LM-3

インデクスサーバを動的生成配置する P2P システム AmorphicNet

P2P System “AmorphicNet” with Dynamic Creation and Arrangement of Index Servers

内田良隆† 吉田紀彦‡ 楢崎修二†
Yoshitaka Uchida Norihiko Yoshida Shuji Narazaki

1. はじめに

現在インターネットの主流であるサーバ･クライアント･システムはサーバにアクセスが集中するという問題を抱えており，これを改善するものとして注目されているのが，コンテンツを各ノード（ピアともいう）に分散させる P2P（Peer to Peer）システムである．P2P を具体化したシステムとして，Napster[1]や Gnutella[2]などが有名である．

P2P ではコンテンツの所在を検索することが重要であり，そのために Napster では所在を一元管理するインデクスサーバを置き，そのサーバへのアクセス集中を招いている．一方で Gnutella では問合せをブロードキャストすることで所在検索を行い，ネットワークトラフィックの増大を招いている．

そこで本研究では，必要に応じてインデクスサーバを動的に生成し，しかも適宜再配置することで，負荷一極集中，トラフィック増大ともに回避する P2P システム「AmorphicNet」を設計し，小規模な実験でその効果を検証した．

2. 背景と関連システム事例

P2P ではコンテンツが各ノードに分散しているため，それを利用するには何らかの方法で所在を割り出す必要がある．現在，その方式には大きく 2 つがある．

(1) Hybrid P2P

Napster で導入された方式で，所在すなわちインデクス情報を一元的に保持するサーバを中央に置く．各ノードはそのサーバに問い合わせを行って，目的とするコンテンツの所在を得る．この方式では，検索数が多くなるとサーバの負荷が大きくなるというサーバ･クライアント･システムと同じ問題が発生する．

これを解決する試みとしては，インデクスサーバのミラーサーバを置く FastTrack[3]があり，KaZaA[4]などに用いられている．この方式では，処理能力の大きいノードを「スーパーノード」とし，インデクス情報を持たせる．

(2) Pure P2P

Gnutella で導入された方式で，全てのノードが対等に接続されている．各ノードは自身に接続している全ノードに問合せをブロードキャストし，それをネットワーク内で繰り返していくことで，目的とするコンテンツにいつかはたどり着くことを期待する．この方式では，ブロードキャストによって大きなトラフィックが発生する．TTL(Time To Live)という値で問合せノード範囲を制御できるが，小さくすると検索能力が抑えられ，逆に大きくするとトラフィックが増大するため，本質的な解決にはなっていない．また，問合せやコンテンツの転送を各ノードでキャッシュする方式も考えられている．

以上とは別に，コンテンツの意味カテゴリから検索空間を分割して検索を効率化しようとする SIO Net[5]も提案されている．

† 長崎大学 Nagasaki University
‡ 埼玉大学 Saitama University

3. インデクスサーバの動的生成配置

AmorphicNet は，Hybrid P2P での負荷集中，Pure P2P での過大なトラフィックをともに軽減するべく，我々のこれまでの成果[6,7]もふまえ，トラフィックに応じてインデクスサーバを動的に生成・配置する P2P システムである．Pure P2P と同じくインデクスサーバ（以下，単にサーバと略す）のない対等ノードからなるフラットなネットワークを初期状態とし，各ノードはトラフィックを監視しつつ，必要に応じて自らをサーバに格上げし，複数サーバ間で再構成を行っていく[8]．以下，システムの動作の概略を述べる．

(1) インデクス情報のキャッシュ

各ノードは自身を通過するパケットからインデクス情報を取り出してキャッシュし，他ノードからの問合せパケットに合致するインデクス情報をキャッシュ内に持つ場合はそれを返信する．キャッシュだけでもトラフィック軽減に寄与するのに加え，本システムではさらに，ノードを通過するトラフィックの近似にキャッシュの大きさを用いる．

(2) サーバノードへの格上げ

トラフィック（キャッシュ数）が一定値を超えたノードは，自らのキャッシュ内容をインデクス情報として持つサーバに自らを格上げする．サーバになったノードはそのことを近隣ノードに通知し，近隣ノードのキャッシュ内容を集める．サーバの存在を知ったノードは，以降の問合せを近隣へのブロードキャストからサーバへの１対１通信に切り替えるとともに，自分の近隣のノードに知らせる（図 1）．なお，サーバへの問合せで所在不明な場合は，それまで通りの問合せブロードキャストを行う．

(3) インデクスサーバの動的再構成

サーバのアクセスが増加して負荷が高まった場合，自身を利用しているノードの 1 つにサーバ格上げの要請を行うとともに，自らのインデクス情報を複製して渡す．これによってサーバを増やし，負荷分散を図る．これらサーバ群は親子関係に従って階層を構成する．親サーバは自身を利用しているノード群の一部に子サーバのことを知らせ，その知らせを受けたノードは以降の問合せを親サーバから子サーバに切り替える（図 2）．なお，アクセスが

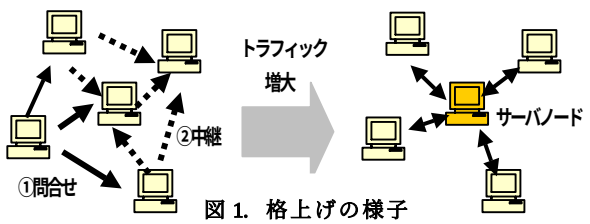

図 1. 格上げの様子

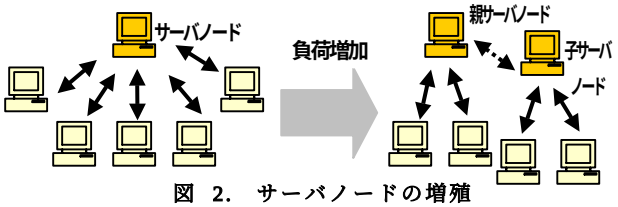

図 2. サーバノードの増殖

減少して複数サーバの必要が無くなった場合は，自らを平ノードに格下げする．

(4) 問合せの転送と回答の中継

サーバは，問合せに合致するインデクス情報を自ら保持しておらず，かつサーバ木のトップに位置する根サーバでない場合は，その問合せを親サーバに転送し，親からの回答を中継する．なお，根サーバ以外のサーバが持つインデクス情報は一定の生存期間（TTL．ただし，Gnutella の TTL とは異なる）を持ち，それを超えたインデクス情報は消失する．

(5) サーバのインデクス情報の更新

コンテンツの更新や追加があったノードは，そのことをサーバに知らせる必要がある．ノードは，サーバの存在を知らないならば，何もしない．知っているならば，そのサーバに更新・追加を知らせる．知らされたサーバは，自らの属するサーバ木の根サーバにもそれを知らせる．この更新・追加は(4)の処理によって，いずれサーバ木全体に反映する．

上記の方式では更新がサーバ木全体に反映するまではサーバ間の一貫性が損なわれるが，通常のウェブ検索にもみられるように，インデクス情報には厳密な一貫性が要求されないため，一貫性管理の負荷を抑えることを優先している．

4. 実装と評価

本研究では AmorphicNet を Java で実装し，まずローカルな WS ネットワークで実験を行った．ノード数 20，サーバ格上げの契機をキャッシュ数 16，サーバ増殖の契機をノード接続数 7 とし，問合せは各ノードでランダムに生成した．最初のフラットなネットワークの形態の一例を図３に，そこからある時間が経過した後にサーバが生成された様子を図４に示す．ネットワーク内の総パケット数，および検索に要するホップ数の平均値について，時間ごとの推移を図５と６に示す．なお，キャッシュを導入すると，コンテンツを保持するノードよりも遠くにあってキャッシュを保持するノードも応答を返すようになるため，平均ホップ数が増えることがある．

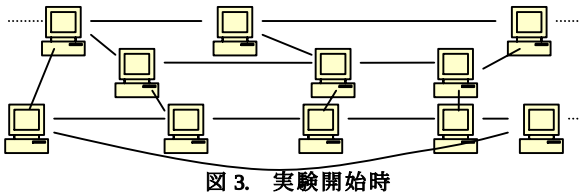

図 3. 実験開始時

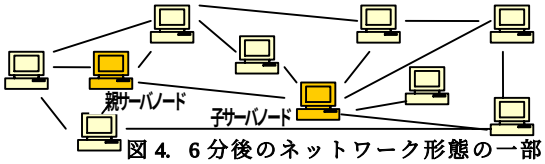

図 4. 6 分後のネットワーク形態の一部

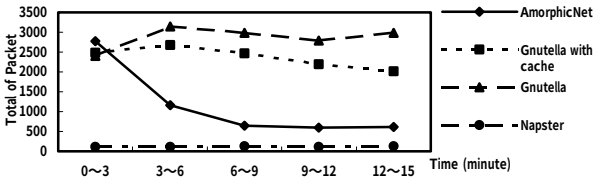

図 5. 総パケット数の推移

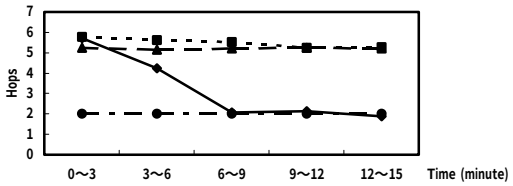

図 6. 平均ホップ数の推移

このように，AmorphicNet では時間の経過に従って Pure P2P から総パケット数も平均ホップ数も減少していくことを確認した．

5. まとめ

本研究で設計・実装した AmorphicNet では，コンテンツの所在を表すインデクス情報を保持するサーバを動的に生成・配置することによって，従来の Pure P2P に比べてネットワークトラフィックの削減，検索効率の向上を達成した．サーバの配置はトラフィックに応じて動的に定まるため，事前に定めておく必要がないのも，本システムの利点である．

なお，負荷分散に関して受付けパケット数が最大のノードの推移についても測定を行ったが，実験が小規模であったため，負荷が完全に分散している Pure P2P の初期状態でも検策パケットがネットワーク内に「充満」してしまい，Hybrid P2P との有意な差が得られなかった．一極サーバを排した負荷分散の効果は，ネットワークが大きくなるほど顕著になるはずである．

現在の実装では，サーバになるためのコードをすべてのノードが最初から内包しており，無駄が多い．そこで，インデクスサーバをモバイルスレッドとして構成し，サーバ格上げ時や子サーバ生成時における動的な移送や複製生成，サーバ切断時や障害時への対応に活用すべく，新たな実装を進めている．また，処理能力やネットワーク容量などに応じたサーバ候補ノードの選別，JXTA[9]の検索プロトコルへの応用などについても，検討を進めている．大規模環境での挙動解析も行う予定である．

参考文献

[1] http://napster.com/
[2] http://www.gnutelliums.com/
[3] http://www.fasttrack.nu/
[4] http://www.kazaa.com/
[5] 星合他，意味情報ネットワークアーキテクチャ，NTT R&D，Vol.50, pp.157-164 (2001).
[6] S.Narazaki, H.Yamamura and N.Yoshida, "Strategies for Selecting Communication Structures in Cooperative Search", Int'l J. of Cooperative Information Systems, Vol.4, No.4, pp.405-422 (1995)
[7] T.Shimokawa, S.Narazaki, H.Hamachi and N.Yoshida, "Dynamic Multi-Server Reconfiguration Using Meta-Level Computation in Distributed Information Sharing", Proc. Conf. on Systems, Cybernetics and Informatics 1999, Vol.5, pp.274-281 (1999)
[8] 内田，ピア・ツー・ピア・システムにおけるインデクス情報管理の動的再構成．長崎大学卒業論文 (2002)
[9] http://www.jxta.org/