固定の際にできるだけ多くの URL を記憶する効果があり，これがヒット率改善の原因と考えられる．

また，文献 2) では，LRU チェーンを世代ごとに分割し，URL をキャッシュに挿入する地点およびアクセス時に移動させる地点を LRU リストの先頭ではなく，URL のアクセス回数に応じた世代（中間点）に挿入する．この方法は，キャッシュが満杯になった場合に LRU 方式と比較してアクセス回数の低い URL をより優先的に破棄できる．この結果記憶領域をより有効利用することが可能であり，これがヒット率の改善に貢献しているものと推察できる．

5.2 分散方式に関する考察

ICP（Internet Caching Protocol）に基づく分散キャッシング方式は複数キャッシングプロキシー間でキャッシュコンテンツを共有させる方式として広く利用されている．この方式によるプロキシーは，クライアントの要求した WWW ページが自分のキャッシュにない場合，他のプロキシーにキャッシュされていないか，ICP というプロトコルで確認し，もし他のプロキシーに WWW ページがキャッシュされていれば，そちらから WWW ページを取得する．

しかし，WWW トラフィックが Zipf の法則に従うことを考慮すると，ICP による分散キャッシュ方式が効果的であるとは考え難い．Zipf の法則によれば，ごく少数のページのみが高頻度でアクセスされる反面，ほとんど再アクセスされないページが非常に多く存在する．このことは，もし個々のプロキシー間でユーザグループの持つ興味が異なっている場合，個々のプロキシーのキャッシュコンテンツが著しく異なることを意味する．この結果として ICP 方式により他のプロキシーから WWW データを取得できる確率は低くなり，ICP の持つプロトコル・オーバヘッドによりシステムのレスポンスが遅くなることが予想される．

実際に運用されているキャッシュ・サイトの実績でも，http によるヒット率に比べ，ICP によるヒット率は低いものが多く（たとえば文献 13)），以上の考察を裏付けている．

分散方式の他の手法として注目されている CARP [19] は，URL のハッシュ値をもとに個々の URL を記憶するプロキシーを決定することにより記憶領域の共有と負荷分散を達成する．本方式は，先の ICP と比較して，ある 1 つの URL は 1 つのプロキシーにしか記憶されないため記憶領域を有効利用できる，URL の記憶先は決まっているために無駄なメッセージ交換が必要ないという利点を持つ．本方式は複数のキャッシュを擬似的に 1 つのキャッシュとして見せることが可能な点において優れているが，WWW トラフィックが Zipf の法則に従うことを考慮すると適切なキャッシュ戦略と組み合わせないと十分効果を発揮できないと考えられる．我々が提案した頻度に基づくキャッシュコンテンツ選択は，この URL ハッシュによる分散方式と組み合わせることも可能である．先の実験における下位プロキシーのヒット率の結果から，提案方式によりヒット率の改善が可能であると推察されるが，より詳細な議論は今後の課題であると考える．

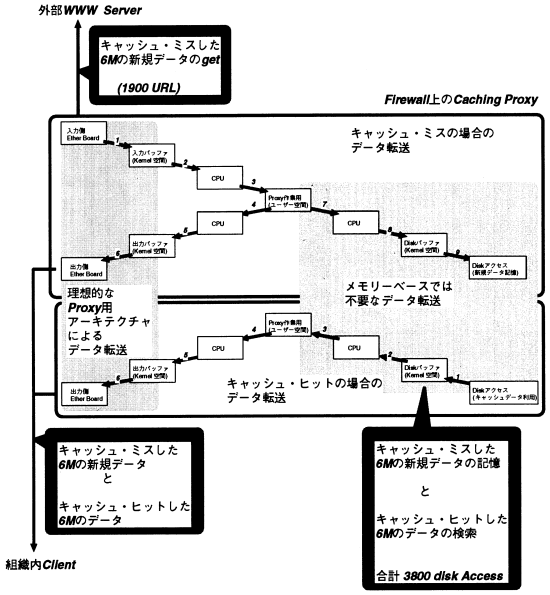


図 12 T3 回線で外とつながった FireWall 上のプロキシー
Fig. 12 Caching proxy on a firewall connected with T3 line.

5.3 計算機アーキテクチャに関する考察

firewall 上で動作しているプロキシー ソフトの I/O を考えた場合，外部への回線速度（たとえば 6 Mbps）に対して 15 倍の負荷が計算機のバスにかかることは注意を要する（**図 12**）．計算を簡単にするため，キャッシュのヒット率を 50%と仮定する．この場合，内部へは外部から取ってきた WWW ページ（6 Mbit）と，キャッシュ内部から取り出した WWW ページ（6 Mbit）をサービスできるが，前者は計算機内部の転送が 9 回（図 12 上半分），後者は 6 回（図 12 下半分）必要であり，合計 15 回分の転送が計算機内部で発生する（実際には実装方法によりオーバヘッドが発生するので，上記は内輪の見積りであることに注意されたい）．

現在パソコンベースのプロキシーシステムが多く使われているが，Fast Page Mode を使ったシステム（カタログ値 132 Mbyte/秒，実効 100 Mbyte/秒程度）を考えた場合，上記の議論を前提に考えれば，45 Mbps の T3 回線が性能上の限界となっている．実際にはこの半分程度の速度で限界に達するか，アクセス速度の早い EDO DRAM/SDRAM 等が必要になる．

メモリベースのプロキシーを作れば，ディスクアクセスに必要な図 12 右の斜線部に相当するデータ転送が不要となり，バスへの負荷は回線速度の 9 倍程度となる．また，一部のパソコンベースに作られたルータに見られるような，直接イーサーボード間，ボード・メモリ間のデータ転送を，ユーザプログラムが制御できるようなシステムソフト（理想的には回線速度の 3 倍のバスへの負荷：図 12 左の斜線部）の整備はこのような状況の改善案といえる．

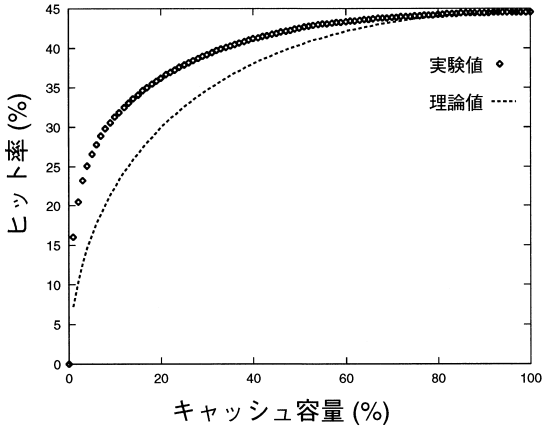


図 13 LRU 方式と頻度方式のヒット率の差異
Fig. 13 Hit rates for LRU and frequncy-based approach.

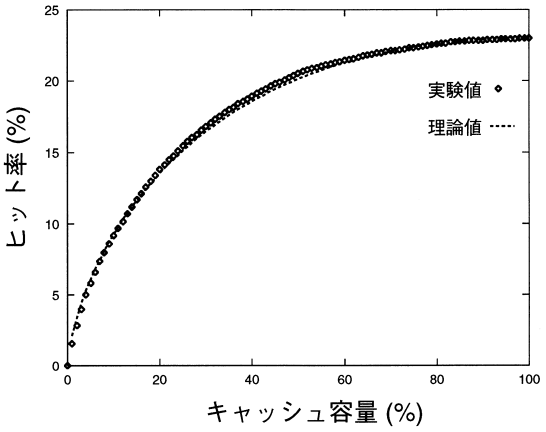


図 14 重複を取り除いた場合のヒット率
Fig. 14 Hit rates without duplication.

5.4 モデリング手法に関する考察

Almeida らは，Zipf の法則で予想されるヒット率と LRU によるヒット率を比較し，Zipf の法則ではヒット率を過小評価してしまうことを報告[10] し，この現象を取り扱うには別のモデリング手法[11]がよいことを報告している．この現象は，我々のデータにおいても確認されている（**図 13**）．

我々は，この原因をローカルキャッシュを持たないユーザによるもの，すなわち，1 人のユーザは短時間で何度も同一ページを行き来する傾向があり，これが前述のアクセス履歴に含まれた結果，LRU 方式のヒット率が向上するもの，と考えた．実際，前述のアクセス履歴から同一クライアントが同じ日付で同一ペー

ジにアクセスした履歴を取り除いた場合，シミュレーションの結果は Zipf の法則と一致する（**図 14**）．

以上の結果は，キャッシュの運用形態や WWW の使用形態が変われば，必要なモデル化技法が変化することを示唆している．今後，急速に変化する状況に即したモデル化技術の開発が重要性を増すと考える．

また，このようなモデル化手法の精緻化は 2.3 節の議論にも影響を与えるが，本研究でベースとした Zipf の法則に基づくモデリングは，メモリ資源を過剰に見積もる傾向にあり，必要な資源を過小に評価する危険が少ない．このことから，3 章で提案した主記憶ベースの分散キャッシュシステムの妥当性は崩れないと考える．

6. ま と め

WWW トラフィックの特徴について分析し，そのアクセスパターンに Zipf の法則が成立するという知見に基づき，種々のキャッシュ方式の性能を評価した．この結果，現在キャッシュ方式の前提とされているハードディスクのような低速・大容量の記憶装置の使用が適切でないことを指摘し，少数のアクセス回数の高いデータを選択的にキャッシュする頻度方式の妥当性について述べた．この結果，従来は明確な基準なしに進めていた WWW キャッシングプロキシーの設備計画を，コストと効果を推定したうえで行うことが可能となった．

また，キャッシュ領域を高速であるが比較的小容量の主記憶のみで構成した分散キャッシュシステムが頻度方式に基づき設計可能であることを示した．さらに，シミュレーションにより外部へのトラフィックを LRU キャッシュ置換え方式の約 1.2 倍削減できることを確認した．

以上報告した本研究では WWW プロキシー squid [17] で取得したログデータをベースに分析と評価実験を行った．squid [17] で取得したログデータには WWW ページの更新頻度の情報が記憶されておらず，本研究では WWW ページの更新頻度の影響が考慮できていない．また，5.4 節で報告したようなモデリング手法に関する課題も残されている．さらに詳細なデータの収集とモデルの改良・アーキテクチャへのフィードバックが今後に残された重要な研究課題である．

参 考 文 献

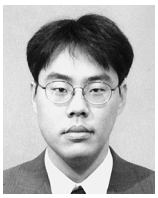
1) Abrams, M., Standridge, C.R., Abdulla, G., Williams, S. and Fox, E.A.: Caching Proxies: Limitations and Potentials, *4th International World Wide Web Conference* (1995).
2) 大澤，早野，弓場，箱崎：WWW サーバのログに基づいたキャッシュ置換えアルゴリズムの評価，マルチメディアと分散処理/グループウェア研究会，pp.191–196 (1996).
3) Wooster, R.P. and Abrams, M.: Proxy Caching that Estimates Page Load Delays, *6th International World Wide Web Conference*, pp.325–334 (1997).
4) Scheuermann, P., Shim, J. and Vingralek, R.: A Case for Delay-Conscious Caching of Web Documents, *6th International World Wide Web Conference*, pp.725–734 (1997).
5) Chankhunthod, A., Danzig, P.B., Neerdaels, C., Schwarts, M.F. and Worrll, K.J.:A Hierarchical Internet Object Cache, *USENIX96*, pp.153–163 (1996).
6) Gwertzman, J. and Seltzer, M.: The Case for Geographical Push-Caching, *HotOSS Conference*. ftp://das-ftp.harvard.edu/trchreports/tr-34-94.ps.gz (1994).
7) 水谷：数理言語学，培風館 (1982).
8) 吉田健一：WWW 用分散キャッシュ構成の検討，インターネットコンファレンス*'96*, pp.59–68 (1996).
9) Internet Caching Protocol (ICP), Vresion 2. http://ds.internic.net/rfc/rfc2186.txt (1997).
10) Almeida, V., Bestavros, A., Crovella, M. and de Oliveira, A.: Characterizing Reference Locality in the WWW, *Proc. PDIS'96, The IEEE Conf. on Parallel and Distributed Information Systems* (1996).
11) Mattson, R., Gecsei, J., Slutz, D. and Traiger, I.: Evaluation Techniques and Storage Hierarchies, *IBM System Journal, 9*, pp.78–117 (1970).
12) Glassman, S.: A caching relay for the World Wide Web, *Proc. 1st International Conference on the World Wide Web* (1994).
13) 1997 年度 WIDE プロジェクト研究報告書 (1998).
14) 知念，山口：先読みによる www アクセスの高速化の可能性，インターネットコンファレンス*'96*, pp.53–58 (1996).
15) 土居：能動的キャッシュ制御を持った proxy サーバーによる www アクセスの高速化，インターネットコンファレンス*'96*, pp.45–52 (1996).
16) http://www.w3.org/pub/www/daemon (1997).
17) http://squid.nlanr.net/squid/ (1997).
18) Microsoft: Microsoft proxy server. http://www.microsoft.com/proxy/default.asp (1997).
19) Valloppillil, V. and Ross, K.W.: Cache Array

Routing Protocol v1.0, Internet-Draft. http://ircache.nlanr.net/Cache/ICP/draft-vinod-carp-v1-03.txt (1998).
20) 佐藤：ETL Delegate Home Page. http://wall.etl.go.jp/delegate/ (1994).

(平成10年12月14日受付)
(平成12年 7 月 5 日採録)

西川 記史（正会員）

昭和41年生．平成3年神戸大学大学院工学研究科計測工学専攻修士課程修了．同年（株）日立製作所入社．システム開発研究所にてネットワーク関連アプリケーションおよびデータベース管理システムの研究・開発に従事．平成11年よりソフトウェア事業部．1998年度情報処理学会山下記念賞受賞．

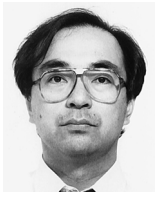


細川 貴史（正会員）

昭和47年生．平成8年京都大学大学院工学研究科修了．同年（株）日立製作所入社．システム開発研究所にてネットワーク指向サービスアプリケーションの研究に従事．平成12年郵政省入省．電気通信局にてインターネットを活用した教育の情報化施策に従事．

森 靖英（正会員）

昭和41年生．平成1年京都大学理学部卒業．平成3年東京大学大学院修士課程（物理）修了．同年（株）日立製作所入社．平成10年度より，技術研究組合新情報処理開発機構（RWCP）出向．神経回路網，パターン認識の研究に従事．日本物理学会，日本神経回路学会，電子情報通信学会各会員．



吉田 健一（正会員）

昭和32年生．昭和55年東京工業大学情報科学科卒業．同年,（株）日立製作所に入社．同社エネルギー研究所，1986年より基礎研究所，1998年よりシステム開発研究所．博士（工学）．1984年日本原子力学会論文賞，1990年電気学会論文賞，1992年人工知能学会論文賞，1991，1996年人工知能学会全国大会優秀論文賞，1995年度人工知能学会研究奨励賞受賞．AAAI，ACM，各会員．

辻 洋（正会員）

昭和28年生．昭和53年（株）日立製作所入社．システム開発研究所にて，意思決定支援システム，知識ベースシステム，グループウェアシステムの研究・開発に従事．この間，カーネギメロン大学客員研究員，システム制御情報学会編集委員，電子協専門委員など歴任．現在，情報処理学会編集委員．博士（工学）．技術士（情報処理部門）．