マルチメディア通信と分散処理 83－9
（1997. 6. 6）

WWWにおける動的経路制御を用いた多段キャッシュシステム

田中 友英　　篠田 陽一

北陸先端科学技術大学院大学情報科学研究科篠田研究室
E-Mail: {tomohide, shinoda}@jaist.ac.jp

WWWは、新しい情報流通形態として注目される反面、バックボーンの帯域浪費が深刻な問題となりつつある。この状況は、今後さらに拡大することが予想され、安定した情報流通は困難になることが予測される。現在の多段WWWキャッシングシステムは、キャッシングプロキシのサービスの対象が局所的な領域に限定されているため、キャッシュした情報の再利用率が極めて低く、WWWサーバやネットワークへの負荷はそれほど減少していない。本稿では、情報の効率的な再利用、資源の浪費抑制のために必要となる広域における多段キャッシングシステム構築についての提案をおこなう。このシステムでは、情報源のWWWサーバを起点とし、情報の利用頻度に応じて、その複製が拡散する効率的な情報流通を実現している。

A Hierarchical Web Caching System based on Dyanamic Path Control

Tomohide Tanaka　　Yoichi Shinoda

Japan Advanced Institute of Science and Technology, Hokuriku
E-Mail: {tomohide, shinoda}@jaist.ac.jp

In this paper, we propose a wide area hierarchical HTTP caching architecture with better object reuse factor and network resource utilization. The current caching architecture suffers the from low reuse factor and high load to the network resources, because the architecture can handle interaction between caching proxy and their client within the limited region of the Internet. The proposed architecture implements an effective caching architecture which gradually spreads objects from sources according to their reference frequency.

1 はじめに

様々な形式の情報 (テキスト、画像など) を対話的に選択し、転送することができる点においてWWWは優れたサービスといえる。しかしながら、以下に示すような重大な問題を抱えている。

ネットワーク網に流通する情報量の増大
操作が非常に簡単なGUIベースのブラウザの出現により、WWWの利用者は現在、増加の一途を辿っている。また、画像、動画などの大容量の情報の増加にともない、流通する情報量が増大し、帯域の浪費を招いている。

転送時間の増加

コンピュータの増加によるネットワーク距離の増加や、大容量の情報の増加などによって、転送時間が増加し、WWW の長所であるはずの対話的な側面が失われつつある。

これらの問題を解決するため、CERN httpd[1] や Squid caching server[2] などにより提案される多段キャッシュが効果的であると考えられている。しかし、以下に示す問題により、現行の多段キャッシュには、広域で使用できないという致命的な欠陥がある。

- キャッシングプロキシの多段階層が深くなるにつれ、上位に位置するキャッシングプロキシへの負荷が集中し、破綻をきたす。
- クライアント-WWW サーバ間がネットワーク的に比較的近い場合でも、キャッシュミスした場合、最上位のキャッシングプロキシを迂回することとなり、遅延、ネットワーク負荷ともにリスクを負う。この迂回を防ぐために、近辺の WWW サーバを全て知識として与えておく方法や FQDN の利用などが提案されいるが、抜本的な解決ではなく、いずれも広域なキャッシュに耐えうるものではない。

本稿では、特に情報流通量の多いとされる WWW において、同一の情報が幾度となく流通するという問題を解決するために、ネットワーク全域での多段キャッシュを提案する。また、このシステムを実現するための方法として、経路探索エージェントによるプロキシ経路探索について提案する。

2 動的な多段キャッシュシステム

2.1 動的な多段キャッシュシステム

現行の静的な多段キャッシュシステムは、最上位ノードで枝が収束する木構造であり、負荷が一点に集中することが最大の問題となっている。これは、メッシュ構造をとることで、改善することが可能となる。しかし、木構造の利点として、クライアントから要求を受けたキャッシングプロキシは、キャッシュミスすれば、静的に指定されている上位のキャッシングプロキシに要求を中継するという単純な仕組みであるため、次に要求するキャッシングプロキシを選択する必要がなく非常に簡単であった。一方、構造が、メッシュ状になると、選択の幅が広がることにより、キャッシュミスの際の中継先を適切に選択しなければならないという問題が生じる。

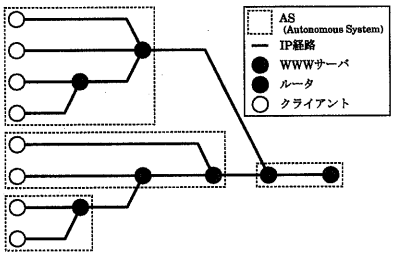

図 1: WWW サーバからの IP 経路の重複

最適なキャッシングプロキシの中継経路 (以下、プロキシ経路) 自体は、非常に簡単である。WWW サーバからみれば、ネットワーク上に存在する任意の 2 つのクライアントへの IP 層での経路 (以下、IP 経路) は、途中まで重複する経路が存在する (図1)。したがって、IP 経路の分岐点にキャッシングプロキシを配置し、クライアントは、分岐点に存在するキャッシングプロキシを多段キャッシュとして利用することで、その分岐点で情報を貯蓄することができる。その後、異なる方向に位置するクライアントからの同一の要求があった場合に、分岐点でキャッシュされていれば、その分岐点に位置するキャッシングプロキシから情報を提供できるようになり、同一の IP 経路に同一の情報が流れるのという帯域の浪費を激減することが可能となる。

このように、WWW サーバとクライアント間の IP 経路からできる限り迂回しないような中継が望ましい。IP 経路の迂回により、情報の広域拡散につながる可能性がないわけではないが、実質転送速度の低下や再利用率の低下につながることが多いため、情報の無意味な拡散は避けるべきである。

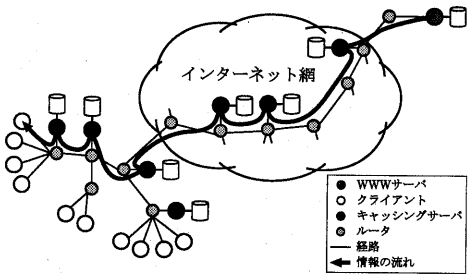

図 2: 動的な多段キャッシングプロキシ

本方式の利点は、WWW サーバを最上位とする木構造を構成することができることにある。これは、情報を管理する WWW サーバを中心として、利用される頻度が高い情報ほど、より高速に、より広域に、波紋状に広がっていき、非常に円滑な情報の拡散が可能となることを意味している。

プロキシ経路を見つけ出すことで、図 2 に示すような広域を対象とする動的な多段キャッシュシステムを構築することが可能となる。ここで最大の問題となるのは、プロキシ経路をどのようにして見つけるかであるが、このための手段として、探経路探索エージェントによるプロキシ経路探索について提案する。

2.2 プロキシ経路探索

経路探索エージェントは、キャッシングプロキシと同一のコンピュータ上に存在し、プロキシ経路を探索するためのエージェントのことである。この経路探索エージェントは、知識として近傍の経路探索エージェントが与えられている。プロキシ経路の探索要求を受けた経路探索エージェントは、知識として与えられている経路探索エージェント (以下、候補エージェント) の中から最も IP 経路に沿った経路探索エージェントを検出し、その候補エージェントに探索要求を出すことを繰り返しおこない、プロキシ経路を探索する。

経路探索には、プロキシ経路探索には、探索委託 (discovery commit)、探索要求 (discovery request)、距離計測 (distance measure)、距離報告 (distance reply) の 4 つのエージェント間の通信のためのコントロールメッセージをもちいておこなう。その手順は、次に示すとおりである。

1. 探索委託メッセージを受けた経路探索エージェント (以下、主導エージェント) は、次の経路探索エージェントを探索するため、知識として与えられた候補エージェントに、距離計測を行うように探索要求メッセージを発行する。

2. 探索要求メッセージを受けた全ての候補エージェントは、WWW サーバに向けて距離計測メッセージを発行する。

3. 距離計測メッセージを受けた WWW サーバは、主導エージェントに対して、距離報告メッセージを送信する。

4. 距離報告メッセージを受けた主導エージェントは、それらの距離報告メッセージの中からもっとも距離が短いものを選択し、次に主導エージェントとなるべき候補エージェントに探索委託メッセージを送る。

これらの操作を繰り返しおこない、順にプロキシ経路を探索する。

次にクライアントから WWW サーバへのプロキシ経路探索の具体例として、図 3に与えられている知識の関係[1]、図 4、5 に探索過程を示す。

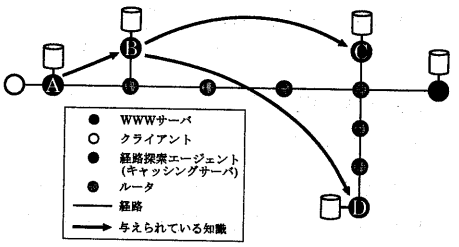

図 3: プロキシ経路探索 (知識)

[1]各経路探索エージェントに与えられている知識は、最低限の知識のみが示されており、この例で使用しない知識については省略されている。

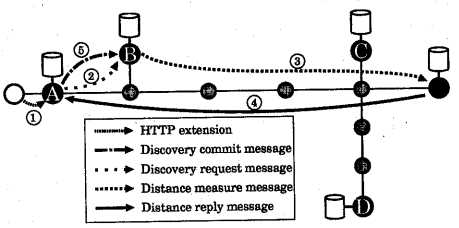

図 4: プロキシ経路探索 (1)

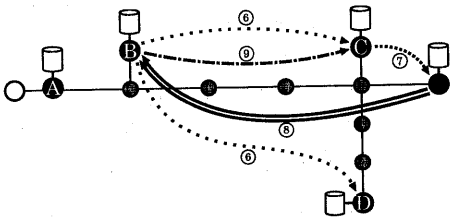

図 5: プロキシ経路探索 (2)

1. クライアントからもっとも近いキャッシングプロキシ (A) への通常の HTTP Request。

2. キャッシングプロキシ (A) と同一のコンピュータ内で動作している経路探索エージェント (A) が、主導エージェントとなり、その知識に基づき、候補エージェント (B) に距離計測を依頼する。

3. 距離計測の依頼を受けた候補エージェント (B) は、WWW サーバまでの距離を計測する。

4. 距離計測の結果を依頼していた主導エージェント (A) に報告する。

5. 計測結果を受けた主導エージェント (A) は、WWW サーバまでの距離が、自分からの距離よりも近い候補エージェント (B) に次の経路探索エージェントの探索を委託する。

6. 探索委託メッセージを受け、主導エージェントとなった (B) は、その知識に基づき、経路探索エージェント (C)、及び (D) に距離計測を要求する。

7. 距離計測の依頼を受けた経路探索エージェント (C)、及び (D) は、ターゲットとなっている WWW サーバまでの距離を計測する。

8. 距離計測の結果を依頼していた主導エージェント (B) に報告する。

9. 計測結果を受けた主導エージェント (B) は、WWW サーバまでの距離が、自分からの距離よりも短く、かつその距離が最も短い候補エージェント (C) に次の経路探索エージェントの探索を委託する。

10. 探索委託メッセージを受け主導エージェントとなった (C) は、与えられている知識がないため、この時点で経路探索を終了する。

3 評価

現在使用されている静的な多段キャッシュ方式と本方式による動的な多段キャッシュ方式において、キャッシングプロキシの飽和の速さを比較するためのシミュレーションを行った。

ネットワークのシミュレーションモデルとして、図 1でも示したようなモデルとし、多分木構造をもちいた。最上位ノードが WWW サーバ、最下位ノードがクライアント、中間ノードが多段キャッシングプロキシ、または単なるルータとしている。

条件として、最下位のキャッシングプロキシ (クライアント) は、WWW サーバの管理する情報 (65536 個) の中から無作為に選択し、その情報を要求する。要求された情報は、キャッシングプロキシまたはルータを経由し、WWW サーバへと向かう。経由中のキャッシングプロキシに目的の情報がキャッシュされていれば、キャッシュヒットした情報を下位のキャッシングプロキシに中継する。このとき、一度キャッシングプロキシを通過した情報は、必ずそのキャッシングプロキシに蓄積されていくものとする。

現在の静的なキャッシングプロキシでは、クライアントから 2 ～4 段の多段キャッシュであることから、クライアントから下位数ノード分をキャッシングプロキシとし、残りの中間ノードをルータて使用する。一方動的なキャッシングプロキシで

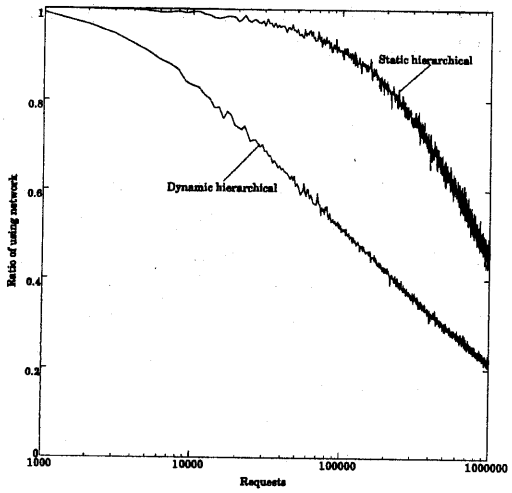

図 6: 初期状態からの成長 (10 hop)

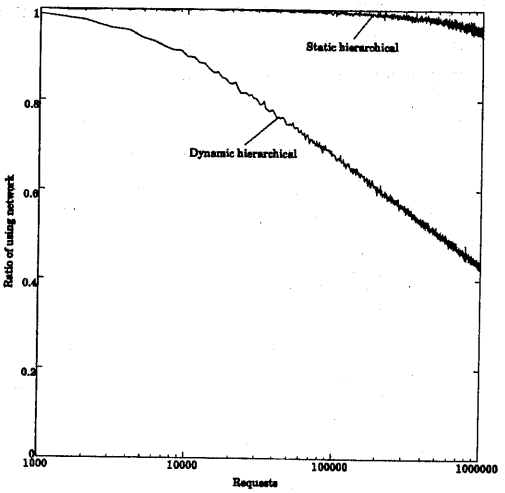

図 7: 初期状態からの成長 (15 hop)

は、ネットワーク全体を多段キャッシングプロキシとして利用できるため、全ノードをキャッシングプロキシとしている。このときのキャッシングプロキシの飽和の速さを比較する。

評価の基準は、クライアントが要求を直接 WWW サーバに出した場合のネットワークの距離を 1 としたとき、キャッシングプロキシを使用することで、使用したネットワークの距離がどれだけ短くなったか、すなわちトラフィックがどれだけ減少したかの比較によっておこなった。

最下位のキャッシングプロキシから WWW サーバまでの経由数を 10, 15 とした場合に、すべてのキャッシングプロキシがキャッシュしていない初期状態から 1,000,000 回の無作為要求で、ネットワークの使用率がどのように減少していくかを片対数グラフ化した (図 6, 7)。これらのグラフでは、縦軸がネットワークの使用率、横軸が要求の回数 (対数表記) を示している。

このグラフより、プロキシ経路が長距離に及ぶ場合 (15 metric)、1,00,000 回の要求後、トラフィックが約 45% にまで押えられている。これは、WWW サーバの近辺に存在するキャッシングプロキシから、遠方のキャッシングプロキシに向かって、ネットワーク全体へと情報が円滑に拡散していることを表していると考えられる。これにより、初期状態から飽和状態に移行するまでの時間が短縮されていると予想できる。

4 考察

経路探索に要するコスト

順にプロキシ経路を探索していく本方式では、コントロールメッセージには、十分な信頼性は必要なく、UDP で十分である。このことを考慮し、UDP による経路探索をおこなう場合、予想されるメッセージの総数は、

$mess$	のべメッセージ数
$metric$	経由する経路探索エージェント (キャッシングプロキシ) の数
$anum$	経路探索エージェントが知識として与えられているまわりの経路探索エージェントの数

とするとき、

$$mess = \sum_{i=0}^{metric} (3 \times anum_i + 1)$$

と表現できる。この式から明らかなように、経路探索エージェントの知識の量や多段キャッシュの段数が増加したとしても、爆発的なメッセージ増加にはつながらず、知識や段数の増加と比例的な増加にしかならない。これは、動的な多段キャッシュシステムによるトラフィック減少の効果と比べると非常に小さいものであり、無視できる範囲であると考えてよい。

また、WWW には通常情報のリンクが存在し、同一の WWW サーバの管理する情報を要求することが非常に多い。キャッシングプロキシのログより実際に計算してみると、クライアントが WWW サーバへ初めて要求を出してから、10 秒後で 61.3%、15 秒後でも 57.5%のクライアントが同一の WWW サーバに対して再度要求をおこなっていた。本方式では経過時間とともにプロキシ経路が判明していくという形態をもちいているため、経路が判明した部分を徐々に利用していくことで、段数が時間の経過とともに深くなり、より広域でのキャッシュがおこなわれることになる。

経路キャッシュ

本方式の最大の問題点は、経路探索にある程度の時間を必要とることであるが、これは経路をキャッシュすることで、かなりの探索を省略することができると考えられる。この経路情報のキャッシュは、基本的に、以前に調べた経路をある程度の間、キャッシュするだけのものであるが、特に要求の多い WWW サーバへの経路はキャッシュされる可能性が高く、非常に効果があると考えられる。

インターネットの状態変化による影響

インターネットでは、経路の切断、経路の変更など、常に状態を変化させている。したがって、IP 経路とキャッシュされているキャッシングプロキシ経路とが大きく異なってくる可能性がある。

この IP 経路の変更は、ルーティング的に大きな迂回を強いられる結果となる可能性がある。しかし、IP 経路が振れる直前まで、経路上にいたキャッシングプロキシを使用していたことで、クライアントが要求の対象としている WWW サーバの情報を既に大量にキャッシュしている可能性もあり、それほど、急激に効率が悪くなるとは言えないと考えられる。

5 まとめ

本稿では、効率がよい方式として提案されている多段キャッシュ方式を格段に効率よくキャッシュするための方式として、動的な多段キャッシュ方式に関する提案をおこなった。これは、木構造からメッシュ構造へと変更すること方式で、現在の狭い範囲を対象とした多段キャッシュ方式から広域ネットワークを対象とした多段キャッシュが可能となり、キャッシングプロキシの負荷分散、ネットワークへの負荷軽減とこれに伴うクライアント環境の向上、及び WWW サーバへの負荷軽減につながることを示した。

また、動的な多段キャッシュ方式において、必要不可欠となるキャッシングプロキシの効率的な選択方法として、探索エージェントによる経路探索について述べた。キャッシングプロキシを経由するにしたがって、WWW サーバへと徐々に近づき、ヒット率があがる経路選択方法として、経路探索エージェントが非常に有効であることを示した。

参考文献

[1] Ari Luotonen, Henrik Frystyk Nielsen, and Tim Berners-Lee. "CERN HTTPD," public domain full-featured hypertext/proxy server with caching, June 1996. Available from <URL:http://www.w3.org/pub/WWW/Daemon/Status.html>.

[2] "Squid Internet Object Cache," high performance proxy caching for Web clients, January 1997. Available from <URL:http://squid.nlanr.net/Squid/>.