実時間検索の実現に向けた分散インデックス法の研究

藤　田　　聡　　広島大学大学院工学研究院教授

1 研究の目的

ユーザがネットワーク上で更新した情報をほかのユーザがリアルタイムに検索することができる**ネットワーク検索の実時間化**が、近年高い注目を集めている。検索処理を実時間化することによって、各ユーザは他のユーザが発信した情報をタイムリーに取得することが可能となり、ネットワーク上の情報共有をより緊密なものにすることができる。また実時間情報検索は、防犯システムにおける不審者情報の共有や広告配信分野などへの応用も考えられており、その実用上のメリットとインパクトには非常に大きいものがある。現在もっとも幅広く利用されているGoogleやYahoo!のような従来型の検索エンジンでは、ウェブ上の文書をクローラによって数日から数週間程度の時間をかけて一か所に集め、集中サーバがその参照構造を適切に解析した上でデータ構造として蓄積し、ユーザからのクエリに対して迅速に対応するという方法がとられている。したがって従来型モデルのままで検索の実時間化を実現しようとすると、クローラによる情報収集の頻度を極端に高くしなくてはならず、ネットワーク負荷の増大が避けられない。この問題を解決するため本研究では、蓄積されるデータをネットワーク上に分散的に保持し、各文書に対してなされた更新がそのデータ構造に低いコストで反映されるようなアプローチをとる。以下ではそのようなデータ構造のことを**分散インデックス**と呼ぶ。本研究の目的は、分散インデックスをコンピュータネットワーク上に効率よく実現するための基盤技術の開発をおこない、情報検索ソフトウェアの実装を通してその効果を実証的に評価していくことである。

本研究課題で新たに提案した手法では､基本となるP2Pアーキテクチャとして3層構造をもった階層型P2Pを想定している。提案手法の主要な貢献は以下の2点である：第1点は、ピア群によって保持されているファイルをスーパーピアたちに動的に割り当てる方法を提供することである。各スーパーピアは、それぞれ通常の検索エンジンにおけるサーバとほぼ同様の働きをし、割り当てられたファイルのインデックス情報の維持・管理をおこなう。第2点は、与えられたクエリからそのクエリに関連するインデックス情報を保持しているスーパーピアを効率よく特定し、そのスーパーピアに向けてクエリ転送をおこなう機能を提供することである。ファイルのスーパーピアへの効果的な割り当てを実現するため、提案手法では、YouTubeなど多くの検索システムで用いられているタグの概念を利用した。具体的には、タグ集合上で定義されるタグの優先度列(priority sequence)という考え方を新たに導入し、その機能をクエリの高速な転送に応用した。提案手法の効果はセンサー情報をインターネット上でグローバルに共有する分散システムの試作を通して実証的に評価される。

2 研究方法

研究の基本方針は以下の通りである。本研究は、① 我々がこれまでに提案してきた分散データ構造の実システム上への実装、② 実時間検索を実現するための新しい機能の提案と実装、③ 実装されたシステムの性能評価と問題点の洗い出し、の３つの段階から構成されている。類似する関連研究との相違点は、我々が実装しようとしている手法が集中型ではなく**分散的に保存されたデータ構造**に基づいているという点である。提案手法を用いることによって、通常の（クライアント/サーバ型を含む）集中型の実装と比べて、情報の更新コストを小さくできることが期待される。

3 研究結果

3-1 システムモデル

想定する3層構造の階層型P2Pの詳細は以下の通りである。第一層はサーバ群から、第二層はスーパーピア群から構成され、一般のピアは最下層の第三層に位置する。以下では、第三層に属する一般のピアのことをユーザピア(UP)と呼ぶ。各UPは、ネットワーク全体で共有される情報をファイル等の形式で保持している。

また以下では簡単のため、サーバはシステム内に唯一存在するものとし、これをシンボルCで表現し(実際にはサーバは複数存在してもよいし、その機能をスーパーピアに割り当てて実現することも可能である)、スーパーピアをS_i,S_jなどと表現する。UPのもつファイルは、内容の類似性や地理的な近さなどによって適切にグループ化される。各グループ(クラスタ)はそれぞれ第二層のスーパーピアに対応付けられ、各ファイルとUPは、対応付けられたスーパーピアと論理的なリンクで結ばれる。スーパーピアは、自身に対応付けられたファイルのインデックスを管理することで、通常の検索エンジンと同様の働きをする。すなわち各ファイルのインデックスは、適宜スーパーピアに収集され維持されるとともに、ユーザから出された検索要求(クエリ)に対する検索処理をおこなう際に参照される。

本システムにおけるサーバのおもな役割は、ファイルとスーパーピアとの対応付けを管理することである。ただしファイルとスーパーピアの対応は静的に固定するのではなく、アクセス頻度やアップロードファイル数などの変化にしたがって適応的に変化させなくてはならない。また一般に検索システムでは、クエリに対するレスポンスを一定時間以内におさえる必要があるが、上述のモデル上でそのような動作を実現するためには、与えられたクエリに関係するスーパーピアを高速に特定し、特定されたスーパーピアに向けてすみやかにクエリを転送する仕組みが必要である。さらに、検索結果にファイルの更新情報を即時に反映させるためには、ファイルを更新したUPが、そのファイルに対応付けられたスーパーピアに更新情報を適宜アップロードする必要もある。

3-2 基本ツール

上述の課題を解決するための具体的な方法として、提案手法では以下のような基本ツールを用いた。

（1）タグに基づくクラスタリング

タグの有限集合$T= \{ t_1,t_2,\ldots, t_n \}$を考える。集合T中の各タグは、あるオブジェクトの実世界での意味をあらわすキーワードやキーフレーズである。たとえばchinaというタグは、国、文化、料理などをあらわすために用いられる。提案手法では、UPの保持するファイルを、そのファイルに付与されたタグの集合によって分類する。ここでタグに基づくファイルのクラスタリングを効果的に実現するためには、ファイルに付与されるタグの集合Tが、タグの出現頻度(人気度)を考慮して適切に決められなくてはならないことに注意しよう。Zipfの第一則が主張するように、我々が文章を書く際などに用いている語の使用頻度はべき乗則にしたがうことが知られており、ごく少数の高頻出語が存在する一方で、ほとんど使われない語も多数存在している。すなわち一般には、高頻出語はファイルの分類には強く寄与しない可能性が高く、低頻出語についても、その語が付与されるファイルの数が少ないため、こちらもやはりファイルの分類にはほとんど寄与しない。なお以下では議論を簡潔にするため、集合Tはシステム管理者や専門家などによってあらかじめ定められているものと仮定する。各ユーザがTの要素を動的に追加・編集する方法については、知識工学の分野で用いられているフォークソノミーなどの手法と組み合わせることで実現できる。

（2）タグの優先度列

Tから集合$\{ 1,2,\ldots,|T| \}$への全単射σを考える。以下では$\sigma(t)$のことを全単射σのもとでのタグtの優先度と呼び、$\sigma(t_1) < \sigma(t_2)$を満たすとき、タグ$t_1$は$\sigma$のもとでタグ$t_2$よりも高い優先度をもっているなどという。また以下では次のような列をσのもとでのタグの**優先度列**と呼ぶ：$\sigma^{-1}(1), \sigma^{-1}(2),\ldots,\sigma^{-1}(|T|)$。ここで$\sigma^{-1}$は、関数$\sigma$の逆写像である。以上の記法を用いて、タグ集合間の包含関係を次のように定義する。

定義：タグの部分集合$T_1,T_2 (\subseteq T)$を考える。T_1は、T_2の優先度列がT_1の優先度列の接頭語になっているとき、σのもとでT_2に包含されるという。

例：$T = \{ t_1,t_2,\ldots,t_9 \}$とし、各iについて$\sigma(t_i) < \sigma(t_{i+1})$が成り立っているとしよう。部分集合$T_1 = \{ t_1,t_2,t_3 \}$は$\sigma$のもとで部分集合$T_2$を包含している。なぜならば、$T_2$の優先度列$t_1,t_2$は$T_1$の優先度列$t_1,t_2,t_3$の接頭語になっているからである。一方、部分集合$T_3 = \{ t_2,t_3,t_4 \}$は$T_2 = \{ t_1,t_2 \}$には包含されていない。

定義：タグの部分集合T1,T2(⊆T)は、お互いに包含関係にないとき、比較不能であるという。

上記の定義のもとで、与えられたタグ集合間の包含関係をチェックする具体的な手続きは以下のように与えられる：

Function INCLUSION(T_1,T_2)
Step 1: もし$|T_1| < |T_2|$ならばfalseを返して終了する。
Step 2: もし$|T_2|=0$ならばtrueを返して終了する。
Step 3: t_1をT_1中でもっとも優先度の高いタグとし、t_2をT_2中でもっとも優先度の高いタグとする。 T_1とT_2からそれぞれt_1、t_2を取り除く。
Step 4: もし$t_1 \neq t_2$ならばfalseを返して終了する。そうでなければ、Step 2へ戻る。

（２）手続き

各UPが保持しているファイルのインデックス情報を対応するスーパーピアにアップロードする手続きは以下の通りである。ここで各スーパーピアはそれぞれタグの集合に対応付けられており、各UPの保持するファイルもまた、少なくともひとつのタグと対応付けられているとする：

Procedure FILE_UPLOAD
Step 1: インデックス情報をアップロードするファイルに付与されたタグの集合をT'とする。
Step 2: T'を包含するタグ集合と対応付けられたスーパーピアS_iを特定する。
Step 3: スーパーピアS_iと接続し、ファイルインデックスをアップロードする。

この手続きは、あるUPが新しいファイルを生成したときと、そのUPがすでにもっているファイルを更新したときに、そのUP自身によって呼び出される。更新リクエストの宛先はサーバによって計算され、対応するファイルのインデックスを管理するスーパーピア名とそのI」Pアドレス、ポート番号が返される。ここで、提案手法では、各スーパーピアがファイルのインデックス情報のみを管理していることに注意されたい。すなわちスーパーピアの負荷は、インデックス情報の管理のみに限定される。なお、与えられたクエリからそのクエリに対応するスーパーピアを特定する手続きも、上記の手続きとほぼ同様の手順で実現される。

3-3 センサー情報の共有システムへの応用

前節で述べた提案方式の有効性を具体的に検証するため、無線センサーネットワーク(WSN)によって検知される局所的な情報をP2Pシステム上でグローバルに共有するシステムを設計し、そのプロトタイプを作成して評価をおこなった。実装環境は以下の通りである：Windows Vista、JDK/1.6、NetBeans IDE 6.5.1。センサデバイスとしてはSun Microsystems社製のSunSpotを用いた。なおこのシステムでは、前述のモデルにおけるUPにWSNのアクセスポイントの役割を兼ねさせることで、IPネットワークとWSNという異種のネットワークの透過的なブリッジ機能も実現している。

（１）概要

まず各UPは、自身がアクセスポイントとなっているWSNから定期的にセンサ情報を受信し、その情報をファイルに保存する。各センサには、後述するようにそのセンサを特徴付けるタグの集合があらかじめ付加されており、UPによって保存されたファイルは、センサのタグ集合とともに管理される。WSN上の情報伝播は、ブロードキャストにより実現する。またユーザからの検索要求は、後述するように、前節で述べた手法を応用することで処理される。

ここで上記のようなシステム構成法を用いることで、以下の二つの意味でシステムの可用性が向上することに注意しておこう：第一点は、二つのネットワークの間を密に接続させられているという点である。センサから送信されたメッセージは到達可能な範囲にあるすべてのUPによって受信され、ファイルとして保存される。そのため、仮に一部のUPがネットワークから離脱した場合にも、システム全体を正しく機能させ続けることができる。第二点は、センサの障害に対するシステムの耐性の高さである。この方法では各センサは、ハードウェアやソフトウェアの仕様に依存しないタグ集合によってのみ識別されているため、障害時に他のセンサーで代替させることが容易におこなえる。

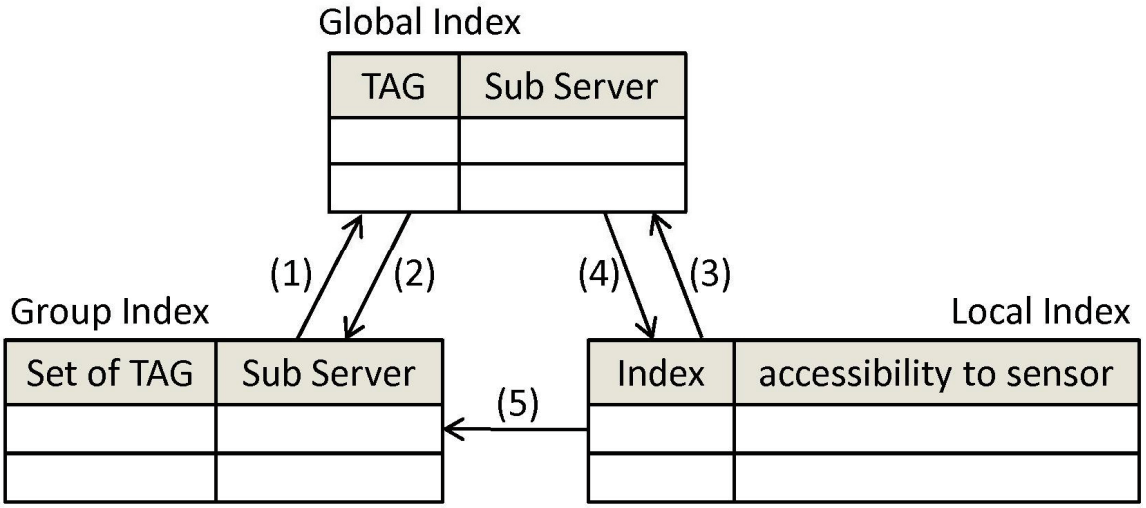

図1　プロトタイプシステムで用いた基本データ構造間の関係

（２）データ構造

プロトタイプシステムでは、Global Index、Group Index、Local Indexの三つの基本データ構造(インデックステーブル)を用いた。図１にそれぞれのインデックステーブル間の関係を示す。図中の番号付き矢印は、これらの3つのデータ構造間の通信の順番を示している。矢印(1)と(3)はそれぞれスーパーピア、UPが送信する登録メッセージであり、矢印(2)と(4)はそれらに対するサーバからのレスポンスである。また矢印(5)はUPからスーパーピアに対するインデックスのアップロードを示す。

各インデックステーブルの詳細は以下の通りである：Global Indexは、タグ集合とそれに対応するスーパーピアのIDの組からなるテーブルであり、サーバによって管理される(ここでIDは、IPアドレスとポート番号の組である)。サーバは、登録メッセージをスーパーピアから受信すると(矢印(1))、メッセージに含まれるタグ集合とスーパーピアのIDをGlobal Indexに追加する。またインデックスのアップロード要求やクエリをUPから受信すると(矢印(3))、Global Indexを参照して対応するスーパーピアを特定し、そのIDを返送する。

Group Indexは、タグ集合とその集合に対応するファイルのインデックスからなるテーブルであり、各スーパーピアによって管理される。スーパーピアは、ファイルインデックスのアップロード要求をUPから受信すると(矢印(5))、メッセージに含まれるタグ集合とインデックスをGroup Indexに追加する(ここでファイルのインデックスは、ファイルの所有者情報を含んでいる)。またクエリを受信すると、Group Indexを参照してクエリにマッチするエントリーを検索し、もしそのようなエントリーが存在していれば、ファイルの所有者情報を返送する。

Local Indexは、ファイルインデックスとセンサへのアクセスの可否をあらわすフラグからなるテーブルであり、各UPによって管理される。各UPは、ネットワーク参加時や新しいファイルの作成時にファイルのインデックスを作成する(ファイル作成は、WSNから新しいセンサ情報を受信した場合と、他UPからファイルをダウンロードした場合に実行される)。インデックス作成後、そのインデックスとセンサへのアクセスの可否をあらわすフラグをLocal Indexに登録する。ここでセンサへアクセスが可能なのは、WSNから直接センサ情報を受信した場合のみに限られる。

（３）メッセージ

各ピアは受信メッセージを扱うListenerをもっており、各メッセージは、Listenerによって作成されたスレッドのメッセージパイプを通って受信され、メッセージキューにプッシュされる。Listenerは受信したメッセージをひとつずつ受け取り、受信したメッセージごとの処理をおこなう。またもし必要であれば、別のピアにメッセージを送信する。一方センサは、定期的にUPに向けてセンサ情報を送信する。センサが送信するメッセージは到達可能な範囲のすべてのUPによって受信される。

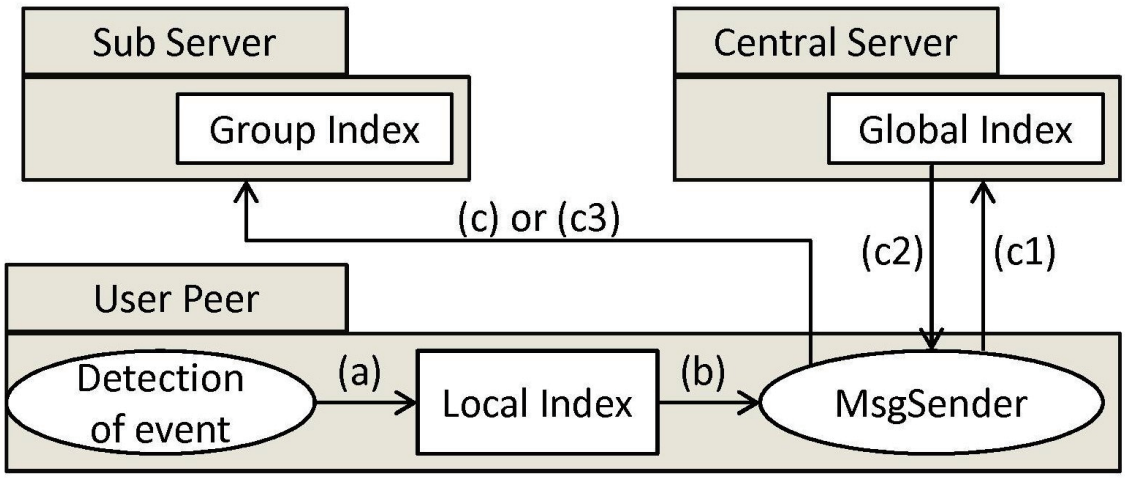

図2　プロトタイプシステムにおける手続き間の呼び出し関係

（４）手続き

インデックスのアップロードの様子を図２に示す。ファイル更新の発見はイベント駆動方式でおこなわれる(ここでイベントとは、UPのネットワークへの参加、センサ情報の受信、ファイルのダウンロードである)。UPは新しいイベントを検知すると、ファイルのインデックスを作成し、Local Indexにそのインデックスを追加した後(矢印(a))、インデックスのアップロードを依頼する(矢印(b))。自分が所属するクラスタに関連するインデックスの場合はスーパーピアに直接アップロードし(矢印(c))、それ以外の場合はサーバにインデックスの転送を依頼する(矢印(d1))。サーバはGlobal Indexを参照し、受信したインデックスをそのインデックスに関連するスーパーピアに転送する(矢印(d2))。スーパーピアは受信したインデックスをGroup Indexに登録する。

UPが送信するクエリは、2種類の方法でスーパーピアに到達する。上で述べたように、UPはローカルメモリに自分が参加しているクラスタのスーパーピアのリストを持ち、このリストを用いて2種類の方法を使い分ける。前者の方法は自分が参加するクラスタのスーパーピアに直接送信する方法であり、後者の方法はサーバにクエリの転送を依頼する方法である。クエリを受信したサーバは、インデックスのアップロードと同様に、クエリをそのクエリに関連するスーパーピアに転送する。クエリを受信したスーパーピアは、クエリを処理してクエリレスポンスをそのクエリを送信したUPに返す。

UPがセンサ情報をダウンロードする方法は2種類存在する。前者の方法は既にセンサ情報が蓄積されたファイルをダウンロードする方法であり、後者の方法はセンサから得た最新のセンサ情報を追記したファイルをダウンロードする方法である。いずれの方法でも、ダウンロードの通信相手の決定にはスーパーピアから受信したクエリレスポンスが利用される。より具体的には、前者の方法では、ファイル要求者はファイルを所有するUPに単にファイルのアップロードを依頼する。アップロード要求を受信したUPは、要求されたファイルをファイル要求者にアップロードする。後者の方法では、ファイル要求者はセンサにアクセス可能なUPにセンサへのクエリ送信を依頼する。センサへのクエリ送信を依頼されたUPはWSNにクエリをブロードキャストし、センサからクエリレスポンスを受信した場合は、受信したセンサ情報をファイルに追記しファイル要求者にファイルをアップロードする。これらの2種類のダウンロードは、UPのインターフェース上でダウンロード方法を選択するだけで実行できるため、ユーザはP2PネットワークとWSNを透過的に利用することができる。

3-4 プロトタイプシステムの評価

本節では、実装したプロトタイプシステムの有効性と拡張性を実証するためにおこなった実験の実験方法と結果を示す。以下では提案システムの有効性を、ファイルのインデックスのアップロード時間とファイルのインデックスの検索時間の二つの観点から評価する。加えて提案システムの拡張性を、UPの数とそれぞれのUP が持つファイル数に注目して評価する。

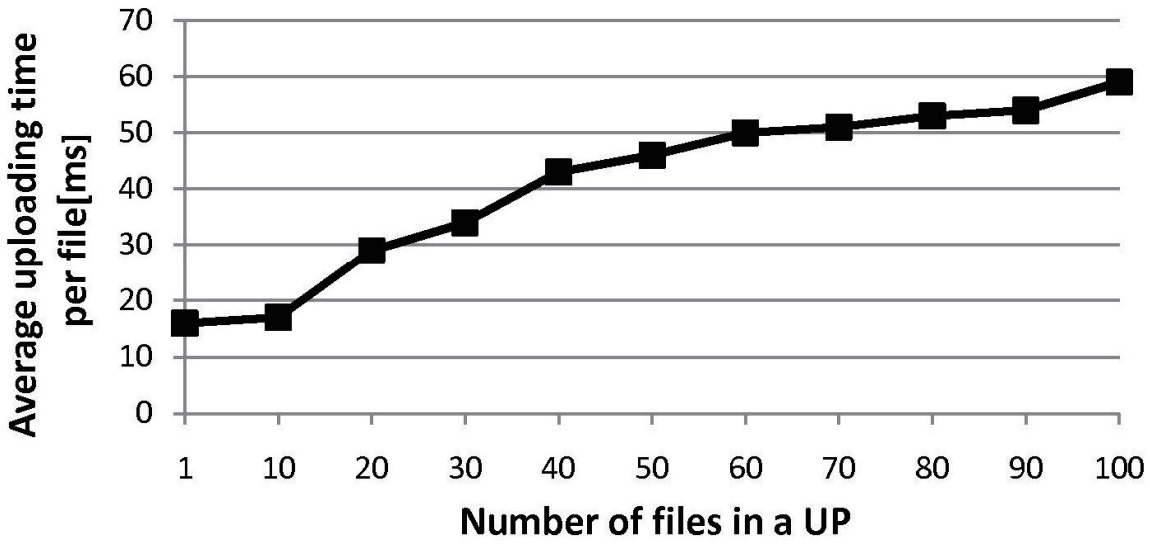

図4　ファイル数ごとのインデックスのアップロード時間.

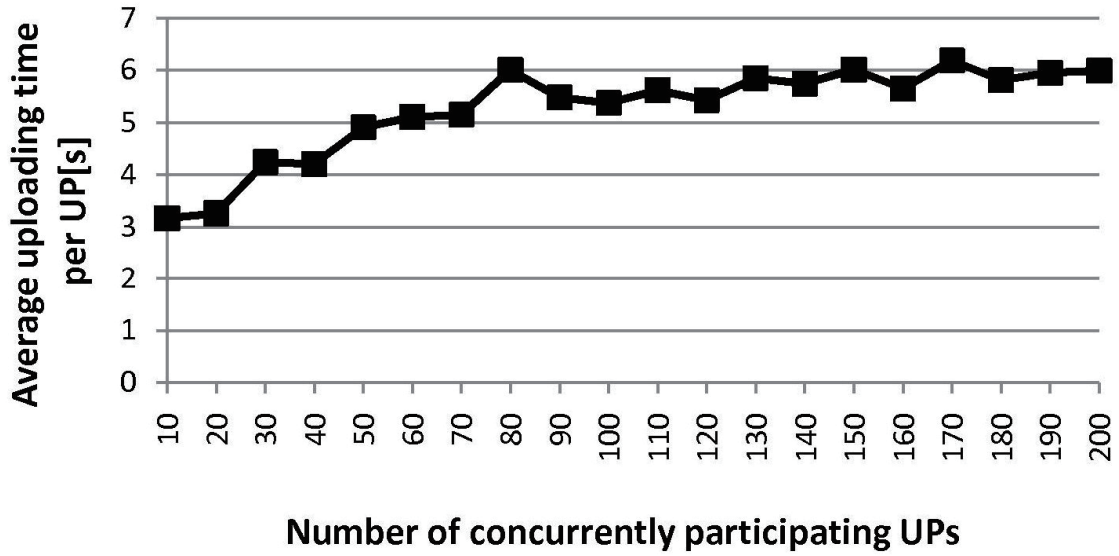

図5　UP 数ごとのインデックスのアップロード時間.

（１）設定

実験で用いた基本設定は以下の通りである。ネットワークを構成するピア数は、中央サーバが1、サブサーバが100とし、UPの数は実験ごとに変化させる。センサ情報を書き込んだファイルを10000 個作成し、各UP に100 個ずつ配置する。なお本実験では計算機を12台用い、1台の計算機で複数のピアを起動している。具体的には、サーバは計算機1台で起動し、サブサーバは計算機1台につき10 ピアずつ起動した。UPはすべて1台の計算機で起動した。

（２）結果

まずファイルのインデックスのアップロード時間を評価する。図4にネットワークに参加するUPを1 ピアに固定した場合における、UP の所有するファイル数とインデックスの平均アップロード時間の関係を示す。図より、UP がひとつのインデックスをアップロードするために必要な時間が16 ミリ秒程度であることが分かる。さらに、UPが持つファイル数が100個以下のとき、インデックスのアップロードにかかる時間は60 ミリ秒以内に抑えられている。図5には、各UPが100個のファイルをもつ場合における、ネットワークに同時に

参加するUPの数とインデックスの平均アップロード時間との関係が示されている。ネットワークに同時に参加するUP数が200以下のとき、インデックスのアップロード時間は数秒に抑えられている。

次にファイルインデックスの検索時間を評価した。実験の結果、1台のUPピアが1回の検索結果を得るために必要な時間が15ミリ秒程度であることが分かった。図6には、同時に検索を実行するUP数が10、20、30の場合における、クエリ連続送信数とインデックスの平均検索時間の関係が示されている。同時に検索をおこなうUP数が20以下でありかつ連続クエリ送信数が60以下の場合、インデックスの検索時間は1秒以内に抑えられていることがわかる。同時に検索をおこなうUP数が30の場合は、同時に検索するUP数が20以下の場合と比べてインデックスの検索時間が長くなっている。これはサーバの過負荷が原因と考えられ、その対策が今後の課題として挙げられる。

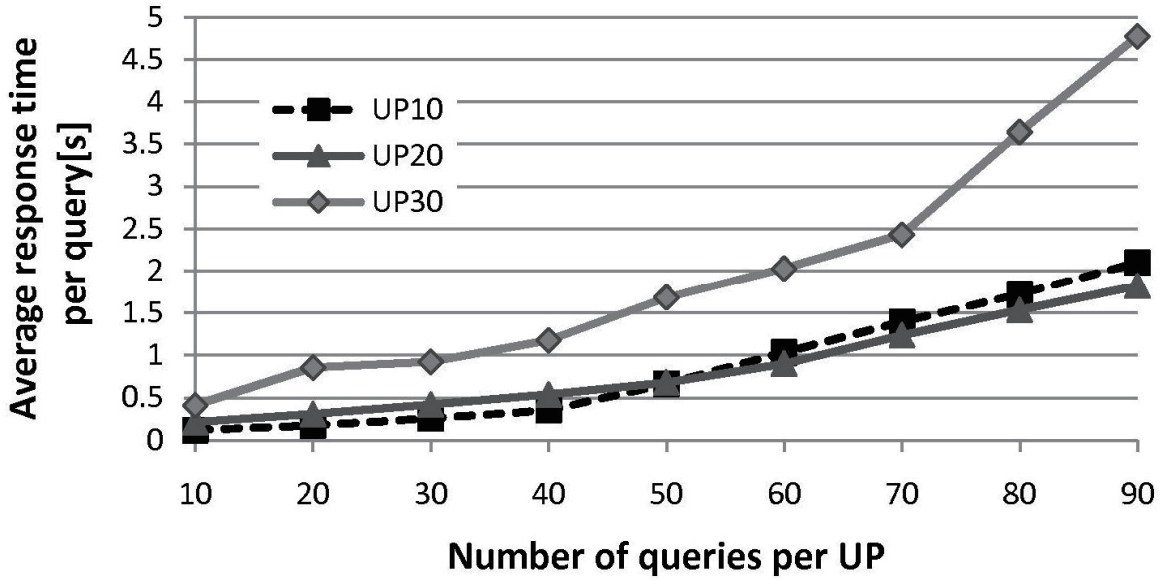

図６　インデックスの平均検索時間

【参考文献】

E. Adar and B. A. Huberman. Free Riding on Gnutella. First Monday, 5(10), 2000.

J. Aspnes and G. Shah. Skip Graphs. In Proc. The 14th Annual ACM-SIAM Symposium on Discrete Algorithms, pp.384-393, Jan. 2003.

H. Balakrishnan, M. F. Kaashoek, D. Karger, and R. Morris. Looking Up Data in P2P Systems. Communications of the ACM, 46(2):43-48, Feb. 2003.

A. Broder and M. Mitzenmacher. Network applications of bloom filters：A survey. Internet Mathematics, 1(4):485-509, 2003.

G. Ding, and B. Bhargava. Peer-to-peer File-sharing over Mobile Ad hoc Networks. In Proc. The 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04), pp.104-108, March 14-March 17.

S. Gribble, E. Brewer, J. Hellerstein, and D. Culler. Scalable, Distributed Data Structures for Internet Service Construction. In Proc. the 4th Conference on Symposium on Operating System Design and Implementation (OSDI), 4:22, October 2000.

M. Harren, J. M. Hellerstein, R. Huebsch, B. T. Loo, S. Shenker, and I. Stoica. Complex Queries in DHT-based Peer-to-Peer Networks. In Proc. IPTPS, Mar. 2002.

V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti. A Local Search Mechanism for Peer-to-Peer Networks. In Proc. CIKM, 2002.

H.　L. Kim, S. H. Hwang, and H. G. Kim. FCA-based approach for mining contextualized folksonomy. In Proc. the 2007 ACM symposium on Applied Computing, pp.1340-1345, 2007.

J. Li, and S. T. Vuong. An Efficient Clustered Architecture for P2P Networks. In Proc. the 18th International Conference on Advanced Information Networking and Applications (AINA), 2:278-283, 2004.
Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In Proc. ACM SIGMETRICS, 2002.
Napster Homepage. http://www.napster.com/
M. Newman. Power laws, Pareto distributions and Zipf's law. Contemporary Physics, 46:323-351, 2005.
T. T. Qin, Q. Cao, Q. Y. Wei, and S. Fujita. A Hierarchical Architecture for Real-Time File Search in Peer-to-Peer Networks, In Proc. PDAA, in conjunction with PDCAT09, Dec. 2009.
A. I. T. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In Proc. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pp.329-350, November 2001.
I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In Proc. ACM SIGCOMM, pp.149-160, August 2001.
B.Yang, and H.Garcia-Molina. Designing a Super-Peer Network. In Proc. the 19th International Conference on Data Engineering (ICDE), pp.49, March 2003.
B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: A Resilient Global-scale Overlay for Service Deployment. IEEE Journal on Selected Areas in Communications, 22(1):41-53, Jan. 2004.

〈発 表 資 料〉

題　　名	掲載誌・学会名等	発表年月
A Hierarchical Architecture for Real-Time Search in Peer-To-Peer Networks	Proc. International Workshop on Parallel and Distributed Algorithms and Applications (PDAA)	2009 年 12 月
Quick Forwarding of Queries to Relevant Peers in a Hierarchical P2p File Search System	International Symposium on Frontiers of Parallel and Distributed Computing (FPDC10)	2010 年 5 月