Peer-to-Peer Networkを用いたWeb Cacheの提案と実装

松本 義秀[†] 河合 栄治[††] 奥田 剛[†] 門林 雄基[†]
{yoshi-ma, eiji-ka, okuda, youki-k}@is.aist-nara.ac.jp

[†] 奈良先端科学技術大学院大学 情報科学研究科 [††] 科学技術振興事業団 さきがけ21

近年，IPv6の普及に伴いP2P (Peer-to-Peer) と呼ばれるクライアント同士の接続方式が注目されている．P2P方式はお互いのノードがサーバを介さずに自律分散的に働くことによって，サービスを提供することが可能である．本論文ではP2PをWeb Cacheへ適用する．従来のWeb Cacheは大規模なLAN環境に最適化され，小規模なLAN環境や個人ユーザにとって効果が薄いことが問題とされてきた．提案システムでは各ユーザのローカルキャッシュを共有し，P2Pネットワークを用いて検索を行う．このことにより，Web Cacheを小規模なLAN環境や個人ユーザにも適用することが可能であり，応答時間の高速化と，サーバサイドのトラフィックの削減が可能である．また，P2Pを用いることで管理コストが全くかからず，耐故障性に優れたシステムが構成できる．

Web Cache System Using Peer-to-Peer Network

Yoshihide Matsumoto[†] , Eiji Kawai[††] , Takeshi Okuda[†] , Youki Kadobayashi[†]

[†] Graduate School of Information Science, Nara Institute of Science and Technology
[††]PRESTO, Japan Science and Technology Corporation

Peer-to-Peer (P2P) network is completely decentralized application layer network. It does not require servers, and each message is routed autonomously on the network. Therefore, it can provide robust, fault tolerant, and management-free services. In this paper, we propose a Web cache system using P2P network. To increase the efficiency of a Web cache system, the number of the cache users is an important factor. Our system solves this issue by providing a large scale Web cache service network to every kind of users such as small LAN users and xDSL users.

1 はじめに

現在，インターネットにおける多くの情報システムはWebを用いて構築されている．そのため，Webのサービス品質は非常に重要であり，特に障害の発生は大きな問題となる．インターネットは，元来軍事目的のARPANETから発展したものであり，耐障害性を考慮した自律的な通信システムの構築を目標に設計された．パケット転送を制御するOSPF[8]等のプロトコルでは，障害発生時に代替パスへの切り換えを行うことにより，耐障害性の向上に貢献している．一方で，現在インターネット上のWebを含むほとんどのサービスは，クライアントサーバモデルを用いている．そのため，信頼性の高いサービスを提供する手段として，システムの分散化および多重化を行ない，最悪の場合でもfail softもしくはfail safeとなるように設計されてきた．このような背景のもと，近年P2P (Peer-to-Peer) と呼ばれるクライアント同士の対等な接続形態を用いた自律分散システムが注目されている．

P2Pは，サーバを介さないため耐障害性に優れ，サーバの管理コストがかからない等の様々な特徴を持つ．今回取り上げるGnutellaプロトコル[1]は，IP層ではなくアプリケーション層で耐障害性に優れたP2Pネットワーク(オーバーレイネットワーク)を構築する．さらには，このP2Pネットワークを用いて検索処理等を行なうことが可能である．本研究では，このような特徴を持つP2P技術を用いてWebサービスの品質を向上させる方式を提案する．

これまで提案されてきたWebにおけるサービス品質向上技術には，送信者側と受信者側の2つのアプローチがある．前者としては，レイヤ4/7スイッチやDNSラウンドロビン等を利用したサーバクラスタリングによる負荷分散や，CDN (Contents Delivery Network) の技術を用いたサーバ選択の最適化などが挙げられる．後者としては，サービス応答時間を短縮する技術としてWebキャッシュを用いる方法がある．両者とも，サービス規模に応じた設備投資および管理コストが必要となり，一般的には非常に高価である．

本提案システムでは，P2Pネットワークを構築することにより，送信者側ではCDNのような分散したコンテンツ配信モデルを実現し，同時に受信者側ではWebキャッシュ技術によるサービス応答時間の短縮を実現する．また，適用範囲が広いのも本技術の特長の1つである．送信者側では従来の技術のような高価な装置やWebサーバの改変は不要である．受信者側でも，キャッシュサーバの様に，効果を上げるためには一定のユーザー数が必要になるといった制約はない．また，P2Pネットワークの特性により，単一障害点がなく，システム全体を一元的に管理する必要がない．

表 1: Gnutella のメッセージ

メッセージ	内容
Ping	ネットワークでノードを探すために用いられる
Pong	Ping に対する応答．自分の IP 等を送信元に返す
Query	P2P ネットワークで検索要求を行なう
QueryHit	P2P ネットワークで検索結果を送信元に返す
Push	ファイアーウォールのノードに対しファイルの送信を要求する

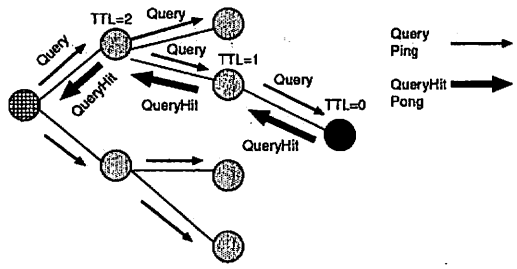

図 1: Gnutella の検索方法

2 Peer-To-Peer ネットワーク

P2P ネットワークは，各ノードがアプリケーションレベルで相互に接続を行って形成される．これはオーバーレイネットワークとも呼ばれ，TCP/IP 層の上に形成される仮想的なネットワークである．各ノードが資源を探す場合には，P2P ネットワークに接続し検索キーを送信する．その検索キーはルーティングされ，キーに合致するノードがレスポンスを返す．これにより，サーバのような特権的ノードを介さずに資源の発見が可能である．また，この P2P ネットワークはサーバが存在しないため，単一障害点がなく耐故障性にすぐれており，管理コストがかからないといった特徴を持つ．この P2P ネットワークのルーティングアルゴリズムとして Gnutella[1], Chord[2], CAN[3], Pastry[5], Tapestry[4] 等が提案されている．

今回使用する Gnutella のアルゴリズムは (仮想的に) 隣のノードに検索キーをフラッディングする．そのため，全てのノードに対し同時に検索が可能である．その反面，検索可能なノード数の増加に伴い消費帯域が指数的に増加するといったデメリットを併せ持つ．

2.1 Gnutella プロトコル

本節では P2P ネットワークを構築するために用いた Gnutella プロトコルについて述べる．

2.1.1 P2P ネットワークの構築

Gnutella においてノードを P2P ネットワークに接続するためには，まず最初の接続先 IP アドレスを 1 つ以上知っておく必要があるが，これは Gnutella ノードであればどれでもよい．P2P ネットワークへの接続は，TCP コネクションを確立した後，ハンドシェイクによって行われる．このハンドシェイクは，リモートノードに接続要求 "GNUTELLA CONNECT 0.4" を送り，応答 "GNUTELLA OK" を受信することにより完了する．そして，そのコネクションを用いて表 1 の 5 つのメッセージをやりとりを行なう．

各ノードは相互に複数の接続 (デフォルトの上限は 4 本) を維持することにより P2P ネットワークが構築される．最初の接続先へのコネクションを確立後，Ping メッセージを送ることで複数のコネクションを確立する．P2P ネットワーク上のノードは Ping メッセージを受信すると，他の接続先すべてに対してパケットを転送し，自分への接続数に余裕があれば Ping が来た方向に Pong メッセージを返す．Ping メッセージを定期的に送ることで接続可能なノードの IP アドレスを収集し保持する．各メッセージは，TTL (初期値=7) を持っており，各ノードで TTL はデクリメントされ TTL が 0 になるまで転送される．

Gnutella の P2P ネットワークは，相手からの切断や自分への接続が随時起こるため，可変なネットワークである．接続が切断された場合は，集めたアドレスリストからコネクションの制限数まで接続を行なう．そのため，常時複数の接続を維持し，耐故障性に優れたネットワークを構築することができる．

2.1.2 コンテンツ検索

検索者は接続しているノードに対し検索キーを含んだ Query メッセージを送信する．Query メッセージを受信したノードは自分のコンテンツを探すと共に，他の接続ノードへ Query を転送する．Query メッセージには GUID と呼ばれるメッセージ ID が含まれ，各ノードは転送すると同時に自分のルーティングテーブルにその GUID を登録し管理する．自分のキャッシュにヒットした場合には，自分のアドレスを含んだ QueryHit メッセージを Query が来た方向へ送信する．QueryHit が送信者へ返って来ることによって，コンテンツを保持しているノードのアドレスを知ることができる．コンテンツの取得においては P2P ネットワークを用いず，直接コンテンツを保持しているノードに接続し転送を行う．これらの一連の動作をまとめたものが図 1 である．

2.1.3 Gnutella のファイル転送

Gnutella プロトコルでは検索後，コンテンツ保持ノードに直接接続し，ファイル転送を行なう．ファイル転送のプロトコルは HTTP を用いている．また，PUSH メッセージを用いることでファイアーウォール内のノードからのファイルを転送も可能である．

2.2 従来の Web キャッシュサーバー

従来 Web の受信者は応答時間の高速化のために，キャッシュ(プロキシ) サーバが用いられてきた．代表的なものとして squid[7] 等の実装がある．しかし，Web キャッシュは大規模な LAN 環境を前提としており次のような問題がある．

1. ユーザ数がサーバ数に対し多すぎる場合，キャッシュサーバの処理がボトルネックとなり応答時間の低下につながる．また，それを改善するためには設備コストや管理コストが大きくなる．
2. ユーザ数が少ない場合にはヒット率が低下しキャッシュの効果が少ない．

この問題を解決するために P2P Web Cache の提案を行なう．

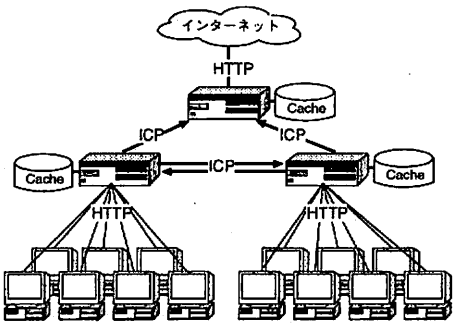

図 2: 分散キャッシュシステム

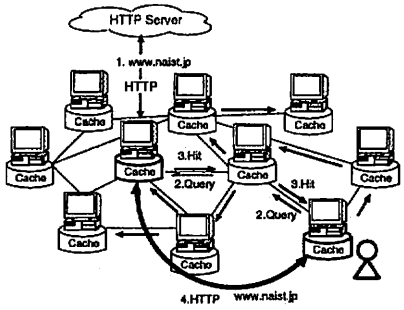

図 3: 提案システム

3 P2P Web Cache システム

以下を目標に P2P を用いた Web キャッシュシステム (P2P Web Cache) を提案する.

1. 小規模な LAN のユーザや個人ユーザにも動的に適応が可能である.
2. サーバ等の管理が不要で, コストがかからない.
3. 耐故障性に優れている.

従来の分散キャッシュシステム (図 2) では, サーバがすべてのキャッシュを保持し単一障害点となっていた. 提案する P2P Web Cache システム (図 3) では, すべてのノードがキャッシュを保持することで, キャッシュの位置を分散している. そして, 分散されたキャッシュを検索することにより, 仮想的な大容量ストレージの共有が可能となる. また, P2P ネットワークを用いることで, キャッシュの検索も完全に分散して行われる. そのため, 各ノードはすべて対等な関係であり, 耐故障性に優れており, 管理コストがかからないといった利点を持つ. 言い替えると, すべてのユーザがキャッシュサービスを提供し, 誰もがお互いのキャッシュを利用できるシステムといえる.

4 P2P Web Cache の設計

P2P Web Cache (図 4) は P2P ネットワーク部とコントロール部とプロキシ部の 3 つに分けられる. P2P ネットワーク部は, P2P ネットワークを構築し, メッセージのルーティング等を行なう. コントロール部は, P2P ネットワークの制御, 接続先, 最大待ち時間の判断等を行なう. プロキシ部は, ローカルキャッシュのメモリ, ディスク, ソケット, I/O の管理や高速化を行なう. また, 今回検索対象となるのは各ノードがローカルに保持しているキャッシュの URL である. CGI 等, 動的に生成されるコンテンツはキャッシュしない. 静的コンテンツのみを対象にし拡張子によって判断する.

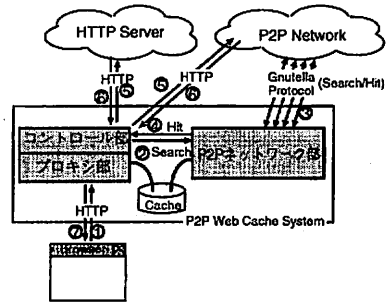

図 4: システム構成

4.1 システムの動作手順

P2P Web Cache は以下の手順で動作する

1. 各ユーザは自分のホストで P2P Web Cache をバックグラウンドで動かし, ブラウザのプロキシの設定で localhost アドレスを指定する. P2P Web Cache は, P2P ネットワークを構築すると同時にローカルプロキシの役割を果たしキャッシュを行なう.
2. 外部からの検索に応答し, ローカルに蓄積されるキャッシュを他人へ公開する.
3. 自分のブラウザからアクセスがありローカルキャッシュにない場合は, P2P ネットワークを用いて外部の公開されたキャッシュの検索 (図 3) を行なう. 場合によっては Web サーバへのアクセスを同時に行なう. 他ノードへの接続手法については次節で検討する.
4. P2P ネットワークの検索結果を考慮し, オリジナルの Web サーバに接続するかキャッシュを保持しているノードに接続するか判断する. この検索タイムアウト値と接続判定方法は次節で述べる.
5. 取得したコンテンツをブラウザへ転送すると同時に, Web サーバから取得したデータをローカルのキャッシュに保存する.

5 実装

実装では OS に Linux 及び FreeBSD を用い, C 言語を用いて開発を行った. プロキシ部と P2P ネットワーク部に関してはオープンソースの一部を利用することも考えられる. しかし, 修正箇所が多く, 修正コストも大きいと判断し, すべて新規に開発を行った. P2P プロトコルは Gnutella プロトコル Ver0.4 を拡張し実装を行った. Proxy 部に関しては HTTP/1.0 の仕様に基づいている.

5.1 コントロール部

5.1.1 接続判定方法

P2P ネットワークのトラフィックの流量と応答時間の関係は方式によって異なる. そこで, P2P ネットワークの検索とサーバに接続するタイミングとして 4 つの方式 (表 2) を比較した. 今回方式 1 を用いて実装を行なった. この方式 1 の場合, 検索タイムアウト値が Web 応答時間に大きく影響する. このタイムアウト値の決定方法については次節で述べる.

表 2: 接続判定方式の比較

手法	条件	帯域	速度
1	P2P で検索を行ない，検索結果を待ち，検索がミスヒットとされる場合のみに origin サーバへ接続する	◎	×
2	P2P での検索と origin サーバを同時に行なう．P2P で検索がヒットした場合は origin サーバの接続を切り，データの途中から転送を行なう	○	△
3	2 の状態で contents-length が大きい場合だけキャッシュノードへ接続を切替える	△	○
4	2 の状態でローカルプロキシにデータを一度ダウンロードし，速く完了した方のデータを転送する	×	◎

5.1.2 検索タイムアウト値の決定方法

個人ユーザのの不安定なネットワークにも適用するため，応答時間のタイムアウト値を推定することにより，大幅な応答時間の増大を防ぐことが可能である．

検索のタイムアウト値の推定方法は，TCP の Round Trip Time (RTT) タイムアウトアルゴリズムである RFC793[14] を用いて，P2P ネットワークの検索 RTT に適用した．RTT タイムアウト値は式 1 で表される．

$$RTO = SRTT + 4 \times DRTT \quad (1)$$

ここで，SRTT は RTT 平均値であり，DRTT は RTT の平均偏差である．平均偏差を加えることは，RTT を固定値や平均値に設定する場合と比べて，RTT の変動が大きい P2P の検索においても動的に適応することが可能である．

5.2 プロキシ部

5.2.1 キャッシュの重複度の削減

分散キャッシュシステムでは，複数のノードに同じコンテンツがキャッシュされ非効率的であるという問題がある．以下のアルゴリズムにより，同じ Web サーバのコンテンツは，単一ノードにできるだけ保存するようにし，キャッシュの重複度を削減した．

同じページ内に存在するファイルは全て同じノードにキャッシュされる可能性が非常に高く，連続して閲覧する可能性も高い．そこで，同一サーバへ連続でリクエストがある場合は，2 回目以降のファイルの取得は P2P ネットワークの検索を用いず，同一の P2P ノードへ直接 HTTP リクエストを送る．

各 P2P ノードは，HTTP のリクエストを受けると，自ノードにキャッシュがない場合でもサーバへ接続しコンテンツを取得する．同時にデータをリクエスト送信元に返し，自ノードにもキャッシュする．

検索者が他のノードのキャッシュを使う場合は，検索者のローカルキャッシュにデータを保存しない．これは，検索者にとってブラウザキャッシュがあるため不要であり，またネットワーク上でキャッシュの重複度をあげないためである．

これにより，キャッシュの重複度の削減だけではなく，P2P 検索回数の削減に伴う帯域の削減の効果と，2 回目以降の検索の高速化につながる．

5.3 P2P ネットワーク部

5.3.1 P2P ネットワークの最適化

Gnutella プロトコルで構成されるネットワークは，物理的な IP 層のネットワークとまったく関連なくランダムに構成される．そのため，隣のノードが海外でその隣が日本のノードであるような場合も考えられる．P2P Web Cache では距離が近いノードから順番にキャッシュの検索を行うことで，応答時間の高速化と，インターネット全体の無駄な帯域の削減が可能である．ここで言う，近いという意味は IP 層のホップ数, RTT, 遅延, 帯域幅等，広くとらえることができる，その他にも興味の同じ人同士が集まった方がよいということも考えられる．

本実装では簡単に測定できる IP 層の Time To Live(TTL) を測定することで，IP 層のホップ数が近いノードを優先的に接続するようにした．IPv4 の TTL の初期値は RFC1700 で定められており TTL=64 が推奨されている．しかしながら，各 OS の初期値はまったく異なっている．そのため，Gnutella の Hand Shake 時のプロトコルに以下の拡張を行ない，TTL の初期値を通知する．

```
GNUTELLA CONNECT 0.4
Defalut TTL=64
```

受信したノードは，この TTL の初期値と測定した値から相手までのホップ数を知ることができる．相手までのホップ数を比較することで，ホップ数の近いノードを優先的に接続を受け付ける．

このように，IP 層における近さ (TTL) の概念を P2P ネットワークに反映させることで，より局所性を考慮した P2P ネットワークを構築できると考えられる．

6 トラフィック分析

トラフィック分析に用いたキャッシュプランニングパラメータは文献 [11] の値を用いた．

6.1 ホップ数の変化に伴う P2P ネットワークの帯域

検索最大ホップ数 n，1 ノード当たり接続数 c(c≥2) の場合，到達可能ノード数は以下の式で表される．

$$Node(n, c) = c \sum_{k=0}^{n-1} (c-1)^k \quad (2)$$

図 5 は 1 ノード当たりの接続数 c=4 の場合の到達ノード数の変化を表しており，TTL の増加に伴いパケットの到達ノード数は指数的に増加する．Gnutella ではクエリパケットと帯域がユーザ数に比例して増加するため，現在の限られた帯域において大規模な検索は不可能でる．そのため，TTL の値は 7 となっており最大約 4000 ノードまで検索が可能である．

Web Cache に適用する場合は URL が検索キーとなり，平均 URL 長を 50byte と仮定すると Query のパケットサイズはヘッダサイズを含め合計 115byte となる．図 6 は常時接続数 c を Gnutella における標準値である 4 とし，1 人のユーザ当たり平均 5, 10, 15, 20[s] 間隔で別のホストにアクセスした場合の帯域の変化を表している．(同一サーバ内のコンテンツは同一キャッシュノードが保持するため，ブラウザのクリック間隔よりも長い時間間隔となる．)

P2P Web Cache では，図 6 より消費帯域を考慮し，実用的な値とし 100kbps 以下となる TTL=6 を選択した．そ

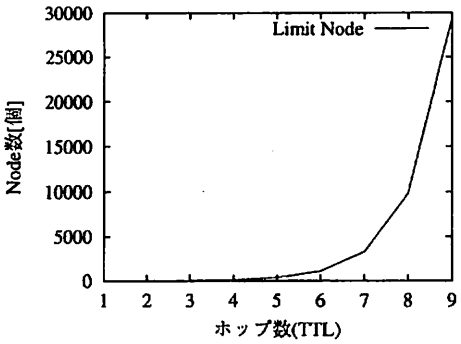

図 5: ホップ数の変化に対する検索可能ノード数

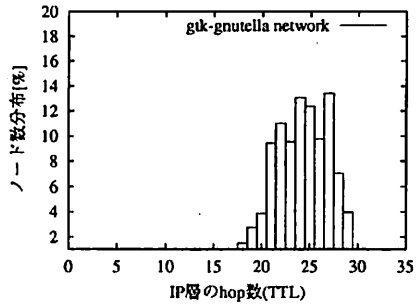

図 7: gnutella ネットワークにおける各ノードの TTL 分布

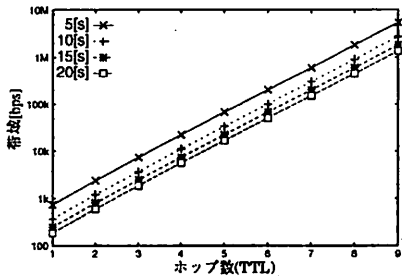

図 6: ホップ数の変化に対する Query メッセージ帯域

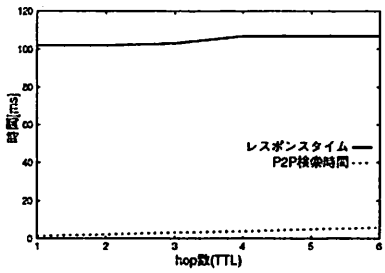

図 8: ホップ数の変化に対する検索時間と応答時間

のため，コネクション数 4，検索の最大ホップ数 6 の場合の最大到達可能ノード数は 1456 となる．しかし，コネクションが重複したり，4 本接続できないノードが存在するため，実際はこれよりも少ないノードへクエリが伝搬される．

6.2 HTTP のネットワーク帯域

ここで TTL=6 とした場合約 1000 ユーザのキャッシュを共有可能であると仮定する．ここで，ユーザが単一のキャッシュに接続する従来のモデルと，P2P キャッシュのモデルを考える．

1 人が平均 20 秒間隔でホストにアクセスすると仮定すると，180000 件/時のアクセスを生成する，コンテンツ平均サイズが 25kbyte の場合，従来のキャッシュの場合，10Mbit/s のトラフィックがキャッシュに集中する．実際にはキャッシュはサーバからコンテンツを取得しクライアントに送信するため，1000 ユーザの典型的なヒット率とされる 30%の場合では 17Mbps のトラフィックがキャッシュに集中する．

P2P Web Cache では 1000 台のノードに分散されるため，1/1000 である平均 10kbps の帯域が消費されることになる．

6.3 キャッシュサイズの比較

10Gbyte のキャッシュを利用したい場合，サーバ型キャッシュではそのまま 10Gbyte のストレージが必要となる．P2P Web Cache において TTL=6 の場合は約 1000 ノードのキャッシュが共有が可能であり，1 ユーザ当たり 1/1000 の 10Mbyte が必要となる．また，キャッシュサーバではコンテンツサイズの制限が存在するが，P2P Web Cache においてはユーザの設定次第で大規模コンテンツのキャッシュも可能である．

7 シミュレーション

最大ホップ数までの応答時間を測定するために，7 台の 100Base-TX の LAN に接続された CPU 800MHz，メモリ 512MB の PC を用いた．

7.1 P2P ネットワークの分布

gtk-gnutella[9] と BearShare[10] クライアントを用いて Gnutella ネットワークの測定を行なった．図 7 は自分から Gunutella ノードまでの IP 層のホップ数の分布を表している．15 ホップまでノードが存在しないのは，クライアントソフトが日本語ではないため日本のユーザが少ないからだと考えられる．しかしながらホップ数は分散されており，ホップ数の近いノードへ優先的に接続することで応答時間の改善が可能であると考えられる．また，同じ LAN にノードが複数存在する環境では，ホップ数が少ないため優先的に接続され，分散キャッシュモデルに近いモデルであると言える．そのため，LAN 内の高速なネットワークが生かせるだけでなく外部帯域の有効利用にもつながると考えられる．

7.2 検索時間と応答時間の比較

図 8 は実験環境において，コンテンツの平均サイズを 25kbyte とした場合のキャッシュの場所の検索時間と，HTTP のファイル転送を含めたレスポンスタイムを表している．Gnutella の検索において，検索クエリはノード間を最大 6 ホップ，検索応答を含めると最大 12 のマルチホップ検索を行う．本実装において，キャッシュの検索時間はホップ数に比例して増加するが，6 ホップの場合にお

いて 10ms 以下におさえられている．また，上記の検索時間を含めた，HTTP の転送完了まで応答時間は約 110ms となっている．以上より，LAN 環境において Gnutella のマルチホップ検索の影響は少ないことが判明した．その理由としては，検索時に TCP コネクションを新しく張るわけではないため，3Way Hand Shake のオーバーヘッドはかからないことが考えられる．つまり，LAN 環境においてマルチホップ検索時間は全体のレスポンスタイムに比べて十分に小さく，Web キャッシュへ適用することは有効である．

8 考察

完全分散型である gnutella プロトコルを用いることにより，P2P ネットワークを維持するための帯域が常に消費される．また，検索ノードを増やすために同時接続数や TTL を増やすことも，帯域が消費される原因となる．これは，アナログモデム等のユーザ自身だけではなく，そこへ接続するノードにとってもデメリットとなる．そのため，帯域の許容量を超えない TTL と，同時接続数の適切な設定が重要であると考える．

LAN 環境における検索時間は，遅延が小さいためマルチホップ検索の影響が小さく，実用的な値といえる．提案システムの応答時間の平均値は約 110ms であったが，同じ環境で Squid を用いた場合は平均 60ms であった．本研究では P2P ネットワークの Web サービスへの適用を目的としているため，Web キャッシュシステム自体の高速化技術に関する最適化を行っていない．今回の実装の性能低下の原因としてスレッドの生成時間やディスク I/O の時間等が考えられる．

9 まとめ

本論文では P2P ネットワークを用いた Web キャッシュシステムが，LAN 環境や個人ユーザの Web Cache に適用可能であることを示した．本システムは，従来の Web キャッシュサーバに比べ，管理コストが全くかからず耐故障性に優れている．

本システムを LAN 等の遅延の少ない環境で用いることにより，マルチホップ検索の影響が少なく，従来の Web Cache に近い性能が得られる．

個人のユーザの環境においては，Gnutella プロトコルを用いたうえで，検索タイムアウト値を推定することにより，耐故障性に優れたキャッシュの検索が可能である．また，広帯域の環境を持つユーザにとって Gnutella の帯域消費量は十分小さく，実用的なシステムであるといえる．

10 今後の課題

P2P Web Cache は人気の高い URL を保持するノードにリクエストが集中するという問題がある．また，本システムではネットワーク上のキャッシュの重複度を削減し，キャッシュの配置場所に偏りを持たせたが，どのくらいの性能の改善が見られるか分かっていない．また，キャッシュの新しさを維持するため，分散されたキャッシュの有効期限の設定方法を今後検討していく予定である．

11 関連研究

関連研究である Squirrel[6] では大規模 LAN 内の Web Cache のみを対象とし，Pastry[5] のルーティングアルゴリズムを用いている．そのため Gnutella[1] アルゴリズムに比べ検索時のネットワークの帯域を抑えることが可能である．

しかし，本システムは大規模 LAN のようなユーザ数の多い環境だけではなく，広帯域回線を持つ個人ユーザや小規模 LAN 環境のユーザをターゲットとしている．そのため各ユーザの PC は常時稼働しているわけではなくシャットダウン等により突然ネットワークから切断される可能性がある．また，各ノード間の RTT は様々であり，キャッシュ検索速度の高速化が求められる．

Squirrel を個人ユーザに適用させた場合，バースト的な負荷の集中や検索時間の長期化が起こる可能性がある．また，各ノードは他のノード持つコンテンツのインデックス情報を管理しているため，ネットワークへ頻繁に join/leave した場合，そのノードの管理するコンテンツがすべて削除されてしまうといった問題が起こる．本システムでは Gnutella プロトコルを用いることによりこれらの問題を解決している．

参考文献

[1] The Gnutella Protocol Specification v0.4, http://www.clip2.com

[2] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In Proceedings of ACM SIGCOMM, Aug 2001.

[3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In Proceedings ACM SIGCOMM, San Diego, CA, Aug 2001.

[4] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-resilient wide-area location and routing . Technical Report UCB//CSD-01-1141, U. C. Berkeley, Apr 2001.

[5] A. Rowstron and P.Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proceedings of the International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, Nov 2001.

[6] S.Lyer, A.Rowstron and P.Druschel. Squirrel: A decentralized peer-to-peer web cache. In the 21th ACM Symposium on Principles of Distributed Computing, July 2002.

[7] Squid Web Proxy Cache, http://www.squid-cache.org, March 1994.

[8] J. Moy, "OSPF Version 2", RFC1583, March 1994.

[9] Gtk-Gnutella, http://gtk-gnutella.sourceforge.net

[10] BearShare, http://www.bearshare.com

[11] アリ・ルオトネン『Web プロキシサーバ (パフォーマンスの最適化とセキュリティ)』, ピアソンエデュケーション, 1998.

[12] J. Reynolds,J. Postel "ASSIGNED NUMBERS", RFC1700 October 1994.

[13] T. Berners-Lee ed., "Hypertext Transfer Protocol - HTTP/1.0", RFC1945 May 1996.

[14] J. Postel, "Transmission Control Protocol", RFC793 Sept 1981.

[15] I. Cooper and J. Dilley, "Known HTTP Proxy/Caching Problems", RFC3143 June 2001.