WWW トラフィック分析と分散キャッシュ

西川記史†，細川貴史†，辻洋†，森靖英‡，吉田健一‡

†日立製作所 システム開発研究所
‡日立製作所 基礎研究所

概要

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．中でも，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，十分な効果を上げられない．本研究では，始めに WWW トラフィックの特徴について分析し，次に分析結果にもとづき望ましいキャッシュ技術について述べる．

WWW Trafic Analysis and Distributed Cache System

N.Nishikawa†, T.Hosokawa†, H.Tsuji†, Y.Mori‡, K.Yoshida‡

†System Development Laboratory, Hitachi, Ltd.
‡Advanced Research Laboratory, Hitachi, Ltd.

Abstract

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 network. Thus, the demand for better tuning methods in wide area networks 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.

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．中でも，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．例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までに急増しており，需要の増加に応じる為に回線容量を増加しても，その翌日から急速にWWWの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事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．キャッシュ技術自身の研究も階層型キャッシュ [1] や分散型キャッシュ，Internet Cache Protocol (ICP) 等盛んに行われている．

しかし，上記研究事例には，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ものが多く，十分な効果を上げていない．本研究では，始めにWWWトラフィックの特徴について分析し，次に分析結果にもとづき望ましいキャッシュ技術について述べる．

2 WWWトラフィックの特徴分析

2.1 Zipfの法則

数理言語学の分野では，いわゆる「Zipfの法則」と呼ばれる法則が知られている [3]．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 を定数とした時に

$$fr^k = C$$

がなりたつという経験則である．

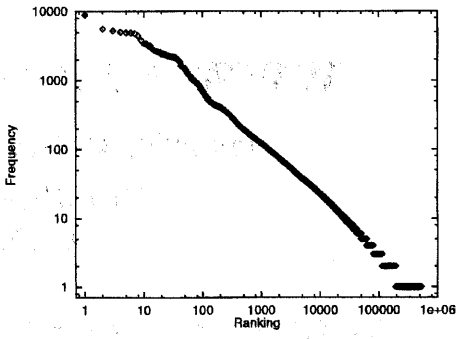

図 1: WWW のアクセス回数分布 (1)

興味深い事に，この法則はWWWデータのアクセス回数分布にもあてはまる．図1は，WWW代理サーバーにて記録したアクセスログ16日分 (約225万アクセス) を分析し，データ毎のアクセス回数をアクセス回数順に示したものである．縦軸をデータ毎のアクセス回数 f，横軸をそのデータのアクセス回数にもとづく順位 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，同じアクセスログをもとに，WWWデータをアクセス回数別に分類した時に，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，全くキャッシュする事に意味の無いデータは63%であり，95%のデータはキャッシュ効果の少ないア

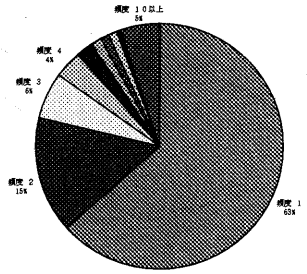

図 2: WWW のアクセス回数分布 (2)

クセス回数10回未満のデータ[1]である.

LRU(Least Recently Used) 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2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 データ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 URL(Universal Resource Locator) データ[2]に関して定数 k,C がわかれば, Zipf の法則により全 URL データの種類 R_t は出現回数1回のデータ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ログを観測すれば計測できる. 世の中全ての URL データの 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URL」の C であれば, 同じくログから計測できる. すなわち, 観測から求めた k とアクセス総数 A を使い,

$$A = \int_0^{R_t} Cr^{-k}dr$$

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

をみたす C を求めれば良い.

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毎に, その性能 (キャッシュのヒット率) が推定できる.

例えばキャッシュが溢れた時にランダムに古いデータを捨てて空き領域を作るようなキャッシュ方式 (以下ランダム方式と呼ぶ) では, 次にアクセス回数 r 位のデータがくる確率 $P(r)$ は,

$$P(r) = \frac{\text{r 位のデータのアクセス回数}}{\text{アクセス総数 A}}$$

$$= (1-k) \times C^{1-\frac{1}{k}} \times r^{-k}$$

また, そのデータが容量 S のキャッシュ (S 個の URL データを記憶できるキャッシュ) に記憶されている確率 P_{Rand} は,

$$P_{Rand} = \frac{\text{キャッシュ容量 S}}{\text{全データ種類} R_t}$$

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

となり, ヒット率は両者の積を積分した値で推定できる. P_{Rand} は r によらず一定であり $\int P(r)dr = 1$ であるので, 結局ランダム方式 (キャッシュ容量 S) のヒット率 $H_{Rand}(S)$ は P_{Rand} そのものとなる. すなわち,

$$H_{Rand}(S) = \frac{S}{C^{\frac{1}{k}}}$$

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,

$$H_{LRU}(S) = \int_0^{R_t} P(r) \times (1-(1-P(r))^D)dr$$

$$= 1 - \int_0^{R_t} P(r) \times (1-P(r))^D dr$$

[1]解析に用いたアクセスログは16日間のログであり, アクセス回数10回以下とは1日あたりのアクセスが1回に満たない事を意味する.

[2]WWW で提供されるデータ

$$= 1 - \frac{((1-k)C^{1-\frac{1}{k}})^{\frac{1}{k}}}{k} \int_{(1-k)C^{-\frac{1}{k}}}^{\infty} t^{-\frac{1}{k}}(1-t)^{\frac{S}{1-k}} dt$$

ここで,

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

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．

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

以上の考え方で，ヒット率を推定し，グラフにまとめたものが図3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，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は前述のアクセスログよりもとめた値 0.75 を用いた．Cはアクセスログの観測期間の長さにより変わるので，前述のログの値に近い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．

図から明らかなように，ランダム方式のヒット率は他の手法に比べ低い．頻度方式は，将来におけるデータのアクセス順序がわからない場合の理論的上限となっているが，LRU方式は比較的良く頻度方式を近似している．

LRUを正確に実現するにはURL情報をメモリ上に記憶する必要がある．URL情報 (前述のアクセスログによれば平均54byte) の記憶には最低でもデータ本体 (同平均約10Kbyte) の1/200の記憶容量が必要となる．すなわちディスク上に配置された2Gbyteのデータ本体用キャッシュ領域の為に，10Mbyteのメモリ容量が必要となる．一部のキャッシュシステムが，正確にはLRUアルゴリズムを実装していないのは，管理プログラムの複雑化と，メモリ消費が原因となっていると推定されるが，その場合のヒット率はランダム方式に近くなり，賢明な判断とは考えにくい．

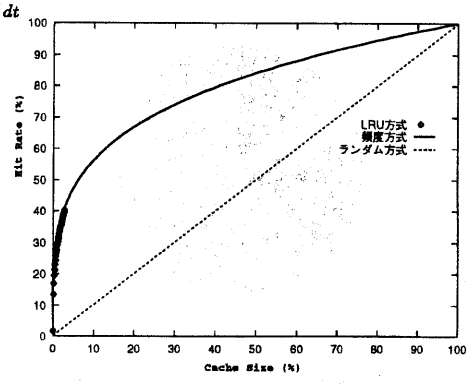

図 3: 方式毎のヒット率の比較

また，頻度方式・LRU方式ともにある容量以上にキャッシュ容量を増やしてもヒット率は増加しない．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．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になるが，図から直観的に判断した限りでは，キャッシュの中に，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 (ヒット率にして40 ～ 50%程度) 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られる．

この時，前述の式を使い，目標ヒット率hとkおよびある期間の総データ通信量 (ア

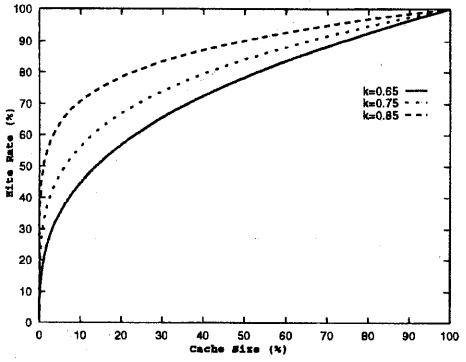

図 4: k によるヒット率の変化

クセス総数 A) から，次の式で必要なキャッシュ容量 S を求める事ができる．

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

3 分散キャッシュ方式

文献 [2] にて，我々はネットワーク上の WWW トラフィックパターンを解析することで分散型キャッシュの構成を最適化し，ネットワークのデータ流量を削減する帰納推論技術を提案した．シミュレーションによる機能確認によれば，提案技術によるキャッシュを利用した場合，キャッシュを使わない場合に比べバックボーン回線への負荷を約 26% 削減できる．この分散型キャッシュによるデータ削減効果は，従来より検討されていた階層型キャッシュの約 2.5 倍である．[2]

前節の議論を用いれば，なぜ分散キャッシュが階層型キャッシュより効率的であったかの理論的分析もできる．

我々が文献 [2] で提案した分散キャッシュ法は，Graph Based Induction (GBI) 法と呼ぶ帰納推論法を用いて，予めネットワークの上に良く流れるデータのパターンを抽出し，抽出されたパターンを構成するデータをキャッシュし，それ以外を無視する．すなわち，図 1左側のアクセス回数が多いデータをキャッシュし，それ以外を無視するよう動作する．すなわち，提案した手法では，複数の caching proxy で負荷を分散することまで考えているが，本質的には図 3に示した頻度方式の動作となっている．一方階層型キャッシュは LRU 方式で動作している．

ここで重要な点は，実際のネットワークの状況を勘案してシュミレーションで設定されたトラフィックの負荷は，図 3において左端に近い極端に記憶容量の足りない状況であるという事である．極端に記憶容量の足りない状況では LRU 方式は頻度方式を旨く近似できない．この事が両者のパフォーマンスが異なる 1 つの理由になっている．すなわち，LRU 方式のキャッシュは基本的に proxy を通過する全てのデータをキャッシュしようとするが，分散キャッシュ方式ではアクセス回数の少ないデータを無視してしまう．これは記憶容量の小さな場合に効果的な戦略と言える．

さらに文献 [2] の方法は URL 単位でアクセス回数を解析せず，サイト単位でアクセス回数を調べ，アクセス回数の多いサイトを選んでいる．これは制御情報 (何処の proxy に何処のデータが cache されているかの情報) の転送量を小さくするためのアイデアであるが，この事の影響評価は重要である．

図 5に URL 単位でアクセス回数を解析して頻度方式を用いた場合と，サイト単位でアクセス回数を解析して頻度方式を用いた場合のヒット率を比較して示す[3]．記憶容量が全てのデータを記憶できる程大きな場合，サイト単位でアクセス回数を解析した

[3] この図は図 1に示した実データを用いて作成した．始めにキャッシュが空である事を仮定している為，各 URL データの最初のアクセスは必ずキャッシュミスを起こし，図3と異なり図右端でヒット率は 100% になっていない．

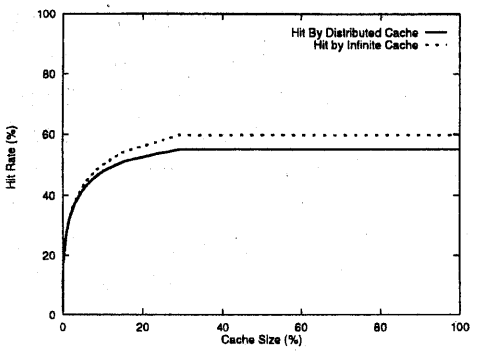

図 5: 分散キャッシュ

のでは，理論的な上限の95%程度のヒット率しか達成できないが，ヒット率40%程度までは両者に大きな差はみられない．前述の「ヒット率40 ～ 50%程度を目標」に従えば，サイト単位の分析が妥当なものであることが結論できる．

図6にサイト毎にアクセス回数を調べた結果を示す．URLのアクセス回数と同様にサイト毎のアクセス回数もZipfの法則がなりたっているのが読み取れる．このサイト単位でもZipfの法則がなりたつ，すなわち，サイトによりアクセス回数の差が大きい事が，GBI法により，比較的簡単に代表的なトラフィックパターンを抽出できることの理由になっている．一般にアクセス回数の多いデータはアクセス回数の多いサイトのデータであり，この事により，本来概念学習方法として提案されたGBI法を，ネットワークのトラフィック削減という全く別の目的に応用することが可能となっている．

4 おわりに

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はZipfの法則がなりたつ事を報告した．この知見にもとづき，種々のキャッシュ方式と記憶容量

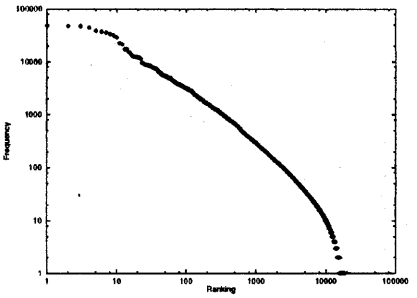

図 6: サイト毎のアクセス回数

毎にキャッシュ・システムの性能を評価した．また，少数のアクセス回数の高いデータを記憶する事に特化した分散キャッシュ方式の妥当性について述べた．

本研究の結果を用いれば，従来は明確な基準無しに進めていたWWW caching proxyの設備計画を，コストと効果を推定した上で行なうことができる．

謝辞

本研究にあたり，種々コメントいただいた奈良先端科学技術大学院大学知念賢一氏，日立製作所丹羽芳樹氏，小野木敏之氏に感謝します．また，本研究に用いたデータは日立製作所情報システム管理本部より御提供いただきました．同所沖山哲氏を始めとするデータ収集に御尽力いただいた方々に感謝します．

参考文献

[1] Chankhunthod, A., Danzig, P. B., Neerdaels, C., Schwartz, M. F., and Worrll, K. J.: A Hierarchical Internet Object Cache, *USENIX96*, 1996, pp. 153~163.

[2] 吉田: 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*, 1996, pp. 59–68.

[3] 水谷: 数理言語学, 培風館, 1982.