2006－DBS－140（Ⅰ）（9）
2006／7／12

複製による負荷分散を可能にしたP2Pプロトコルの提案

佐治　和弘†　　有次　正義††

† 群馬大学大学院工学研究科情報工学専攻　〒376-8515 群馬県桐生市天神町1-5-1
†† 群馬大学工学部情報工学科　〒376-8515 群馬県桐生市天神町1-5-1
E-mail: †kazuhiro@dbms.cs.gunma-u.ac.jp, ††aritsugi@cs.gunma-u.ac.jp

あらまし　P2Pシステムに対する効率的な負荷分散は重要な問題である．BATONでは，各ノードが自律的に自身の持つ値の範囲を調整し，負荷分散を実現している．しかし，システム内に人気の高いオブジェクトが存在する場合，人気オブジェクトを持つノードのオブジェクト送信負荷が高くなってしまい，BATONの負荷分散だけでは対応しきれなくなってしまう恐れがある．本稿では，BATONにSCOPEで提供されている複製の分散管理手法を組み合わせ，人気オブジェクトを持つノードのオブジェクト送信負荷を複製を持つノードに分散するP2Pプロトコルを提案，検討し，今後考えなければならない課題を明らかにする．

キーワード　ピアツーピア，複製，負荷分散

A P2P Protocol with Load Balancing using Replicas: A Proposal

Kazuhiro SAJI† and Masayoshi ARITSUGI††

† Department of Computer Science, Graduate School of Engineering, Gunma University
1-5-1 Tenjin-cho, Kiryu, Gunma 376-8515, Japan
†† Department of Computer Science, Faculty of Engineering, Gunma University
1-5-1 Tenjin-cho, Kiryu, Gunma 376-8515, Japan
E-mail: †kazuhiro@dbms.cs.gunma-u.ac.jp, ††aritsugi@cs.gunma-u.ac.jp

Abstract　It is important to realize efficient load balancing mechanisms in P2P systems. BATON, a P2P system, supports a mechanism in which each node changes the range of objects to be managed autonomously. However, if there are nodes managing objects with high request rates in a network, the mechanism of BATON may fail to achieve good load balance because of the high loads for serving such objects. In this paper, we propose a P2P protocol in which we integrate a method of replica management proposed by SCOPE into BATON and discuss issues for developing a system with good load balance mechanisms.

Key words　P2P, Replica, Load balancing

1. はじめに

Structured P2Pシステムの代表的な手法として，DHTを用いた手法がいくつか提案されており[1]，Chord[2]，Pastry[3]等のシステムが開発されている．これらのDHTを用いた手法は，効率的なデータの配置方法と探索方法を提供するが，データに識別子を割り当てるために用いるハッシュ関数が似ているデータ同士にも全く異なる識別子を割り当て，ばらばらにデータを配置するため，レンジクエリをサポートするためのデータの配置には不適切である．また，ハッシュ関数を用いない場合，Chordのようなリング型オーバーレイでは，データの偏りがデータの分布に大きく影響してしまうため，多くのデータを保持するノードが出現してしまう恐れがある．

レンジクエリをサポート可能なオーバーレイネットワークとして，BATON[4]が挙げられる．BATONではP2Pのオーバーレイネットワークの構造に平衡2分木を用いており，DHTを用いた手法ではサポートできなかったレンジクエリのサポートが可能となっている．また，BATONでは各ノードが，自律的に自身が受け持つ値の範囲を調節し，システム内の各ノードに掛かるクエリ数，アクセス数，管理するデータ数等の負荷を均一に保つことができる．

しかし，多くのアクセスを受けるような人気オブジェクトが存在する場合，人気オブジェクトを管理するノードは過負荷状態となってしまい，値の範囲調節だけでは対応しきれなくなってしまう恐れがある．この問題の解決策の1つとして，人気のあるオブジェクトの複製を生成して他のノードに持たせ，負荷を分散する方法が考えられる．しかし複製が多数存在する場合を考えると，複製の管理は容易ではない．

これらの問題を解決する手法として，本稿では BATON に SCOPE [5] で提案されている複製の分散管理手法を組み合わせることで，効率的な複製の分散管理の手法と，複製を用いて人気オブジェクトの送信負荷を分散する手法を提案する．本手法の特徴を以下に示す．

- 複製を用いた負荷分散

ノードが自身の管理するオブジェクトに対する要求を多数受け取った時，データの送信をオブジェクトの複製を持つノードに任せることで，オブジェクトの送信に掛かる負荷を分散し，オリジナルのオブジェクトを持つノードの負荷の軽減を目指す．

- 複製の分散管理

SCOPE で提案されている複製の位置情報の分散管理手法を組み合わせることで，複製を管理する負荷の軽減を目指す．

- レンジクエリをサポート

オーバーレイネットワークとして BATON を用いているので，レンジクエリをサポートできる．

- ノード参加時のルーティングテーブルの更新コストが低い

BATON のプロトコルを用いるため，システム内のノード数が N の時，ノード参加位置の発見と，ノード離脱時の代わりとなるノードの発見は，$O(logN)$ ステップで可能である．さらに新たにノードが参加した際にルーティングテーブルの更新に必要となるコストは $O(logN)$ である．これは，Chord 等の他の DHT を用いた手法に比べると非常に低い．例えば Chord のルーティングテーブル更新コストは $O(log^2N)$ である．

eDonkey，BitTorrent，FastTrack，WinMX 等の P2P サービスのトラフィックの計測を行った文献 [6] では，P2P のトラフィックを，データのダウンロードのためのトラフィックと，クエリの処理やネットワークの維持のためのトラフィックの 2 つに分けており，その閾値を 20K bytes としている．また eDonkey のトラフィックを計測した文献 [7] によると，download ストリームの平均サイズが 2.48M bytes なのに対し，non-download ストリームの平均サイズが 16.7K bytes と計測されており，データダウンロードに使われるデータのサイズはそれ以外のメッセージデータのサイズに比べて非常に大きく，総送信コストの大半がオブジェクトの送信コストであることがわかる．このことから，他ノードに比べオブジェクト送信量の多いノードが過負荷状態であると考えられる．本稿では，この過負荷状態の解消のため，過負荷状態であるノードで生じたオブジェクト送信作業のいくつかを複製を持つノードに任せる．これにより，オブジェクト送信コストを複製を持つ他のノードに分散することができる．複製を持つノードにオブジェクトの送信を任せる場合，オリジナルオブジェクトを持つノードが新たに複製を持つノードを探索する必要がある．提案手法では複製の管理・探索に SCOPE で用いられている複製の分散管理手法と探索手法を用いた．複製の位置情報を分散管理することで，1 つのオブジェクトの複製が多数生成された場合も，1 ノードに複製の管理コストが集中することを避けることができる．

2. 提 案 手 法

2.1 BATON の構造

本手法では，オーバーレイネットワークに BATON [4] の平衡 2 分木の構造を用いる．木構造内の 1 ノードは Peer-to-Peer システムの 1 ノードと対応している．図 1 に BATON の木構造の例とノード m が持つルーティングテーブルを示す．各ノードには level と number が割り当てられる．level は木構造内でのそのノードの高さ，number は同一 level 内でのノードに左から番号を割り振ったものである．ノードの左隣，右隣のノードをそれぞれ left adjacent ノード，right adjacent ノードと呼ぶ．図中のノード m が親ノード f，右子ノード r，left adjacent ノード f，right adjacent ノード r へのリンクと，左子ノードに null を登録しているように，各ノードは親ノード，子ノード，左右の adjacent ノードへのリンクを持つ．また，各ノードは，右ルーティングテーブルと左ルーティングテーブルを持ち，level L のノードはそれぞれのテーブルに同じレベルのノードを最大で L 個管理できる．number が N のノードの右（左）ルーティングテーブルの j 番目のエントリには，number $N+2^{j-1}$（$N-2^{j-1}$）のノードへのリンクが置かれる．該当するノードがない時は，ルーティングテーブルのエントリを空にする．図中のノード m の左ルーティングテーブルには，ノード m と同じ level で，number が $6-2^0=5$，$6-2^1=4$，$6-2^2=2$ のノード l,k,i へのリンクを，右ルーティングテーブルには level 3 で，number が $6+2^0=7$，$6+2^1=8$ のノード n,o へのリンクを登録する．

2.1.1 ノードの持つ値の範囲

木構造内の全ての葉ノードと内部ノードには値の範囲を割り当てる．ノード内の各リンクには，リンク先のノードが管理している値の範囲を記録し，ノードの値の範囲が変更された時は，リンクに記録されている値の範囲も変更される．各ノードに割り当てられた値の範囲の例を図 2 に示す．各ノードは left adjacent ノードの最大値以上から right adjacent ノードの最小値未満の値の範囲を管理する．図中のノード e は，ノード j の最大値 23 以上，ノード p の最小値 29 未満の値の範囲を管理し，ノード q は，k の最大値 39 以上，ノード a の最小値 45 未満の値の範囲を管理する．

2.2 SCOPE の分散管理手法

先に説明した BATON の平衡 2 分木の他に，提案手法では複製の分散管理を行うために SCOPE [5] で用いられている RPT(replica partition tree) を使用する．3 ビットの識別子空間における，オブジェクト 101 の複製が識別子 000 のノード，識別子 111 のノードに配置された時の，RPT の例を図 3 に示す．RPT は各オブジェクトに対して生成される．RPT の各ノードを代表ノード，ノードが持っている 2 ビットの値をパーティションベクトルと呼ぶ．以降に，複製の分散管理手法を，RPT の説明と共に述べる．

BATON のノードが，あるオブジェクトを自身の値の範囲に含んでいる場合，ノードはそのオブジェクトの Primary ノードであると言う．SCOPE では，Primary ノードが過負荷状態と

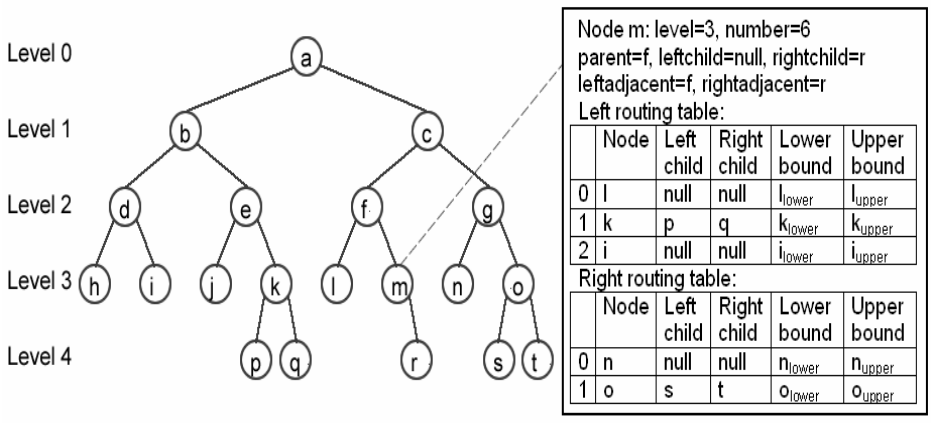

図 1　BATON の木構造とルーティングテーブル [4]
Fig. 1　BATON tree structure and routingtable [4]

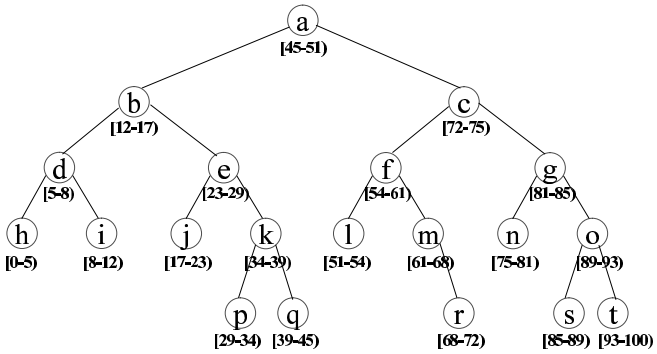

図 2　各ノードの値の範囲の例 [4]
Fig. 2　Example of range of value [4]

なることを避けるために RPT を構築し，複製の情報を各ノードに分散して記録する．SCOPE では識別子空間をパーティションに分割し，パーティションごとに代表ノードを決める．代表ノードは，自身が代表するパーティションをさらに分割して得られる各サブパーティションにおける複製の有無をパーティションベクトルに記録する．下位のサブパーティションも同様に分割され，最も小さなパーティションの代表ノードのパーティションベクトルは複製を持つノードの識別子を示す．代表ノードの決定方法を説明する．以下 1 回の分割で得られるパーティションの数をパーティションサイズと呼ぶ．パーティションベクトルの長さはパーティションサイズと等しくなる．識別子空間が 2^m ビットで表現される時，各ノードとオブジェクトは m ビットの識別子で表現される．パーティションサイズを 2^n とした時，識別子の下位 $(m-n)$ ビットを固定し上位 n ビットに異なる値を付けることで，代表ノードの識別子を決める．

例として図 3 のパーティションサイズを 2 とした時の，オブジェクト 5 = 101 の代表ノードの決定方法を以下に示す．また，各代表ノードが管理するパーティションの範囲を図中の Space に示す．

- Original identifier space

識別子空間の 101 にオブジェクト 5 が割り当てられる．RPT のルートノードが決まる．

- First level partition

下位 2(= 3 − 1) ビットを固定し，上位 1 ビットを可変にする．この時，001 を値の範囲に含む BATON のノードと，101 を値の範囲に含む BATON のノードが first level partition の代表ノードとなり，RPT の中間ノードとなる．

- Second level partition

下位 1 ビットを固定し，上位 2 ビットを可変にする．この時，001，011，101，111 を値の範囲に含む BATON のノードが second level partition の代表ノードとなり，RPT の葉ノードとなる．

RPT の各代表ノードは，下位のサブパーティションの複製の有無を，パーティションベクトルを使用して示す．下位のパーティションに複製が存在する時はパーティションベクトルの対応するビットに 1 を立て，複製が存在しないときはパーティションベクトルの対応するビットを 0 とする．RPT の葉ノードのパーティションベクトルは各識別子における複製の有無を直接示している．図 3 において，オブジェクトの Primary ノードであるルートノード 101 は，下位サブパーティション 000〜011 に複製が存在することをパーティションベクトルの左側のビット 1 を立てることで示し，サブパーティション 100〜111 に複製が存在することをパーティションベクトルの右側のビットに 1 を立てることで示している．RPT にはノード 101 のように，複数のパーティションレベルにおいて代表ノードとなるノードが存在する．これにより，1 ノードが同一オブジェクト

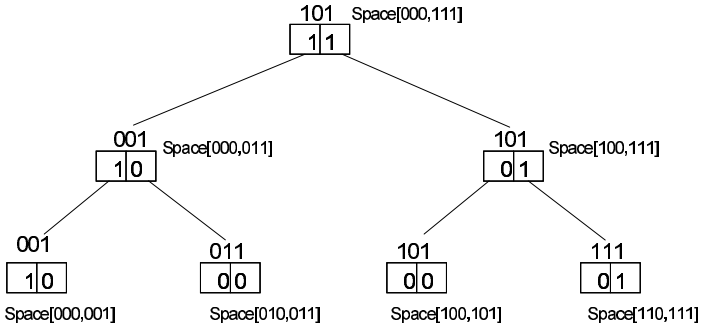

図 3　RPT の例 [5]
Fig. 3　Example of RPT [5]

に対する複数のパーティションベクトルを管理している状況が起きるため，ノード 101 のように，パーティションベクトルを 3 つ持つようなノードが現れる．これにより，パーティションベクトルによるストレージコストの偏りが起きるが，このようなノードは，ノード間の通信を行わずに複数のパーティションベクトルを参照できるメリットも持つ．

2.3　オペレーション

以下，提案手法で使われるオペレーションについて説明する．

2.3.1　Subscribe

複製を持つノードをそのオブジェクトの Subscriber と呼ぶ．Subscribe オペレーションは，Subscriber が複製の位置を Primary ノードに知らせるために行う処理である．本稿ではオブジェクトを受信した際に，検索要求のクエリを発行したノードに複製を配置する．この手法はオーナ複製法と呼ばれており [8]，Gnutella でも用いられている方式である．Subscriber ノードは自身の識別子の下位 n ビットをオリジナルデータの識別子と一致させることで，RPT の葉ノードとなる代表ノードを探す．図 3 において，ノード 010 がオブジェクト 101 の複製を持って Subscriber となった時，ノード 010 はまず，下位 1 ビットをオブジェクト 101 の識別子と一致させて，RPT の葉ノードである代表ノードが 011 であると判断する．ノード 010 はノード 011 に Subscribe オペレーションを送る．オペレーションを受け取った代表ノード 011 は，パーティションベクトルの左側のビットに 1 を立てる．同様の処理を Primary ノード，またはノード 001 のような既に 1 の立っているパーティションベクトルを持つ代表ノードに辿り着くまで続ける．ノード 010 が複製を消去する時は Unsubscribe 処理を実行する．ノード 010 は代表ノード 011 を探索し，ノード 011 はパーティションベクトルの左側のビットを 0 にする．同様の処理を RPT のルートノード，またはパーティションベクトルに 2 つ以上 1 が立っている代表ノードに辿り着くまで続ける．

BATON では各ノードが値の範囲を持ち，範囲は動的に変化する．本システムでは複製を持つノードが持つ値の範囲が変化しても RPT を維持し，複製の位置を代表ノードを辿って把握できるように，複製をノードの値の 1 つにマッピングする．図 2 のノード d が複製を生成した時，複製は 5，6，7 のいずれかの値にマッピングされる．値の範囲が変化し，ノード d の値の範囲が 5〜6 となり，代わりにノード i の値の範囲が 8〜11 から 7〜11 に変化した時，値 7 の管理はノード d からノード i に移るので，値 7 にマッピングされていた複製，パーティションベクトルはノード i に移動する．システム内のオブジェクト数，複製数が増えると 1 ノードが持つパーティションベクトルの量はかなり多くなってしまう．1 ノードが同一オブジェクトの複数のパーティションベクトルを含む場合は，不必要と判断されたパーティションベクトルを削除し，パーティションベクトルによるストレージコストの増大を抑えていく必要がある．図 3 において，000〜001 を値の範囲として持つノードが存在する場合，ノードはパーティションベクトルを 2 つ管理していることになる．このような場合は，上位のパーティションの代表ノード 001 だけを残し，葉ノード 001 のパーティションベクトルは削除可能になると考えられる．パーティションベクトル削除の判断規則，削除後の複製と RPT の管理方法の提案は今後の課題の 1 つである．

複製のマッピングには Subscriber が管理する値の範囲の内からランダムに 1 つの値を選ぶ．Subscriber は選ばれた値に既にマッピングされているオブジェクトと複製の数の合計数を確認し，閾値 T を超えなければ，複製のマッピングを行う．閾値 T の式を以下に示す．

$$T = \frac{o+r}{m} + \beta$$

ここでノードが管理する識別子数を m，ノードにマッピングされているオブジェクトの数を o，ノードにマッピングされている複製の数を r としており，β は定数である．選ばれた識別子にマッピングされているオブジェクトと複製の総数が閾値 T を超えた時は，複製をマッピングする識別子は選び直す．これにより，複製を Subscriber の値の範囲内に均一にマッピングすることができる．複製を多数マッピングされているノードは BATON の負荷分散で提供されている値の範囲の調節機能により値の範囲の一部を他ノードに移し，複製数を減らすことができる．

2.3.2　Exact Match Query

ノード n がデータ v を探索する Exact match query を発行，または受け取った時，ノード n は以下のアルゴリズム Search exact に従ってクエリのルーティングを行う．

Search exact(node n, value v)

If(((LowerBound(n) $\leqq v$)*and*($v \leqq UpperBound(n)$)
or(*node n has replica of v*))
n returns v
Else
If($UpperBound(n) < v$)
m = *The FarthestNode Satisfying Condition*
($LowerBound(m) \leqq v$)
If(*there exists such an m*)
Search exact(m, v)
Else
If($RightChild(n) \neq null$)
Search exact($RightChild(n)$, v)
Else
Search exact($RightAdjacentNode(n)$, v)
EndIf
EndIf
Else
A similar process is followed towards the left
EndIf
EndIf

ここで，UpperBound(n),LowerBound(n) は，それぞれノード n の値の範囲の最小値と最大値を表している．探索は，v の値を範囲内に含むノードか，v の複製を持つノードに辿り着くまで行われる．探索に必要なステップ数は，$O(logN)$ である．

探索後，ノードはオブジェクト送信要求を発行したノードに，要求されたオブジェクトを返す．ここでノードがたくさんのオブジェクト送信要求を受け取って過負荷状態になっている時は，オブジェクトの送信作業を Subscriber に頼む．Primary ノードはまずシステム内におけるオブジェクトの複製の有無をパーティションベクトルから確認し，下位の代表ノードにオブジェクト送信要求を転送する．要求を受けた代表ノードは，パーティションベクトルから Subscriber の位置を判断し，下位の代表ノードにオブジェクト送信要求を送る．オブジェクト送信要求は RPT を辿り Subscriber まで届けられる．代表ノードを含んでいる BATON のノードが図 3 の識別子 011〜101 を値の範囲に含むような場合，ノードは，自身の持つ 4 つのパーティションベクトルの中から，ビット 1 が立っていて，最も下位にある，Space[100,111] の代表ノード 101 のパーティションベクトルを参照し，代表ノード 111 にオブジェクト送信要求を送る．

この時，各代表ノード間のオブジェクト送信要求の送信には Search exact を使って，$O(logN)$ ノードを経由して送られている．RPT を辿って Subscriber を見つけるために，Search Exact を送信する回数は平均で $O(logN)$ なので，Subscriber 発見のために必要となるホップ数は，平均で O (Log^2N) となる．複製による負荷分散は過負荷ノードの解消に繋がるが，Exact match query による探索が，オブジェクトの複製を見つけないまま Primary ノードに辿り着いた場合，Primary ノードはオブジェクト送信要求を Subscriber に送るか，自身が要求されたオブジェクトの送信を行うか判断する．Subscriber に送信要求を送る場合，複製発見までに多くのノードを経由する必要があるので，Primary ノードが過負荷状態である場合以外では，Primary ノードがオブジェクトの送信を行うのが望ましい．単位時間あたりのオブジェクト送信回数が閾値 A を越えたノードは，A + 1 回目以降のオブジェクトの送信処理の a % を Subscriber に行わせることで負荷を分散することを考える．ここで，複製を持つ Subscriber の負荷も考慮したオブジェクト送信負荷の分散が今後の課題として挙げられる．Subscriber が過負荷状態となった時は，それが自身の持つオリジナルオブジェクトの送信による負荷が高いためか，複製の送信による負荷が高いためかを判断し，複製の送信による負荷が高い時は複製を削除し，Unsubscribe オペレーションの呼び出しを適切に行っていく必要がある．

2.3.3 Range Query

Range query も Exact match query と同じ方法で処理される．まず検索する範囲を値の範囲に含むノードを Exact match query で 1 つ探し，そこから左右の adjacent ノードを辿り，残りの検索範囲を持つノードを探す．検索範囲が X 個のノードでカバーされている時，検索に必要なホップ数は $O(logN)+O(X)$ である．

オブジェクトの送信を Subscriber に行わせる場合，各オブジェクトに対して 1 つずつ RPT が存在するため，全てのデータの複製を探すのは，非常にコストが掛かる．Range query の処理に適した RPT の改良は今後の課題の 1 つである．

2.3.4 Update

更新処理は，Primary ノードがオブジェクトを更新した時，または Primary ノードが Subscriber からオブジェクトの更新要求を受け取った時に始まる．更新の通知は Primary ノードから RPT を辿り，全ての Subscriber に送られる．要求を受け取った各レベルの代表ノードはパーティションベクトルの値に応じて下位レベルの代表ノードに要求を送る．RPT を使うことにより Primary ノードは複製の位置の管理コストをパーティションベクトル維持のみに抑えることができ，更新情報も 1 レベル下位の代表ノードに送るだけで済むため，Primary ノードにかかる更新情報データの送信負荷を分散することができる．ここで，各代表ノード発見のために行われる探索はメッセージのやり取りのみで行われ，更新情報は代表ノードが発見されたあと，代表ノード間で直接やり取りされる．これにより更新情報の移動コストを代表ノード間の移動だけで抑えることができる．

2.3.5 Network Restructuring

ネットワークの再構築はオーバーレイネットワークの平衡木のバランスが崩れた時に行われる adjacent link を経由してノードをシフトしていく作業であり，BATON で提供されている機能である．ネットワーク再構築の際には，ノード間のオブジェクトの移動は起きず，変更されるのはノードの level，number，ルーティングテーブルである．また各ノードの値の範囲の変更は行わないので，オブジェクト，複製，パーティションベクトルの変更は起きず，RPT にも影響はない．

2.3.6 Join and Departure

ノードの参加・離脱はBATONのオペレーションを使用する．また，ノードの参加・離脱時にノード間で値の範囲の移動が起きた際には，移動が起こった識別子にマッピングされているオブジェクト，複製，パーティションベクトルも共に移動する．

2.3.7 Data Insertion and Deletion

オブジェクトの挿入と削除するオブジェクトの発見には，BATONのオペレーションを使用する．オブジェクトの削除が行われた時，Primaryノードはネットワーク内に存在するオブジェクトの複製を削除する必要があるので，削除命令を送信する．Primaryノードは RPT を辿り，全ての Subscriber に通知を送る．通知を受け取った各レベルの代表ノードはパーティションベクトルの値から複製の位置を判断し，下位レベルの代表ノードに削除命令を送信する．削除命令を送信後，代表ノードは削除されたオブジェクトに関するパーティションベクトルを削除する．

2.4 負荷分散

BATONの負荷分散処理では，ノードが他のノードへ値の範囲の一部を分けること，他のノードから値の範囲の一部を得ることを許可する．値の範囲の移動がノード間で起こった時，移動された識別子にマッピングされていたオブジェクト，複製，パーティションベクトルの移動も同時に行われる．これにより，システム内のノードの負荷を均等にすることができる．

2.5 Node failure

ノードが Fail した際には，BATON，SCOPE の修復作業を実行し，オーバーレイネットワークと RPT の修復処理を行う．Fail したノードのオブジェクトの複製が他ノードに存在する場合は，修復されたノードは RPT を辿り，オブジェクトを Subscriber ノードから修復することが可能である．

3. おわりに

本稿では，BATON に SCOPE で提供されている複製の分散管理手法を組み合わせることで，人気オブジェクトを持つノードのオブジェクト送信負荷を複製を持つノードに分散する P2P プロトコルを提案した．

Structured P2P システムで複製を利用して性能の改善を目指している手法では，文献 [9] や文献 [10] が挙げられる．文献 [9] では複製を用いた Lookup 遅延の改善を行っており，Zipf 法則に基づいたクエリ分布に対して $O(1)$ の Lookup 性能を提供している．文献 [10] ではファイルの人気度に応じた複製の作成やファイルの再配置を行うアルゴリズムを使用した DHT 基盤のファイル共有システムを提案している．

今後の課題の 1 つとして，提案手法の有効性と，問題点を検討するために評価実験を行うことが挙げられる．提案手法では，過負荷ノードに掛かるオブジェクトの送信負荷の解消が期待できるが，複製を用いることで，複製の管理のためのメッセージトラフィックの増加，パーティションベクトルによるストレージコストの増加，送信要求を Subscriber に任せる時発生する代表ノード探索のための遅延等が起きることが予想される．シミュレーションによる評価実験によって，トラフィック，ストレージコスト，探索のホップ数の計測を行い，提案手法の評価を行うことが必要になる．

課題のもう 1 つは，複製の配置方法の改善である．提案手法は，RPT を用いて複製の分散管理を実現できるが，RPT を使用しての複製の探索や，レンジクエリの複製によるサポートを行う時，複製を発見するまでに多くのノードを経由しなければならないことが問題として挙げられる．これらの問題は，複製の配置方法の改善，複製の位置情報を管理するパーティションベクトルの改良によって解決できると考えられる．

文　献

[1] H. Balakrishnan, M. F. Kaashoek, D. Karger, R. Morris and I. Stoica: “Looking up data in P2P systems”, CACM, **46**, 2, pp. 43–48 (2003).

[2] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan: “Chord: A scalable peer-to-peer lookup service for internet applications”, IEEE/ACM Transactions on Networking, **11**, 1, pp. 17–32 (2003).

[3] A. Rowstron and P. Druschel: “Pastry: Scalable, distributed object iocation and routing for large-scale peer-to-peer systems”, Proceedings of IFIP/ACM Middleware (2001).

[4] H. V. Jagadish, B. C. Ooi and Q. H. Vu: “BATON: A balanced tree structure for peer-to-peer networks”, Proceedings of the 31st VLDB Conference, pp. 661–672 (2005).

[5] X. Chen, S. Ren, H. Wang and X. Zhang: “SCOPE: Scalable consistency maintenance in structured P2P system”, Proceedings of IEEE INFOCOM 2005 Conference, pp. 13–17 (2005).

[6] L. Plissonneau, J.-L. Costeux and P. Brown: “Analysis of peer-to-peer traffic on ADSL”, Proceedings of Passive and Active Network Measurement, pp. 69–82 (2005).

[7] K. Tutschku: “A measurement-based traffic profile of the eDonkey filesharing service”, Proceedings of Passive and Active Network Measurement, pp. 12–21 (2004).

[8] Q. Lv, P. Cao, E. Cohen, K. Li and S. Shenker: “Search and replication in unstructured peer-to-peer networks”, Proceedings of 16th ACM International Conference on Supercomputing (2002).

[9] V. Ramasubramanian and E. G. Sirer: “Beehive: Exploiting power law query distributions for O(1) lookup performance in peer to peer overlays”, Proceedings of Networked System Design and Implementation (NSDI) (2004).

[10] J. Kangashrju, K. W. Ross and D. A. Turner: “Adaptive content management in structured P2P communities”, First International Conference on Scalable Information Systems (2006).