システムソフトウェアと
オペレーティング・システム 77－36
マルチメディア通信と分散処理 87－36
（1998. 2. 27）

World Wide Web におけるコネクションキャッシングの評価

大澤 範高　武居 太郎　柳川 和久　弓場 敏嗣
〒182-8585 東京都 調布市 調布ヶ丘 1-5-1
電気通信大学大学院情報システム学研究科
E-mail: {osawa, taro, katze, yuba}@yuba.is.uec.ac.jp

概要　本稿では、World Wide Web(WWW) にコネクションキャッシングを適用した場合に、ページにアクセスする際のオーバーヘッドの低下の評価を報告する。コネクションキャッシングとは、WWW サーバ、または、WWW プロキシが、転送が終わった後もクライアントとのコネクションを解除せずに保持し、再び利用する事である。本稿では、WWW のコネクションキャッシングのヒット率が高く、この手法が WWW に対して有効である事を示す。16 コネクションでのキャッシングのヒット率は、プロキシとサーバの両方において 81%以上であった。

キーワード　コネクションキャッシング, World Wide Web, トレースベースシミュレーション

Effectiveness of Connection Caching on the World Wide Web

Noritaka OSAWA　Taro TAKEI　Kazuhisa YANAGAWA　Toshitsugu YUBA
Graduate School of Information Systems
The University of Electro-Communications
Chofugaoka 1-5-1, Chofu, Tokyo 182-8585, JAPAN

Abstract　This paper proposes and evaluates connection caching to reduce the overhead of accessing pages on the World Wide Web (WWW). Connection caching means that a WWW server or a WWW proxy does not release its connection with a client or a peer but retains it, and uses it again after the transmission is completed. This paper shows that the hit rate of connection caching for WWW accesses is high and connection caching is effective on the WWW. The caching of 16 connections gives hit rates of more than 81% at both a proxy and a server.

key words　connection caching, World Wide Web, trace-based simulation

1　はじめに

World Wide Web(WWW) はインターネットにおいて広く普及している。インターネットのネットワークトラフィックの大部分が、WWW のアクセスで占められている。ユーザのアクセス遅延の増加とサーバの負荷も問題である。その主な原因はネットワークの遅延とサーバの過負荷である。

キャッシング技術を用いれば、この問題を緩和することができる。サーバ、プロキシ、クライアントでデータキャッシングがよく用いられている。クライアントに近い所に、繰り返してアクセスされるデータをキャッシュしておけば、ネットワークトラフィックとアクセス遅延を減少させることができる。その上、キャッシングによってアクセス回数が減少することにより、WWW サーバの負荷もまた減少する。データキャッシングに関する様々な提案がある [1] [7] [8] [13] [17]。我々も、プロキシサーバの世代型キャッシングについて提案している [14]。

データキャッシングだけでなく、コネクションキャッシングもまた可能である [16]。コネクションキャッシングとは、WWW サーバまたは、WWW プロキシが、データ転送の終了後もクライアントとのコネクションを切らずに保持しておき、後に再利用することを表す。そのまま繋いでいるコネクションは、キャッシュされ、将来アクセスがあった時に使われる。コネクションを確立する際には、ネットワークトラフィックが発生し、サーバの応答が必要であるが、コネクションキャッシングを用いることにより、ネットワークトラフィックとサーバの負荷を減らすことができる。

本稿では、WWW サーバ、およびプロキシに対するコネクションキャッシングの提案と評価報告をする。本稿では、電気通信大学 (以下、UEC) の総合情報処理センター (以下、IPC) のアクセスログを用いて、プロキシとサーバにおけるコネクションキャッシングのヒット率について評価と考察を行

う。評価は350日間のログに基づいている。

WWWのコネクションキャッシングに対する解析は、以前の論文[15]で報告した。その解析では、ホスト間にはコネクションが一つ以上確立することがないとした。もし、プロキシからサーバに対してただ一つのコネクションしか用いられない場合、サーバへのアクセスが同時に複数要求されると、アクセス遅延が増大する。一方、本稿ではサーバに対して複数コネクションを確立することが許される場合の評価を報告する。IPCのWWWサーバおよび、プロキシにおいて、16コネクション・キャッシュのヒット率は81%以上である。また、クライアントからプロキシへのリクエストの29%以上が、グローバル16コネクション・キャッシュにおいて、サーバへのコネクションを共有する。

2 ログに基づく評価法

ここでは、ログ収集に使われたサーバとログについて説明する。Universal Resource Locator(URL)[4]で参照される実体を、本稿の中では、ページと呼ぶ。コネクションキャッシングの解析では、URLのホスト部をホストの特定に利用する。

2.1 サーバとユーザ

UECのIPCにおけるプロキシ、サーバとそのユーザについて述べる。

IPCにおける教育用ワークステーション(WS)は、直接学外のサーバと通信することはできない。したがって、教育用WSのユーザはプロキシを用いなければならない。IPCでは、プロキシにCERN httpd[12]を用いていた。このプロキシはIPCの教育用WSのユーザ全てが用いていたと考えられる。このプロキシは、キャッシングプロキシサーバを持っていない各学部や研究室でも使われていた。

IPCにおける、教育用WSのWWWクライアントにはNCSA Mosaic[2]が用いられていた。Mosaicは、一セッション分以上のキャッシュを保持しなかった。また、IPCの教育用WS以外では、Mosaic以外のWWWクライアントも用いられていた。Netscape Navigatorや、Internet Explorerを用いているホストもあり、これらはセッションをこえてキャッシュ内容を保持する。

UEC(http://www.uec.ac.jp/)のWWWサーバはUEC内部、外部、両方から参照される。ここには、UECの紹介があり、学部へのリンクがある。

2.2 ログ

クライアント、プロキシ、サーバの関係を図1に示す。クライアントホストはプロキシを経由して、あるいは直接、WWWページにアクセスする。プロキシのログはクライアントホストのアドレスと、サーバのホストアドレスを共に記録している。WWWサーバのログはサーバ上のページにアクセスしたホストを記録している。

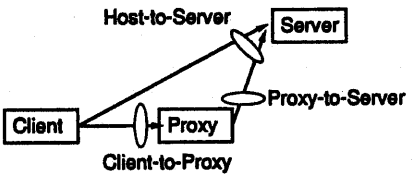

図1: クライアント、プロキシ、サーバの関係

2.3 ログデータ

ここにログ中の記録の例を示す。

gold.yuba.is.uec.ac.jp - - [31/Oct/1995:13:31:32 +0900] "GET http://www.yuba.is.uec.ac.jp/~osawa HTTP/1.0" 200 1620

共通のログファイルフォーマットは次の通りである。フォーマット中の各項目は、表1に示す通りである。

```
remotehost rfc931 authuser [date] "request" status bytes
```

remotehost	リモートホストの名前
rfc931	リモートホストの使用ユーザ名
authuser	認証ユーザ名
date	リクエストの日付と時刻
request	クライアントからのリクエスト
status	HTTP状態コード
bytes	転送された文書の全長

表1: 通常のログファイルフォーマットの欄の意味

プロキシのログでは、remotehost (例ではgold.yuba.is.uec.ac.jp) となっている部分がクライアントホストを示し、request 部に含まれる URL のホスト部 (例ではwww.yuba.is.uec.ac.jp) がサーバホストを記録している。サーバのログでは、remotehostとなっている部分がクライアントを表している。

date は、リクエストが処理され始めた時刻のタイムスタンプである。行の順序は、リクエストの完了した順番を示している。したがって、リクエストの開始順序と、終了した順序をログから読みとることができる。リクエストの開始順序は、タイムスタンプで特定できる。終了順序は、ログファイル内に前に述べたフォーマットで記述されているため、行の順序で識別することができる。

ページの転送時間はログに記載されていないため、このログを基にしてタイミングや遅延時間の正確な評価をするのは不可能である。しかし、このログはコネクションキャッシングのヒット率評価には十分な情報を含んでいる。

3 ログデータの特徴

Hyper Text Transfer Protocol(HTTP)[3] による通信を解析した。正常終了したアクセス (status コードが 200 であるもの) だけでなく、正常終了しなかったアクセスについても、コネクションを必要とする。両者とも解析対象とした。ログデータは、1995 年 10 月 24 日 13 時から、1996 年 10 月 8 日 13 時までのものである。ログは 350 日分になる。プロキシへの総アクセス回数は、15,220,541 回であった。一日の平均アクセス数は 43,487.3 回/日である。WWW サーバへの総アクセス回数は、1,371,193 回であった。一日の平均アクセス数は 3,917.7 回/日である。

図 2と図 3は、時間帯ごと、曜日ごとの、それぞれのアクセス回数の比率を表している。時間帯のピークは、午後 3 時から午後 4 時の間にある。曜日では、平日にアクセスが多い。アクセスの分布は予想されるものである。我々のログに特別な点はない。

Glassman[7] は、WWW サーバ上のページのアクセス頻度は Zipf の法則 [10] に従うと述べている。もし、Zipf の法則が成り立っているのなら、アクセスに局所性があるはずである。ホストのアクセス頻度と、同じホストへのアクセスの間隔を調査した。我々の知る範囲では、このことに関する報告はない。

いま、アクセス頻度 fでアクセスするホストの数を $P(f)$ とする。アクセス頻度が fであるアクセス回数の総数は $A(f) = fP(f)$ になる。ここで、Zipf の法則が成り立つのなら、$A(f) = M/f$ (M は定数) となる。変形すると、$P(f) = M/f^2$ という式が得られる。$P(f)$ は両対数グラフで右肩下がりの直線となる。同じアクセス頻度を持つホストの数を図 4に示す。X 軸はアクセス頻度、Y 軸は同じアクセス頻度のホストの数である。

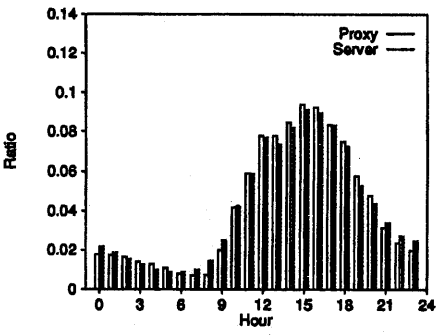

図 2: プロキシにおける、時間ごとの正規化したアクセス頻度。アクセス頻度はアクセスの総数によって正規化している。Proxy と Server は、それぞれ、プロキシへのアクセスとサーバへのアクセスを示す。

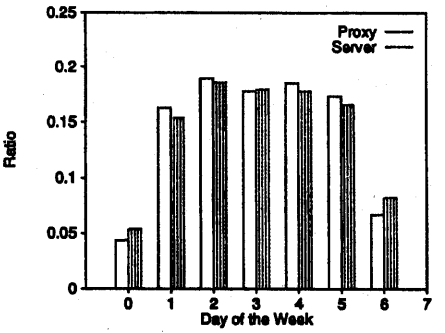

図 3: プロキシにおける、曜日ごとの正規化したアクセス頻度。アクセス頻度はアクセスの総数によって正規化している。X 軸の 0 から 6 までの数値はそれぞれ日曜日から土曜日までを示す。

図 4では、アクセス頻度が低い場合のプロキシ-サーバ間とクライアントホスト-サーバ間には Zipf の法則が当てはまる。クライアントホスト-プロキシ間では Zipf の法則があまり当てはまらない。これは、クライアントが大学内の端末であり、プロキシのクライアント数が限られているためと考えられる。

プロキシ-サーバ間でもっとも高いアクセス頻度を持つホストは IPC の教育用 WWW サーバホストであり、ここには教育用コンピュータに関連したホームページがある。教育用 WS の Mosaic は、セッション間のキャッシュデータを保持しないため、起動時にそのホームページにアクセスする。

図 5は、アクセス間隔の分布を示している。X 軸

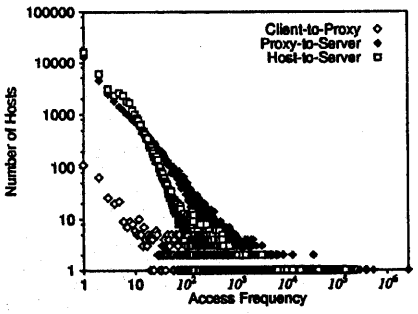

図 4: 同じアクセス頻度をもつホストの数。Client-to-Proxy、Proxy-to-Server、Host-to-Serverは、それぞれ、クライアントからプロキシ、プロキシからサーバ、リモートホストからサーバへのアクセスを示す。

は、同じホストに対するアクセス間隔である。アクセス間隔とは、あるホストに対するあるアクセスから次のアクセスまでの間隔である。アクセス間隔はアクセスの局所性を示す。高い局所性があると、キャッシュのヒット率が向上する。キャッシュのヒット率が上がれば、アクセスの負荷が減少する。図 5は、ホストに対するアクセスの局所性が高いことを示し、コネクションキャッシングが有効である事を示唆している。コネクションキャッシングのヒット率の定量的評価は後ほど示す。図 5では、同じホストに対するアクセスの間隔が短い区間で、Zipf の法則が成り立っている。

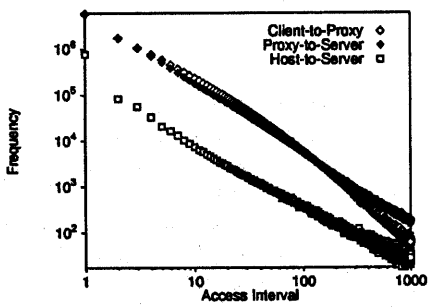

図 5: 同じホストに対するアクセス間隔の分布。

4 コネクションキャッシング

ここでは、クライアントからプロキシ、プロキシからサーバ、クライアント又はプロキシからWWWサーバへのコネクションキャッシングのヒット率の評価を示す。この評価はIPC におけるプロキシサーバと WWW サーバでのログに基づく。

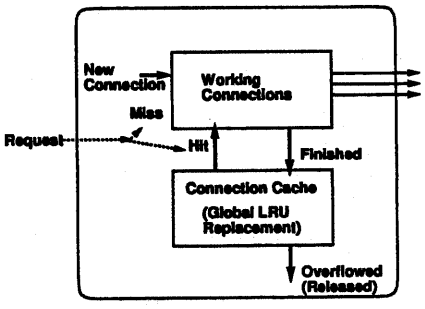

図 6: コネクションキャッシュ

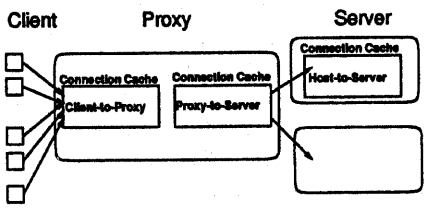

図 7: コネクションキャッシュの利用法

図 6 と 図 7 はそれぞれ、本稿で提案している、コネクションキャッシュの内部構造とコネクションキャッシュの利用法を図式化したものである。

サーバとプロキシでは、すべてのコネクション要求が受理される。コネクションを用いた処理が終了したとき、コネクションはすぐには解放されず、コネクションキャッシュに入れられる。コネクションキャッシュのアルゴリズムはLRU アルゴリズムとする。LRU アルゴリズムでは、もっとも長い間使われていないエントリを新たなエントリに置き換える。LRU アルゴリズムと単純化された LRU アルゴリズムはキャッシュの置き換えで広く用いられている。置き換えが起こると、図 6のように、キャッシュからあふれたコネクションが解放される。

クライアントがプロキシを経由してあるページを要求したとき、プロキシはページのあるサーバへのコネクションを探す。プロキシが、コネクションキャッシュ中に、そのサーバへのコネクションのうち使われていないものを見つけると、そのコネクションを用いて、サーバと通信する。さもなく

ば、通信遅延を起こさないようにプロキシはサーバへの新たなコネクションを確立する。

同じホストへの複数のコネクションがコネクションキャッシュ中に存在できる。これは、通常のキャッシュとは異なる。通常のキャッシュでは重複したエントリは許されていないが、コネクションキャッシュでは複数の同一ホストへのコネクションが存在できる。

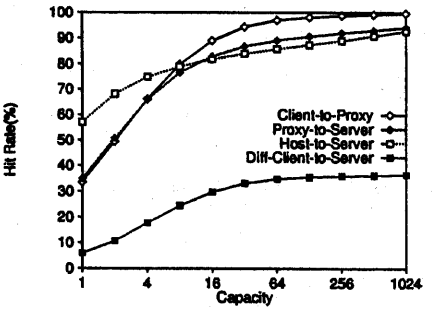

図 8: コネクションキャッシングのヒット率。`Diff-Client-to-Server`は、異なるクライアントから同じサーバへのアクセスを示す。

図 8は、16 コネクションのキャッシングを行なったとき、クライアント-プロキシ間、プロキシ-サーバ間、ホスト-サーバ間のすべてのアクセスにおいて 81%以上のヒット率が得られることを示している。この結果から、コネクションキャッシングが有効であることが分かる。

ここで用いているコネクションキャッシングアルゴリズムはグローバルアルゴリズムである。つまり、キャッシュされたコネクションは、異なるクライアントが共有できる。要求したクライアントが確立したコネクションの中からのみ、再利用対象、置き換え対象が選ばれるわけではなく、キャッシュ中のすべてのコネクションを対象にして再利用及び置き換えがなされる。クライアント `A` によって確立、もしくは用いられたコネクションは、異なるクライアント `B` も用いることができる。直前に `A` によって利用されたコネクションのクライアント `B` による使用は、異なるクライアントからの利用と見なされる。一方、新たに確立されたコネクションが、異なるクライアントからのアクセスと見なされることはない。図 8は、キャッシュ中のコネクションが、一つのクライアントだけでなく、複数のクライアントによって利用されることも示している。図 8中の `Diff-Client-to-Server` は、異なるクライアントからのコネクション利用の割合を示している。

コネクションが共有できるのは、プロキシのクライアントのすべてが一つの組織に属している事に起因するかも知れない。しかし、このようなプロキシの使い方が一般的である。したがって、グローバルコネクションキャッシュは有効であると考える。

5 関連研究

複数のページを一つのコネクションで転送するプロトコルが、[16] で提案された。これは、long-lived コネクションを使用する。ユーザが参照するあるページから参照されるページは、共に転送される。この提案はアクセス遅延の減少を目的としている。これは、クライアントが確立したコネクションをそのまま使うことができるときだけ、クライアント-サーバ間のコネクションを再利用する。言い換えれば、ローカルキャッシング技術を用いている。一方、本論では、グローバルキャッシュを用いることによって、確立しているコネクションを、任意のクライアントが使用することができる。クライアントからプロキシへのリクエストの 29%以上が、16 コネクション・キャッシュ内にあるコネクションを共有していた。我々の研究では、更に、クライアントからプロキシへ、プロキシからサーバへ、そしてクライアントからサーバへの、コネクションキャッシングの定量的な評価を行った。我々の研究は [16] とは異なっている。

Jomer[9] はトラフィックの負荷、コネクションのコストを考慮した、X.25 ネットワークがサポートする、OSI のコネクションレスネットワークプロトコル (CLNP) によってコネクションを切断するアルゴリズムについて述べている。ただし、複数コネクションが考慮されていない。

Yang 等は、[18][19] の中で、ホストグルーピングが用いられた場合のコネクションキャッシングとその性能について、シミュレーションを使って解析している。彼らは、合成したネットワークトラフィックを利用しており、WWW のトラフィックについては言及していない。

Dittia 等 [5] は、ATM ホスト-ネットワークインターフェイスチップの中でコネクションキャッシングを用いている。ただし、コネクションキャッシングの有効性は、定量的に示されていない。

6 考察

WWW ページに対するアクセスは、TCP/IP プロトコルを用いてコネクションが確立され、ペー

ジデータを転送し終わったら、コネクションが解放される。この転送方式では、アクセス遅延やサーバの負荷が増大する。しかし、TCP/IPプロトコルは信頼性のある通信を提供し、ファイアウォールを通る事ができる。TCP/IPを用いることによって、エラーおよび喪失パケットの再転送が自動的に行われるので、転送エラー訂正は問題にならない。ネットワークファイアウォールは、ネットワークセキュリティを強化するために、広く用いられている。ファイアウォールは、未知のプロトコルやUDP/IPを通さないが、TCP/IPは通す。したがって、コネクションを確立するTCP/IPの利用が現実的であり、コネクションレスプロトコルを用いるのは難しい。コネクションを確立するプロトコルを用いる際は、コネクションの確立、解放のオーバーヘッドの減少が重要である。

HTTP/1.1[6]では、永続的なコネクションの機能が導入された。それは、明示的にクローズされるまで保持される。このことにより、コネクションキャッシュ法は、永続的なコネクションを持たない標準的なHTTP/1.0の環境よりも、より簡単に実装できる。例えば、コネクションキャッシュは永続的コネクションの配列として実装できる。置き換えが起こった際には、コネクションが解放される。コネクションの解放は適切に行われるべきであり、具体的なプロトコルを研究する必要がある。

7 結論

350日間のログに基づいて、WWWサーバとプロキシに対するコネクションキャッシングの提案と評価を行なった。WWWサーバ及び、プロキシにおいて、16コネクションのキャッシュを用いた際にヒット率が81%以上である。コネクションキャッシングはアクセスの局所性を利用する。コネクションキャッシングによって十分な効果が期待できる。また、クライアントからプロキシに対するリクエストの29%以上がサーバへのコネクションを共有する。ヒット率と性能を向上させる手段として、ローカルコネクションキャッシングよりもグローバルコネクションキャッシングの方が有効である。

謝辞

ログデータの入手に便宜を図って頂いた、電気通信大学の総合情報処理センターの各位、特に伊藤秀一教授、小林克志博士に感謝致します。

参考文献

[1] Abrams, M., Standridge, C.R., Abdulla, G., Williams, S., & Fox, E.A. (1995). Caching Proxies: Limitations and Potentials. *4th Int'l World Wide Web Conference.*

[2] Andreessen, M. (1993). *NCSA Mosaic Technical Summary.* National Center for Supercomputing Applications.

[3] Berners-Lee, T., Fielding R., & Frystyk, H. (1996). *Hypertext Transfer Protocol - HTTP/1.0.* HTTP Working Group Internet-Draft.

[4] Berners-Lee, T., Masinter L., & McCahill, M. (1994). *Uniform Resource Locators.* RFC 1738.

[5] Dittia, Zubin D., Jerome R. Cox, Jr., and Guru M. Parulkar, (1995). "Design of the APIC: A High Performance ATM Host-Network Interface Chip," *IEEE INFOCOM '95 - Vol.1*, pp.179-187

[6] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., & Berners-Lee, T. (1997) *Hypertext Transfer Protocol - HTTP/1.1.* RFC 2068.

[7] Glassman, S. (1994). A Caching relay for the World Wide Web. *1st Int'l Conf. on WWW, 1994*, Geneva.; also appeared in (1994) *Computer Networks and ISDN Systems*, 27, 165–173.

[8] 一井 信吾, 中山 雅哉,(1995) "キャンパスネットワークにおける WWW キャッシングの効果." , 情報処理学会分散システム運用技術研究グループ資料 DSM-9505033,

[9] Jomer, Per,(1989). "Connection Caching of Traffic Adaptive Dynamic Virtual Circuits," *SIGCOMM 89 Symp.: Communications, Architectures and Protocols*, pp.13-24.

[10] Knuth, D. (1973). *The Art of Computer Programming, Vol. 3. Sorting and Searching.* Addison-Wesley.

[11] Kwan, T.T., MacGrath R.E., & Reed, D. (1995). NCSA's World Wide Web Server: Design and Performance. *IEEE Computer*, 28(11), 68–74.

[12] Luotonen, A., & Altis, K. (1994). World-Wide Web Proxies. *1st Int'l Conf. on WWW 1994*, Geneva.; also appeared in (1994) *Computer Networks and ISDN Systems*, 27, 147–154.

[13] 大澤 範高, 早野 文孝, 弓場 敏嗣, 箱崎 勝哉,(1996). WWW プロキシ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, 情報研報, *DPS74-33*. pp. 191-196.

[14] Osawa, N., Toshitsugu, Y., & Hakozaki, K. (1997). Generational Replacement Schemes for a WWW Caching Proxy Server. *High Performance Computing and Networking Europe (HPCN97)*, LNCS 1225, 940–949.

[15] Osawa, N., Yanagawa K. & Toshitsugu, Y. (1997). Connection Caching for WWW Servers and Proxies. *World Conference of the WWW, Internet, and Intranet (WebNet97)*, pp. 364-369.

[16] Padmanabhan, V.N., & Mogul, J.C. (1994). Improving HTTP Latency. *2nd Int'l WWW Conf*, 995–1005.

[17] Pitkow, J.E., & Recker, M.M. (1994). A Simple Yet Robust Caching Algorithm Based on Dynamic Access Patterns. *2nd Int'l WWW Conf.*, 1039–1046.

[18] Yang, Soo-mee and Yookun Cho,(1995). "A Performance Study of A Connection Caching Technique," *Proc. of IEEE WESCANEX Communications, Power, and Computing Conference - Part1*, pp.90-94.

[19] Yang, Soo-mee and Yookun Cho,(1996). "Connection Caching Technique with Host Grouping," *Proc. of 3rd Int'l Conf. on High Performance Computing*, pp.374-378.