エージェント機能を用いた 分散Webキャッシングの有効性

岸本 克巳† 谷口 祐治‡ 河野 真治† 玉城 史朗†
†琉球大学理工学研究科情報工学専攻 ‡琉球大学総合情報処理センター

概 要

WWW ネットワークトラフィックを削減するためにはクライアントから要求があったデータをキャッシュすることが有効である． キャッシュを複数の WWW プロキシに分散させて個々のプロキシの負荷を減らし，ネットワーク形態や利用状況に応じてデータの蓄積量や保存期限を定めることによって，キャッシュの有効性を高めることが期待できる．ここでは，Java の 移動エージェントである Aglets を用いて情報発信側から能動的に分散 Web キャッシュ間の協調を行う方法を提案する．そして従来型のプロキシシステムのログ解析から提案した方法の有効性について考察する．

Effectiveness of Mobile Agents in distributed Web cache system

†Katsumi KISHIMOTO ‡Yuji TANIGUCHI †Shinji KONO †Shiro TAMAKI
†Department of Information Engineering, University of the Ryukyus
‡Center for Integrated information processings, University of the Ryukyus

Abstract

To reduce WWW network traffic, one of the effective method is proxy cache which stores clients' request. To reduce the the load of the proxy, a cache can be distributed as multiple WWW proxies. Also to increse the effectiveness of the cache, a proxy's data size and caching duration are compensated according to the network usage and configuration. Here we show how to use Aglets (Java mobile agents) to coordinate a distributed Web cache proxies. Aglets autonomously start from informatoin providing sides. We investimate the effectiveness of our method from analysis of log information of conventional WWW proxies.

1 情報提供サービスの高速化

中継装置の処理速度や伝搬速度の壁を越える Internet における情報サービスの高速化が重要である．そこでプロキシという概念を導入し通信量自体を減らす様々な研究，評価が行われている．プロキシとは「インターネット上を流れる幾つかのプロトコルを代理中継し，クライアントに代わってそのサーバに取りにいき，その結果をクライアントに渡すもの」である．

このプロキシサーバへ先読み等の能動的な動きを実装することによるヒット率の向上が報告されている [6]．これはプロキシにインテリジェンスを持たせることが有効であることを示してる．

プロキシやキャッシュを，Internet 上に複数置くことにより，プロキシの負荷を減らし，仮想的にキャッシュの容量を大きくすることが可能になると考えられる．これが分散 Web キャッシュングである．そこで本研究では自律的な動作を実現する Java 移動エージェント Aglets を用いて分散 Web キャッシング間の協調を行う能動的な情報取得と提供の形態を提案する．

また，従来の Web プロキシキャッシュのログの解析から提案する方法の有効性について考察する．

1.1 Agletsによる簡単なプロキシ選択

最も簡単な分散Webキャッシュは，相互に独立なプロキシを自動的にブラウザ側から選択するものである．Aglets移動エージェントを用いてユーザの利用するブラウザからブックマーク情報を取得し，それをもとに最もキャッシュのヒットするプロキシサーバを選択するプログラムを作成した．

実行の流れを以下に示す (図 1)．Aglet Viewer とは ASDK1.0.3 に含まれるサンプルサーバプログラムである．

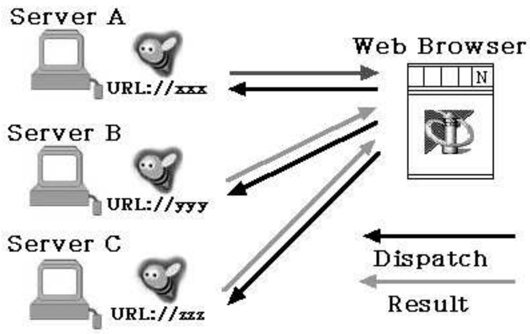

図 1: Agletsのプログラム例

1. Aglet Viewer から ProxyChanger を起動する．
2. ProxyChanger は URLSaver を呼び出し，ブラウザのブックマーク情報を取得する．そしてブックマークの数と派遣先のプロキシサーバの数だけ Worker が生成される．
3. 次に ProxyChanger はデータ入力画面を表示し，指定されたプロキシサーバのもとへ，ブックマーク情報を保持した Worker が派遣される．
4. 派遣先に到着した Worker は保持していたブックマーク情報とキャッシュ情報を比較し，ヒットする回数を計算する．
5. 処理を終えた Worker は派遣元の Aglet Viewer へ帰還し，ProxyChanger へ結果を報告する．
6. ProxyChanger は全ての Worker の報告を受けた後，処理結果から最もヒット率の高いプロキシサーバを選択し，WritePref によってブラウザの設定を行う．

実装を評価した結果，この一連の動作には，以下の問題点があることが判明した．

1. ヒット率の高いプロキシのアクセスが集中してしまい，他のプロキシのキャッシュが有効に使われない．
2. ブックマークに登録された URL とプロキシサーバの数に比例した，エージェント自体の転送によるネットワークトラフィックが発生する．
3. プロキシサーバ毎にエージェントのセキュリティポリシーを設定する必要がある．

2 システム構成と動作の概要

前述の簡単なプロキシプログラムの問題点を踏まえ，サーバ側からプロキシやクライアント側へ能動的に情報配信が行えるシステムを提案する. このシステムは以下のような動作を行う．

1. Web サーバはプロキシ資源を提供してくれる Server A と B に対して，Web サーバへのリクエストを監視する Aglet を派遣する (図 2)．通常，Server A と B はプロキシサーバとして動作する．
派遣された Aglet は Web サーバから与えられた条件 (Web サーバへのリクエスト数，時間帯等) をもとにネットワークストリームを監視する． 派遣先は UNIX コマンドの netstate，traceroute 等による情報によって決定される．そして派遣先に到達した Aglet は Web サーバによる設定に従い，状況報告を行う．

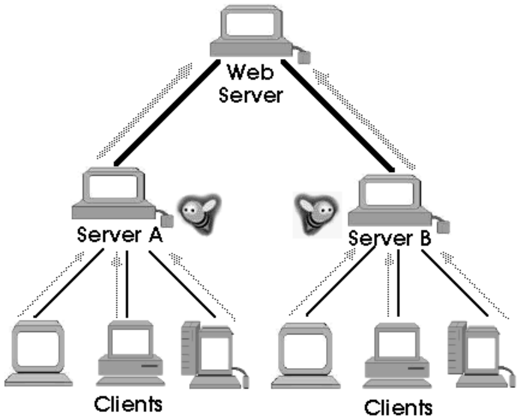

図 2: システムの動作 1

2. Aglet は派遣先の Server や Web サーバに変化があった場合に Web サーバへと帰還し，状況を報告する (図 3)．Web サーバは Aglet からの報告を受けると，別の機能を持った Aglet を生成する (図 4)．新しく生成された Aglet が行う処理は以下のようなものが挙げられる

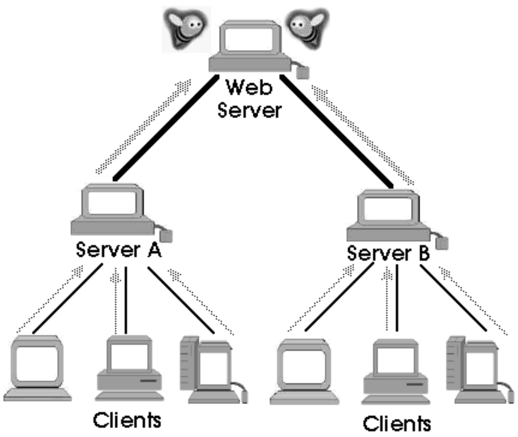

図 3: システムの動作 2

- Web サーバ向けのリクエストに応答するプロキシシサーバを構築する．Server A，B に無いリクエストであれば Web サーバへリクエストを送信する．取得した情報はクライアントへ返し，Server A，B で保持する．
- Web サーバへのリクエスト状況や時間帯等の条件によって Server A，B の資源を開放する．

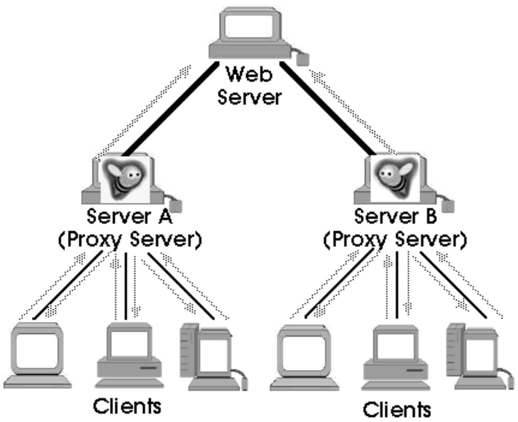

図 4: システムの動作 3

このシステムでは従来法のミラーサーバと比較すると以下のような特徴が挙げられる．

1. 各サーバは能力に応じた資源の提供が可能であり，そのセキュリティは各サーバ毎に決定する．
2. ftp ミラープログラムなどによる固定時間帯の急激なトラフィックや更新の多い Web サーバへのトラフィック等を回避でき，状況に応じてネットワークトラフィックの負荷を分散する．
3. ネットワークトラフィックの性質に合わせたプロキシ網を動的に構築する．

これらの特徴より，従来法では実現できなかったサーバ側からの動的な Web キャッシングとシステム構築が可能になる．

3 実験環境

本節では, 従来のプロキシサーバのログのデータを使い, 提案したシステムの有効性を調べる.

3.1 ネットワークモデル

従来のプロキシサーバのデータ取得は，琉球大学総合情報処理センターの情報ネットワークシステム (RAINS-G) 内の教育用サーバ及び各学部研究用サーバで行っている [8]．また，本研究のデータ取得と，実験運用は OSN(Open System Network) ネットワークで行っている [7]．

3.2 WWW 利用の傾向

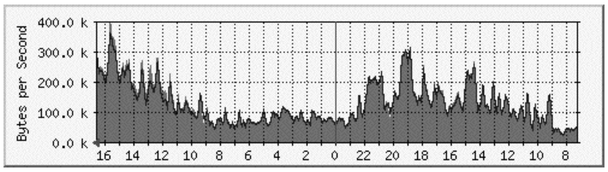

図 5: RAINS $\leftrightarrow$ SINET の 1 日のトラフィック (但し 5 分間隔の平均値を表示)

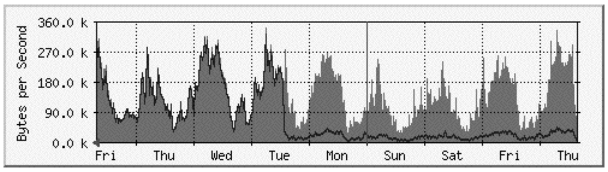

図 6: RAINS $\leftrightarrow$ SINET の 1 週間のトラフィック (但し 2 時間間隔の平均値を表示)

Internet 利用によるパケット・トラフィックはその殆どが WWW である．OSN におけるトラフィックは以下のようになっており，黒の折れ線グラフは RAINS$\rightarrow$SINET，面グラフは SINET$\rightarrow$RAINS のトラフィック量を秒単位のバイト数で示している．

図 5，6 から 1 日のトラフィック量は午後から増え始め，1 週間を通して同じような波形が読みとれる．

3.3 従来型プロキシ網の構築

図7はOSNゾーンに構築されたSquidによるプロキシサーバの階層構造を示している．プロキシサーバ網はproxy1を親サーバ，proxy2，capoeiraは子サーバとして互いにsibling接続を行い階層構造を構築し，ログ解析を行った [9, 10]．

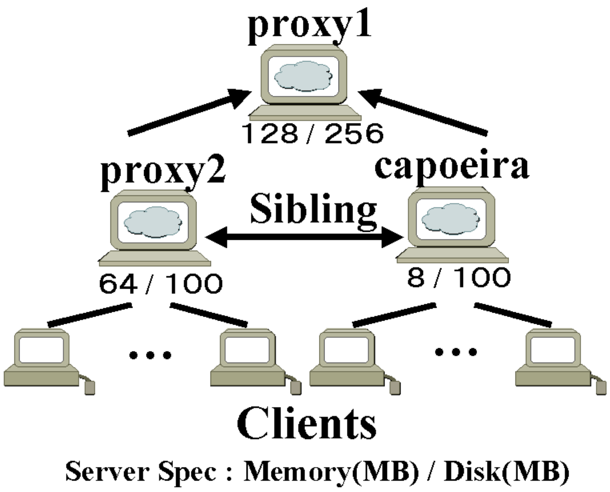

図 7: Squid による2階層プロキシサーバ構成

4 考察

表1のTCP COUNTSにおいて，proxy1におけるcapoeiraのヒット率９４％，capoeiraにおけるproxy1のヒット率６７％から，子プロキシサーバ間にsibling接続を行い，連係を取る手法が有効とわかる．capoeiraの性能が他のプロキシサーバと比べて低いが，capoeiraサーバの総リクエスト数は167468と非常に多くなっている (図8)．この結果からcapoeira単体でのプロキシの性能は十分であるとは言えず，単体でのヒット率の向上は難しい．また，proxy1サーバの利用率が子サーバを除いた１６％と低いため，階層構造の効果が見られない．

図9のクライアンのヒット率を分布図で見ると，全体的に１０％より低い位置に点在している．この結果はプロキシサーバの利用者数が少ないこと，プロキシサーバの性能と各サーバにおける利用者数にばらつきがあることが原因と考えられる．

図7の親サーバを本システムのWebサーバ，子サーバをWebサーバのプロキシサーバデータと想定すると，Webサーバへの負荷はプロキシサーバへと分散され，Webサーバ $\leftrightarrow$ プロキシサーバ間の

表 1: 各サーバ間のヒット率

サーバ名	parent 接続 又は sibling 接続	TCP COUNTS		
		counts	%all	%hit
proxy1 (親サーバ)	capoeira	19514	53%	1%
	proxy2	11442	31%	2%
proxy2 (子サーバ)	capoeira	126	1%	94%
capoeira (子サーバ)	proxy2	46	0%	67%

トラフィック軽減が予測される．また，下位のプロキシサーバのトラフィックは更に階層を重ねることによってWebサーバを中心としたネットワークトラフィックと各サーバの負荷分散を行うことが可能であり，本システムを実装した場合，capoeiraは能力に応じたWebキャッシングとヒット率向上を行える．

本研究のプロキシサーバはサーバ側である情報提供者の立場から如何に効率良くデータをクライアントに配信するかに着目している．そして提案するシステムは，Webサーバへのリクエストを他のサーバ上でAgletを用いて監視し，回線を流れるリクエストの数やその他の要因に応じてそのサーバ上にプロキシサーバを構築し，環境の変化に応じてプロキシサーバの存続，運営する形態である．これは従来のミラーサーバと比較すると以下のような特徴がある。

- 従来法によるプロキシサーバは，サーバ管理者が自主的に導入を行い，割り当てられてサーバの資源は固定されることになる．また，管理の面でもサーバ管理者が直接関わることになる．しかし本システムの場合，サーバ管理者はサーバの資源と処理能力に応じてプロキシサーバを導入することができ，また自動的に構築が行われる．

- ネットワークトラフィックとネットワーク形態に応じたプロキシ網の構築が可能なので，時間帯によって起こるトラフィックの負荷分散を行える．Squidではドメインによって負荷分散を行う機能があるが，ある特定のサーバ