冗長メタサーバによる高信頼分散サーチエンジン

佐藤 永欣　宇田川　稔　上原 稔　酒井 義文　森 秀樹
E-mail: {jju,ti980039}@ds.cs.toyo.ac.jp, {uehara,sakai,mori}@cs.toyo.ac.jp
東洋大学工学部情報工学科

集中型アーキテクチャに基づくサーチエンジンでは新鮮な情報検索が困難である。そこで我々は分散型アーキテクチャに基づく協調サーチエンジン (Cooperative Search Engine, CSE) を開発した。分散型アーキテクチャは耐故障性に優れる。しかし、CSE では、1台のみのメタサーバが single point of failure となってしまう。そこで、本文では、メタサーバの冗長度を増し、信頼性を向上させる。CSE では更新時の遅延は隠蔽できるため深さ優先による放送が最善の方式であった。

A Reliable Distributed Search Engine by Redundant Meta Servers

Nobuyoshi Sato, Minoru Udagawa, Minoru Uehara, Yoshifumi Sakai, Hideki Mori
Department of Information and Computer Sciences, Toyo University

It is difficult for centralized search engines to retrieve fresh information. So, we have developed Cooperative Search Engine(CSE), which is based on distributed architecture. Essentially, a distributed search engine has an advantage of fault tolerance. In CSE, however, only one meta server is a single point of failure. In this paper, we describe about reliable architecture based on redundant meta servers. Since CSE can hide the communication latency at updating, we conclude that the depth first routing is the best way in order to realize broadcast of updating.

1 はじめに

近年、ナレッジマネジメントやデータマイニングのために組織内情報検索が重要となってきた。組織内情報検索では新鮮な情報検索が必要である。特にビジネス分野では必要な情報が得られないと機会を失うことになり、これは重大なミスとされる。また、大域的に新鮮な情報を検索できれば機会発見に役立つ。情報検索には一般的にサーチエンジンが用いられている。しかし、組織の規模が大きくなると集中型サーチエンジンでは新鮮な情報検索が困難となる。これは、集中型サーチエンジンでは、ロボットで文書を収集し、インデックスを更新するまでに長い時間がかかるためである。したがって、新鮮な情報検索には分散して文書収集、インデックスの作成、更新が可能な分散サーチエンジンが適している。

そこで、我々は、分散型アーキテクチャに基づく協調サーチエンジン (Cooperative Search Engine, CSE) を開発した [1][6]。CSE は、各 Web サーバに配置された局所サーチエンジンをメタサーバで統合した大域的サーチエンジンである。CSE はボトムアップで文書収集、インデックス作成等の更新作業を行う。このため、CSE は更新に関してスケーラブルであり、規模にかかわらず短時間でインデックスを更新できる。

しかし、一般に分散サーチエンジンは検索時に相互に通信する必要があり、通信により様々な遅延が発生してしまうため、大規模化は困難と考えられていた。CSE も同様に検索時に遅延が生じる。しかし、後述する各種の高速化技法を採用することで検索時にもある程度のスケーラビリティを実現することができた。

また、一般に分散システムは本質的に耐故障性に優れると言われる。これは、分散システムは複数の構成要素からなるため、単一要素の故障をマスクすることができるからである。しかし、CSE では、1台しかないメタサーバが single point of failure となるため、信頼性が低かった。そこで、本論文では、メタサーバの冗長度を増すことで、信頼性を向上させる手法を提案する。

本文の構成は以下の通りである。2章では、協調サーチエンジンについて述べる。3章では、提案する耐故障性アーキテクチャについて述べる。4章では、評価

について述べる。5章で、関連研究と比較し、最後に結論を述べる。

2 協調サーチエンジン

CSEはFig.1に示されるような以下の部品から構成される。

- Location Server (LS): LSはFK (Forward Knowledge) を一元管理する。LSはFKを用いてクエリに基づくサイト選択（後述）を行う。LSはサイト選択キャッシュ(Site selection Cache, SC) を持つ。
- Cache Server (CS): CSは、サイト選択の結果と検索結果をキャッシュするサーバである。検索結果をキャッシュすることで継続検索（次の10件の検索）を実現する。また、CSは後述のLMSEを並列に呼び出し、並列検索を行う。CSは検索結果キャッシュ(Retrieval Cache, RC) とサイト選択キャッシュを持つ。CSのサイト選択キャッシュはLSのサイト選択キャッシュの部分的、不完全なコピーである。CSのサイト選択キャッシュに検索によって生じた変化はLSのサイト選択キャッシュに反映される。
- Local Meta Search Engine (LMSE): LMSEは、ユーザからの要求を受け付けCSに転送したり（Fig.1のUser I/F）、後述のLSEを呼び出し局所的な検索をしたり（Fig.1のEngine I/F）する。LSEの差異を吸収するメタサーチエンジンである。
- Local Search Engine(LSE): LSEは局所的な文書収集（Fig.1のGatherer）、インデックス作成（Fig.1のIndexer）、検索（Fig.1のEngine）を行う。

更新時における各構成要素の振る舞いは以下の通りである。

1. LMSEはLSEを用いて文書を収集する。
2. LMSEはLSEを用いてインデックスを更新する。
3. LMSEはLSにメタインデックス（語の集合、全文書数、語を含む文書数とその最高スコア）を送信する。

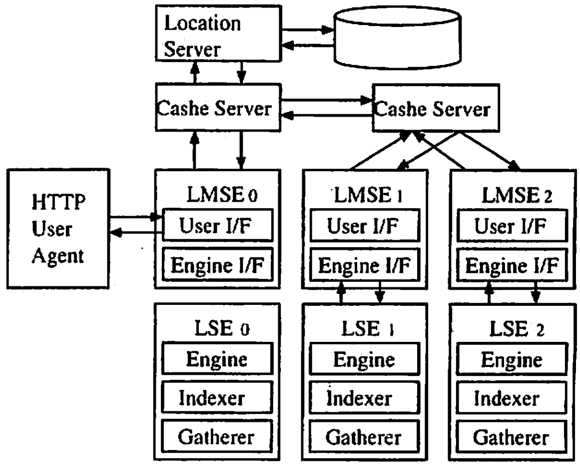

Fig. 1. CSEの概要

CSEでは、NFS等のファイルシステム経由で直接収集できる文書はファイルシステムを通して直接収集し、クラスタを用いて並列にインデックス作成を行う事が可能である。また、直接収集できない文書はWebサーバに用意した文書収集用CGIを用いて一括転送する事も可能である。これらが使用できない場合のみ、通常のロボットによる収集が用いられる。この結果、東洋大学の事例では約1分で全文書のインデックスを更新できた。

一方、検索時における各構成要素の振る舞いは以下の通りである。

1. ユーザはブラウザにより身近な $LMSE_0$ に検索を依頼する。
2. $LMSE_0$ はCSに検索を依頼する。
3. もしCSのRCに検索結果の次ページまでキャッシュされていたら8へ。もし、該当ページまでしかキャッシュされていなければ5へ。
4. CSはSCにサイト選択結果がキャッシュされていたら5へ。そうでなければ、LSにクエリに基づくサイト検索を依頼し、スコアの降順に並んだサイトのリストと各語の *idf* を受信する。
5. CSは次ページまでの項目を埋めるのに必要なだけリスト上位からサイト $LMSE_i$ を選び、検索要求を並行に送信する。

6. $LMSE_i$ はLSEに検索を依頼し、検索結果と次の最高スコアをCSに返す。

7. CSは結果をマージし、サイトのリストをスコア順に並び替える。

8. CSは結果を $LMSE_0$ へ返す。

9. $LMSE_0$ は結果をHTMLに整形し、ユーザへ返す。

CSEでは、検索時に通信による様々な遅延のため応答時間が長くなる。そこで、一回あたりの通信量を極力減らす、不要な通信は行わない等、種々の高速化技法が提案されている。

- クエリに基づくサイト選択 (QbSS) [2]。CSEは積 (AND)、和 (OR)、差 (NOT) の論理検索をサポートしている。ここで、クエリ A、B に対する選択サイトを S_A、S_B とすると、"A and B"、"A or B"、"A not B" はそれぞれ $S_A \cap S_B$、$S_A \cup S_B$、S_A となる。これによりサイト集合を1/10に絞り込むことができた。

- 先読みキャッシュ[3]。「次の10件」をバックグラウンドで先読みすることで応答時間を短縮する。これにより2ページ以降の連続した「次の10件」検索は常にキャッシュにヒットする。「次の10件」検索とは、検索結果全てを一度に表示せずに、一般的なサーチエンジンのように1ページに10件程度ずつ表示することを指す。

- スコアに基づくサイト選択 (SbSS)[4]。各サイトのクエリに対する最高スコアを収集し、「次の10件」検索時にクエリを送信するサイトを多くとも10サイトに限定する。これにより論理検索の1ページを除く連続した「次の10件」検索では、規模に依存せず一定の応答時間を実現した。

- 大域的共有キャッシュ[5]。異なるLMSEが異なるCSに同じクエリを送ったとき、LSは先にクエリを検索したCSを紹介し、CS同士でキャッシュを転送することで検索を抑制し、応答時間を改善する。

- 永続的キャッシュ[7]。更新間隔の短いCSEではキャッシュを長く用いることができない。そこで、更新後に再検索を行うことでキャッシュの寿命を永続的にする。これにより一度検索された (論理型) クエリの応答時間も規模に依存せず一定となった。

以上の技法は以下のような場合に適用できる。第一に、検索式が単一キーワードであるかOR演算子のみを含む場合、または2ページ以降の「次の10件」検索では、スコアに基づくサイト選択により一定の応答時間を実現できる。第二に、更新前にクエリが検索されていれば、永続的キャッシュにより、一定の応答時間を実現できる。第三に、更新後にクエリが検索されていれば、大域的共有キャッシュにより即座に応答可能である。最後に、それ以外の場合はクエリに基づくサイト選択を行う。クエリに基づくサイト選択による応答時間は検索対象サイトの数に依存し、検索対象サイトの数は一般にCSEの規模に依存する。

3 耐故障性アーキテクチャ

はじめに本文における故障をノードとリンクの停止故障と定める。LSはCSEのsingle point of failureである。そこで、LSを冗長化する。冗長化には多数決方式やプライマリ・バックアップ方式などがある。多数決方式では通信が増え、プライマリ・バックアップ方式ではプライマリに負荷が集中する。冗長化されたLSのメタ情報はいずれも等しいとすると多数決は不要である。そこでP2Pに基づき更新メッセージを放送することで各LSの一貫性を維持する。これにより検索時の通信遅延を抑制し、かつ負荷を分散できる。

LS以外の構成要素については以下の通りである。CSに関しては、少なくとも1台故障していないCSが存在すれば検索可能なので、冗長化する必要はない。LMSEはWebサーバに依存するため冗長化は困難である。万一、一部のLMSEが停止しても、そのLMSEが所有する文書の検索が出来なくなるものの、全体が停止するわけではない。LS、CSおよびLMSEが互いを参照する様子をFig.2に示す。

LMSEはLSの参照を複数持つ。更新時に複数のLSから任意のひとつを選択し、更新メッセージを送信する。また、検索時にLMSEは任意のLSから任意のCSの情報を教えてもらい、任意のCS (ネットワーク的に近いCSが優先される) を選択し、クエリを送信する。このように、複数の参照に対して2種類のマルチキャスト通信を用いる。ひとつはanycastで、もうひとつはmulticastである。これらはunicastの組み合わ

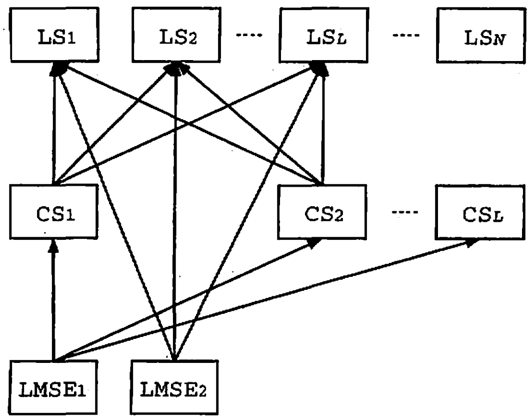

Fig. 2. 要素間の参照関係

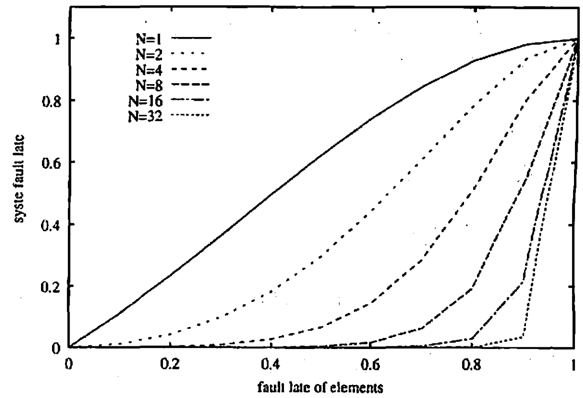

Fig. 3. $M = 2N$ における故障率

せで実現される。

マルチキャスト通信のスケーラビリティは、LS の数、ランク（すなわちリンク数）、ルーティングに依存する。ルーティングは幅優先と深さ優先に大別される。さらに幅優先は TTL(Time-To-Live) に依存する。そこで、以下の 3 方式を比較した。

Depth First Routing (DF) DF では、訪問ノードのリストを含むメッセージを受信したノードが未訪問のノードで転送する。DF はノードが少ないとき適している。

Breadth First Routing, TTL=0 (BF0) BF0 では、各ノードが直接その他の全ノードへ転送する。BF0 はリンク故障がないとき最善である。

Breadth First Routing, TTL=L (BFL) BFL では、各ノードは受信したメッセージの TTL を 1 減じ、TTL > 0 なら隣接ノードでメッセージを転送する。BFL はリンク故障にも耐えるが、メッセージ数が指数的に増加する。

ここで、上記方式と P2P を比較する。典型的な P2P でも TTL による幅優先のルーティングが行われている。したがって、BFL は P2P と考えることができる。

リンク故障によりネットワークが切断されると、放送を用いてもメタインデックスを共有できなくなる。この問題に対処するには以下のような方法が考えられる。LMSE は更新情報を直接 LS に送信するのではなく、いったん CS に送る。CS は複数の LMSE からの更新情報をまとめて LS へ送る。LS は他の LS へ放送する。ここで、未配達の LS が検出されたら、他の CS を経由して未配達の LS へ送信しようとする。CS は LS より多いため転送が成功する確率が高い。

4 評価

ノードのみが故障する場合、LS と CS がそれぞれ少なくとも 1 つ正常に動作していれば、システム全体は停止しない。したがって、システムの故障率 F は以下の式で表せる。

$$F = f^N + f^M - f^{N+M}$$

ここで、f は要素 (LS,CS) の故障率、N と M はそれぞれ LS と CS の数である。Fig.3 に $M = 2N$ における関係を示す。リンク故障がなければ結果はルーティングに依存しない。故障率 0.9 のときシステム故障率を 0.2 以下にするには $N \geq 16$ でよい。

次に、リンクのみが故障する場合、$N = 32$ におけるリンクの故障率に対するメッセージの到着率の関係を Fig.4 に示す。ここで、到着率とは放送メッセージを受信したノードの割合のことである。BF0 は他に比べて到着率が低い。にメッセージ数を Fig.5 に示す。BF32 のメッセージ数は他に比べて大きい。したがって、DF が最善の戦略である。前章において BFL は P2P であると述べたが、通常 P2P の TTL は 7 程度である。よって BF32(TTL=32) は P2P より大きな TTL を用いている。これは信頼性を高めるためであるが、それによって無駄なメッセージが発生していると考えられる。よって実際の P2P はもう少し効率がよい。しかし、決して BF0 や DF と同等にはならない。

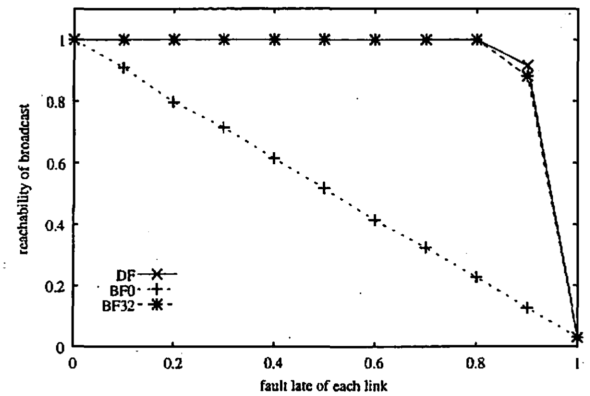

Fig. 4. リンク故障と到着率 ($N = 32$)

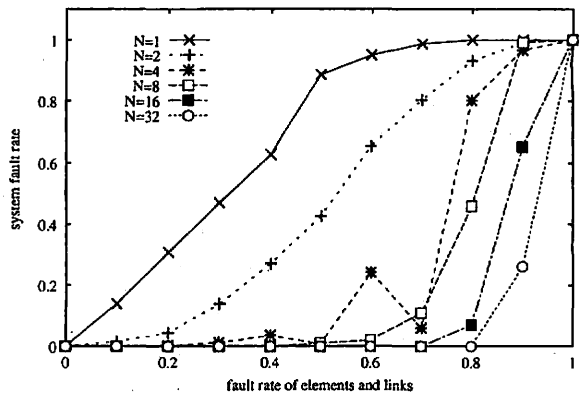

Fig. 6. ノード及びリンク故障に対する故障率

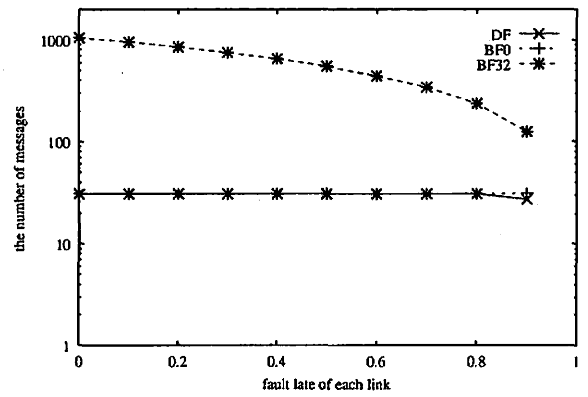

Fig. 5. リンク故障とメッセージ数 ($N = 32$)

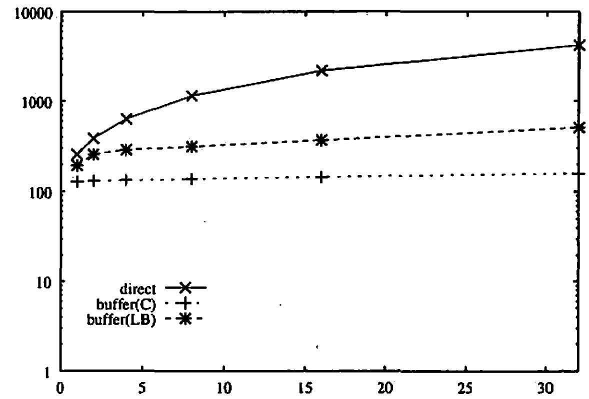

Fig. 7. メッセージ数 ($\#CS = 64$, $\#LMSE = 128$)

次に、ノードとリンクの両方が等しい率で故障する場合の故障率を Fig.6 に示す。故障率 0.9 のときシステム故障率を 0.25 以下にするには $N \geq 32$ でよい。これはリンク故障のため Fig.3 より悪い。前章において CS 経由で他の LS へ放送メッセージを転送する方式を提案したが、他の CS を知るために LS へ問い合わせることができないためその効果は低い。ノードとリンクの両方が故障する場合は、通信可能な CS と LS の組が少なくとも 1 つは必要である。

CS 経由の更新は、信頼性の向上にはあまり効果がないが、メッセージ数を減少させることには効果がある。LMSE は更新メッセージを CS に送り、CS は複数の LMSE から受信した更新情報を LS へまとめて転送する。これによりメッセージ数を削減することができる。Fig.7 に CS の数が 64 のときと、128 の場合における更新メッセージ数と LS 数の関係を示す。ここで、direct は LMSE が直接 LS に更新メッセージを送信する方式である。また、buffer(C) は更新メッセージを 1 つの CS に集めて一度に LS に送信する方式である。Buffer(LB) は更新メッセージを複数の CS に均等に集める。Direct 方式のメッセージ数は $(N+1) * \#LMSE$ に等しい。Buffer(C) 方式のメッセージ数は $N + \#LMSE$ に等しい。Buffer(LB) 方式のメッセージ数は buffer(C) より大きいが、direct よりずっと小さい。Buffer(C) は負荷が大きいためスケーラブルでないが、Buffer(LB) は負荷が均一なのでスケーラブルである。

5 関連研究

分散システムにおいて信頼のできるグループ通信を行う研究は長く行われてきた。代表的なものとして ISIS[8] がある。ISIS では、2 相コミットに基づく ABCAST と因果関係に基づき仮想同期を行う CBCAST

などがサポートされた。ISIS はグループの規模が比較的小さい場合に有効である。また、ISIS は原子性が必要な応用に適する。CSE は原子性を要求しないため、より効率的な手段を採用することができる。

グループの規模が大きくなると P2P の方式が有効となる。純粋な P2P には Gnutella[9] などがある。しかし、これらのシステムで放送を行うと、膨大なメッセージが発生する。そのため効率的なルーティングによりメッセージ数を削減した P2P がいくつか提案されている [10][11]。しかし、JXTA[10] ではハブを導入することによりメッセージ数を削減しているものの基本的な原理は Gnutella と変わらない。P-Grid[11] は、P2P 上に 2 進木を構成することで検索や放送を効率よく行うことができるが、信頼性は 2 進木の冗長度に依存する。本文で提案した方式はメッセージ数の点でこれらの P2P より効率がよい。CSE では、P2P に比べると少数のサーバしか存在しないため、深さ優先では TTL による制限を行わなかった。しかし、より大規模なネットワークでは TTL による制限を行うことで対応することが可能と考えられる。また、一般に P2P では遅延を予想することが困難であるため、リアルタイム性を重視する CSE には適さない。CSE では、遅延を隠蔽できる更新時のみ放送を行う。

6 まとめ

本論文では、CSE の single point of failure であるメタサーバ LS の冗長度を高めることで耐故障性を向上した。これにより、比較的少数（100 未満）のメタサーバでも十分な信頼性が得られることがわかった。本論文における評価はシミュレーションによるものであるため、実証性の点で不十分である。今後、実機を用いた評価実験を行うことを計画している。また、LS の増加はスケーラビリティの改善に貢献すると考えられるため、LS の数とスケーラビリティとの関係を考察する。

謝辞

本研究は東洋大学特別研究「モバイルエージェントによる Web ロボットの開発」および科学研究費補助金（若手研究 (B)731「新鮮な情報検索のためのスケーラブルな分散型サーチエンジン」 課題番号 14780242）の助成を受けて行われた。関係者に感謝する。

参考文献

[1] 佐藤 永欣，上原 稔，酒井 義文，森 秀樹，“最新情報の検索のための分散型サーチエンジン”，情報処理学会論文誌， 第 43 巻，第 2 号，pp.321-331，情報処理学会 (2002)

[2] 酒井義文，上原 稔，佐藤 永欣，森 秀樹，“協調サーチエンジンにおける検索クエリの最適な単調化”，マルチメディア・分散・協調とモーバイル (DICOMO'2001) シンポジウム論文集，Vol.2001, No.7, ISSN1344-0640, pp.453-458 (2001)

[3] 佐藤永欣，山本 崇，西田喜裕，上原 稔，森 秀樹，“協調サーチエンジンにおける継続検索のための先読みキャッシュ方式”，情報処理学会マルチメディア通信と分散処理研究会ワークショップ論文集，Vol.2000, No. 15, ISSN1344-0640，pp.205-210 (2000)

[4] 佐藤永欣，上原 稔，酒井義文，森 秀樹，“協調サーチエンジンにおけるスコアに基づくサイト選択”，マルチメディア・分散・協調とモーバイル (DICOMO'2001) シンポジウム論文集，Vol.2001, No.7, ISSN1344-0640, pp.465-470 (2001)

[5] 佐藤永欣，上原 稔，酒井義文，森 秀樹，“協調サーチエンジンにおける大域的共有キャッシュ”，情報処理学会マルチメディア通信と分散処理研究会ワークショップ論文集，Vol.2001, No.13, ISSN1344-0640，pp.219-224 (2001)

[6] Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, “On Updating in Very Short Time by Distributed Search Engines”, In proc. of The 2002 International Symposium on Applications and the Internet (SAINT 2002), ISBN 0-7695-1447-2, pp.176-183 (2002)

[7] Nobuyoshi Sato, Minoru Uehara, Yoshifumi Sakai, Hideki Mori, “Persistent Cache in Cooperative Search Engine”, In proc of MNSA'02 (2002) (to be appeared)

[8] Birman, K.P., “The Process Group Approach to Reliable Distributed Computing”, Commun. of the ACM, vol. 36, pp.36-53, Dec. 1993

[9] Matei Ripeanu, Adriana Iamnitchi, Ian Foster, “Mapping the Gnutella Network”, IEEE Internet Computing, Vol.6, No.1, pp.51-57 (2002)

[10] Steve Waterhouse, David M. Doolin, Gene Kan, Yaroslav Faybishenko, “Distributed Search in P2P Networks”, IEEE Internet Computing, Vol.6, No.1, pp.68-72 (2002)

[11] Karl Aberer, Magdalena Punceva, Manfred Hauswirth, Roman Schmidt, “Improving Data Access in P2P Systems”, IEEE Internet Computing, Vol.6, No.1, pp.58-67 (2002)