Peer-to-Peerシステム上での効率的なデータ配置による問合せ処理とロードバランスへの寄与

山田太造[†] 相原健郎[†,††]
高須淳宏[†,††] 安達淳[††]

現在，P2P システムは一般的に認知されはじめ，単なるデータ交換システムだけではなく，協調作業のシステムとしても注目を浴びている．しかしながら，協調作業を P2P システム上で行うためには，効果的なデータ共有を行う環境整備が求められている．本研究ではシステム上の各ピアの接続状況が激しく変動するシステムを仮定し，そこでの P2P システム上での効果的なデータ共有を行うための効率的なデータ配置方法を考案している．ここでの提案方式では，問合せ処理およびロードバランシングの向上を期待した方式である．またシステム上の各データの有用性によって複製対象データを決定し，複製されるデータ数を決定する方法も提案する．さらに，データの更新時のデータ再配置に関しても考慮している．提案方式の効果を確かめ，しかもデータ配置のシステムへの寄与の状況を確かめるべく，各種実験を行っている．これらの実験では問合せ処理およびロードバランシングの状況を計測し，効果的な問合せ処理およびロードバランシングを行うことが可能であることが示す．

An Effect on Query Processing and Load-balancing by Efficient Data Placement on Peer-to-Peer Systems

Taizo Yamada,[†] Kenro Aihara,[†,††] Atsuhiro Takasu[†,††] and Jun Adachi[††]

Recently P2P systems have been popular as not only mere data changing systems but also cooperation working systems. However, to do the cooperation working on P2P systems, effective data sharing systems are requested. In the paper, we assume systems that a condition of connection of each peer on systems changes violently, and propose techniques of efficient data placement to do effective data sharing on P2P systems. The techniques which we propose are expected that query processing and load-balancing are improved. Moreover, we propose a technique which a usefulness of data a peer hold defines replicated data, and a usefulness of the peer gives the definition of the number of replicas with which each peer can be generated. In addition, we consider data replacement when each data on systems is updated. To ensure effectiveness of the proposal techniques, we show experimental results on query processing and load-balancing.

1. はじめに

データ交換システムとして普及した Peer-to-Peer（P2P）システムであるが，現在では，単なるデータ交換システム以外にも，グループウェア[7]やオンラインコミュニティ[8]等様々な形態で利用されている．Gnutella[6]や Freenet[5]に代表される Pure P2P（PP2P）システムを利用した協調作業のためのシステム，たとえば，文献に関するアノテーションを配布するシステムや掲示板システム等を考えてみる．このようなシステムでは，各ユーザは他のユーザの保有するデータに対して更新を行うことができ，また関心のあるグループへ自由に参加できる環境であることが望まれる．このようなシステムでは，システム上の各ピアは join/leave を頻繁に行うことが予想される．このようなシステムでは参加しているユーザ数および，共有データ数が増加するにつれ，問合せ処理の低下やアクセス集中により，P2P システムにおけるスケーラビリティの低下を招く恐れがある．よって P2P システムはそのスケーラビリティの向上という課題を有する．この解決の方法の 1 つとして，データを複製し，それらをシステム上のいくつかのピア上へ配置する方法が

† 総合研究大学院大学
The Graduate University for Advanced Studies
†† 国立情報学研究所
National Instisute of Informatics

ある．複製配置法は以下のように分類できる．

- No-Replication：この方法では複製の生産は行わない．
- Full-Replication：複製配置を行おうとするピアは，あるピア集合に所属する全ピア上に複製を配置する．
- Partial-Replication：複製配置を行おうとするピアは，システム上のいくつかのピア上に複製を配置する．
- Data Trading [3]：各ピアの保有するデータをオークション形式で，システム上の他のピアへ配置する．これは主にディジタルアーカイブシステムとして用いることを想定した方法である．

Sun らは Napster [9] や OpenNap [11] に代表される HybridP2P（HP2P）システム上での Full-Replication および Partial-Replication に関する研究を行っている [14]．他方特別な構造を持つシステム，たとえば CAN [12]，Chord [13]，Tapestry [15] のような DHTs（Distributed Hash Tables）を用いたシステムや P-Grid [1] のような平衡木構造のシステムでのデータ配置および更新方法に関する研究もされている [2),4]．

複製配置は分散データベースシステム（DDBS）で従来から研究されてきたが，P2P システムでは以下の点が異なっている [10]．

- P2P システムではシステム上の各ピアは動的であり，任意に join/leave を行うことができるが，DDBS での各ノードは静的である．
- P2P システムでは決まったスキーマを必要することは少なく，各ピアは動的であるために，あるピアでのスキーマは共有データへ反映しにくい．DDBS では各ノードは静的であるため，共有スキーマの知識を各ノードで保有することができる．
- P2P システムでは全ノードが接続しているとは限らないため，検索結果はシステム全体の結果であることはありえない．DDBS ではほとんどの場合，システム全体での完全な検索結果の集合を返すことが期待される．
- P2P システムでは伝言（word-of-mouth）方法で問合せメッセージをルーティングする．DDBS では各データの位置は厳密に特定されるため，問合せメッセージは直接共有データを保有するノードへ転送される．

このことは，DDBS での複製配置の手法は直接 P2P システムに適応できないことを意味する．

また P2P システムでは，その利用形態によっては共有データの更新が必要になることがある．単なるデータ交換システム等の場合では，データ更新の必要はあまり必要とされないが，本論文が想定している協調作業等では，各共有データの更新が必要とされることがある．特に，複製配置の方法によってデータ更新の方法は大きく異なる．Full-Replication および Partial-Replication では複製を他のピア上に配置するため，複製を配置してあるピアへ新たにデータを配置する必要がある．そのためデータの同期処理が必要になる．

本論文では，PP2P システム上での，問合せ処理向上，ロードバランシングのための Partial-Replication 形式でのデータ配置およびその更新に関する新たな手法を提案する．さらに，システム上で各ピアが複製対象とするデータの選択方法およびその複製数の決定方法も提案する．本論文では以降，2 章では本研究のデータ配置およびデータ更新の方法を示し，3 章では提案する各種データ配置方式での実験の結果について示す．

2. データ配置と更新

本研究では効率的な問合せ処理およびロードバランシング可能な PP2P システムを実現するために，システム上の各ピアはデータを複製し他のピアへ配置する．各ピアは新規データの追加，データの更新および隣接ピアの追加が行われた場合にデータ配置を行う．本章では，本論文で提唱するデータの配置および更新方法について述べ，また，複製対象となるデータの選択方法に関しても述べる．

2.1 複製対象データの選択

本研究では，掲示板システム，Weblog，Wiki または文献のアノテーション共有システム等で用いられる P2P システムを対象としている．これらのシステムにおいて，各ユーザはなるべく参照回数が多く，新規のデータを求めると考えられる．そのため各データはデータの参照回数の増加にともないその有用性は高まり，データの最終更新時刻からの経過時間の増加にともないその有用性を低下することが望ましいと考えられる．また，あるデータを複製する場合，そのデータが各ユーザにとってあまり有用ではない場合，そのデータの複製はあまり効率的であるとはいえない．そこで本研究では，各データの有用性を基に，複製対象となるデータを選択し，そのデータの複製数を決定する方法を提案する．そのために，本研究では各データに**有用度**を設定する．データの有用度はデータの参照回数およびデータの最終更新時刻からの経過時間に基づき決定する．そこで，あるピア p によって保有されているオリジナルデータ d の有用度を以下のように

定めることにした．

$$E_{data}(d) \equiv \frac{d_{ref}+1}{\log(d_{et})+1} \quad (1)$$

ここで d_{ref} を d の参照回数，d_{et} を d を公開時刻もしくは最終更新時刻からの経過時間とする．

PP2P システムでは，サーバが存在しないため，システム全体のデータの情報を集約することは困難である．また，各データの有用度は参照回数や時間の経過により非常に変動しやすい．そのため，データの有用度でデータの複製数を決定するためにはデータの有用度の情報を頻繁に交換しできるだけ多く得る必要があるため，データの有用度のみを用いてデータの複製数を決定することはシステムのパフォーマンス低下させる要因になる可能性がある．そこで各データの有用度を用いてシステム上のピアに有用度を設け，ピア p とその各隣接ピアの有用度によって各ピアの複製数を相対的に決定することにした．そのために，各ピアの有用度を決定する必要がある．ピア p の有用度は p が保有するオリジナルデータの有用度の総和によって決定することにした．これを式 (2) に示す．

$$E_{peer}(p) \equiv \sum_{d \in D(p)} E_{data}(d) \quad (2)$$

ここで，$D(p)$ はピア p の保有するデータ集合とする．

次に p における複製数を決定する．p とその隣接ピアの集合を $\mathcal{P}$，1 ピアあたり c 個の複製を生成する場合，p とその隣接ピア全体での複製の合計数は $c \cdot |\mathcal{P}|$ となる．このとき p の複製数 $P_{peer}(p)$ は次の式によって決定した．

$$P_{peer}(p) = c \cdot |\mathcal{P}| \cdot \frac{E_{peer}(p)}{\sum_{p_i \in \mathcal{P}} E_{peer}(p_i)} \quad (3)$$

もし $P_{peer}(p)$ の値が発散するのであれば各ピアの複製数を計算できない．各ピアの 1 単位時間あたりのデータ転送は有限であるため，1 単位時間あたりの各データへの参照回数は有限になる．各データの更新中および生成中はそのデータへの参照は不可能であるため $d_{et} > 0$ である．各データの最終更新時間からの経過時間も有限であるため式 (1) は発散しない．また各ピアの記憶領域は有限であるため，各ピアが保有可能なデータ数は有限である．これにより式 (2) は発散しない．また，p はいかなるデータも保有していない場合，$E_{peer}(p) = 0$ とする．よって，$0 \le P_{peer}(p) \le c \cdot |\mathcal{P}|$ となり，式 (3) は計算可能である．

$P_{peer}(p)$ は変動しやすい値であるが，頻繁に再計算を行うのは効率上問題がある．そこで，隣接ピアに変化があるとき，すなわち，隣接ピアからの join もしくは leave の処理があった場合に再計算を行うようにすることで，頻繁な再計算を抑えることができる．

ピア p は以下の手続きで複製対象となるオリジナルデータを選択し，その複製を他のピアへ配置する．

(1) p が保有している各オリジナルデータ d，d の，すでに存在する複製の個数を d_{rep} とする．d_{rep} の合計が $P_{peer}(p)$ 以上であればこの手続きを終了する．
(2) d の有用度 $E_{data}(d)$ を計算する．
(3) $E_{data}(d)/(d_{rep}+1)$ を計算し，この値に応じて所有するオリジナルデータを降順に並べる．
(4) 最もランクの高いデータを選択し，そのデータの複製を配置する．配置の方法は 2.2 節で述べる．
(5) (1) に戻る．

2.2 データの配置方法

次に，複製データ配置の場所を決定する方法が必要になる．本研究では，複製データ配置の方法として以下の 4 つの方法を検討する．

- PN（Placement on Neighbors）方式：各ピアの隣接ピアへ配置する．この方法では，複製データを配置するピアに 1 ホップで到達できるため，データを配置しているピアの状態をすばやく把握することが可能である．
- PIR（Placement on Inverted References）方式：P2P システムでは，あるデータを取得するときに直接ピア間でデータを転送する．ここで，あるデータ d へ参照を行う場合，参照を行ったピアは d に対して関心があり，d を保有するピアと d を参照したピアの間には関連性があると考えられる．あるピア p_1 が保有する任意のデータが他のピア p_2 より参照されたことがあれば，この方式では，p_1 は複製を p_2 上へ配置しようとする．
- PRP（Placement on Related Peers）方式：各ピアはそのピアと類似の問合せを発行するピア上へ複製を配置する．たとえば，ピア p_1 と p_2 はともに “database” というキーワードでの問合せを行うことが多い場合，p_1 と p_2 の関連性は高く，p_1 が複製を配置する場合，p_2 は複製配置の対象となる．データ配置を行うピアはそのピアが既知であるピアから配置対象となるピアを選択するが，このデータ配置方式を用いる場合，各ピアは他のピア情報を得るときに，他のピアの存在以外にも問合せの状況情報も得る必要がある．
- ランダム方式：各ピアは，そのピアが既知である任意のピア上へランダムに複製を配置する．

ランダム方式以外の各データ配置方式においてデータ配置対象となるピアを見つけ出すことができない場合がある．その場合，ランダム方式を用いてデータを配置する．

join の状況にある各ピアは他のピアの情報を次のように得ることができる．

- システム上のピアが join を行うときに，そのピア間にてそれぞれのピア間で既知であるピアの情報を交換する．
- 問合せ処理を行うときに，問合せメッセージを送信したピアと受信したピアは互いの情報を交換する．

2.3 データの更新

オリジナルデータの更新が発生するとき，そのデータに複製が存在すれば，同期処理を行う必要がある．そこで，各オリジナルデータを保有するピアはその複製を配置したピアへデータの再配置を行う．しかしながら，各ピアがデータを同時にアップロードできる数は有限であるため，更新されたデータの複製を同時に再配置できるとは限らない．よって更新の遅延が発生する．本論文では，データ更新の遅延をなるべく抑えるため，各ピアは以下のデータ更新の手続きを行う．ピア p は保有するデータ d を更新した場合，まず，p は d の複製を保有しているピアへデータ更新情報を送信し，この情報を受け取ったピアは d の複製を非同期にする．次に，p はすでにデータ配置を終えており，データ配置を行うことが可能なピアへ複製配置対象のピアのリストを渡す．このとき，d の再配置を終えデータ配置に協力可能なピアの集合を $R_f(d)$，データ更新を終えていないピアの集合を $R_n(d)$ とすると，p は，$R_n(d)$ のリストを $min(|R_n(d)|, |R_f(d)|)$ 個の互いに異なる部分リストに分割し，この個数だけ $R_f(d)$ からピアを無作為に選び，そのピアへ部分リストを送信する．次に，複製配置対象のピアのリストを受け取ったピアは複製対象ピアの再配置の状況を確認する．このとき，複製対象ピアをはシステム上に確認できれば，新データを再配置する．そしてまだデータ再配置が行われていないピアのリストを p へ送信する．以上の手続きを繰り返し，データの再配置を完了する．これにより，更新の遅延を抑えることが可能になる．

また，P2P システムは非常に動的なシステムであるため，データの更新情報が複製配置を行ったピアに到達しない場合もある．たとえば，複製を保有するピアが leave の状況にあるときに，データの更新が行われる場合もありうる．このため，再び join の状況になったときに所有している複製の更新状況を調べる．更新情報を発見したとき，改めてデータの再配置を依頼することでデータの更新を行う．オリジナルデータを保有しているピア p が，以前複製を配置したピア p_i へデータを再配置する際に，データ更新を行いデータを再配置しようとした時刻から時間 t_{ml} 経過しても p は p_i をシステム上に確認できない場合，p_i は長期間 leave している状況であると呼ぶことにする．複製を配置したピアが長期にわたり leave の状態にあるために，データの更新を行うことができない．このとき，オリジナルデータ所有のピアは時間 t_{ml} を経過しても複製配置対象のピアの存在を確認できなければ他のピアにデータ配置を行うことで長期にわたり leave の状況にあるピアによる弊害を抑えることができる．

3. 実　　験

3.1 実験の概要

ここではデータ配置の効果を明らかにするために，以下の観点から実験を行った．

- 1 問合せメッセージあたりの検索結果数：
  “flooding” による問合せ処理によって得られる検索結果数によってを計測した．このとき，各問合せメッセージの伝播は TTL によって制限した．
- 問合せのレスポンスタイム：
  一定以上の検索結果を得るために要したホップ数を用いた．以下の理由のため，本実験では，50 件以上の検索結果数を取得するために要したホップ数を計測した．この実験では任意のピア間での通信速度は一定であると仮定しており，また，P2P システムでの問合せ処理の多くの時間はネットワーク上の通信時間である．
- ロードバランシング：
  アクセス集中の状況を計測するため，各ピアにおけるデータのアップロード頻度を計測した．ここでは，アップロードとは問合せに応じてデータを転送することを意味する．

本実験では，各ピアは 1 単位時間にシステムへの参加状況の変化（join/leave），問合せ処理，データ取得・データ配置，データの更新・作成のメソッドのうちいずれかを実行させた．**表 1** は実験で用いた各定数および各変数を示している．また，この表に示されている各変数の値は，その変数を固定した場合の値である．これらの変数は主に 2 つのカテゴリに分けることができる．1 つは，システム状況を変化させる要因であり，その要因をして，ピア数，1 ピアあたりの複製数およびデータのカテゴリ数がある．問合せ処理を行う場合，これらの値を変えることで，システム全体

表 1 実験に用いた各定数と各変数
Table 1 Constants and variables.

定数	値
ファンアウト数	4
最大データ転送数	4
最大データ取得数	4
TTL	8
最大複製保有数	20
データ作成頻度	0.01
leave 状況の待ち時間	10

変数	説明	値
$join\_ratio$	join を行う確率	0.4
$leave\_ratio$	leave を行う確率	0.4
$query\_ratio$	問合せ処理を行う確率	0.1
$data\_ratio$	データ更新を行う確率	0.09
c	1 ピアあたりの平均複製数	4
num_topic	データのカテゴリ数	20
num_peer	ピア数	10,000

のデータ数を増減させ，問合せ処理の状況を変化させる．もう 1 つは，ピアの各メソッドの実行頻度である．ピアの各メソッドの実行頻度を変化させることによってシステム全体のピア数の増減や，ネットワークの安定性，システム全体のデータ数等の問合せ処理状況を変化させた．本研究では join/leave の頻度を高く設定した場合，データ作成以外の他のメソッド実行頻度を相対的に低下するように設定した．また，問合せ処理の実行頻度を高く設定した場合，データの更新の頻度を相対的に低く設定した．

各ピアはシステムへ再び join を行うとき，最後にシステムへ参加していたときの隣接ピア，これらのピアに対して join できない場合はよく似た問合せメッセージを出していたピア，その他のピアの順序で join を行おうとする．また各ピアには嗜好するカテゴリを設けており，問合せを行う場合，嗜好するカテゴリへの問合せを行う確率は 0.6 とし，データを生成する場合はつねに嗜好するカテゴリの内容のデータを生成する．初期データ分布として偏在のあるデータ分布を用いた．実際にはシステム全体の 80%のデータをシステム上の 20%のピアが保有するデータ分布を用いた．この実験では全データのサイズは同じであると仮定している．

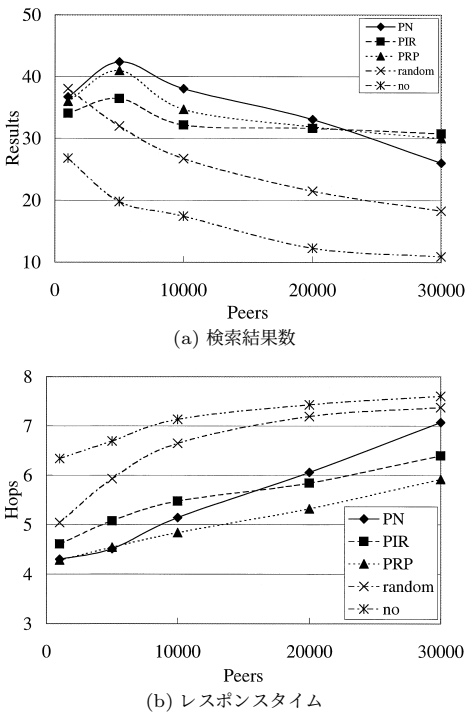
データ配置の方法は 2.2 節で与えた 4 つの方式およびデータ配置を行わない No-Replication の 5 つの方式と，2.3 節で述べたデータ更新方式を組み合わせて計測を行った．また，用いたネットワークトポロジはサイクルの存在する木構造とし，検索方法は Gnutella と同様の “flooding” 方式を用い，検索モデルとしてブーリアンモデルを用いた．P2P システムの問合せ処理では，同一のデータを異なるピアから受け取ることがある．本研究では同一のデータを複数個受け取った場合，1 つの結果を得たものとした．

3.2 実験結果

3.2.1 問合せ処理

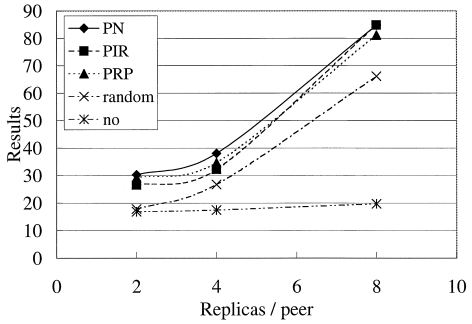
図 1 は，ピア数の変動による問合せ処理への影響を示す．ピア数の変動はシステム全体のデータ数に変化がないため，各配置方式のパフォーマンスは，ピア数の増加にともない低下する傾向にある．ただしピア数が 1,000 のとき，No-Replication 以外の配置方式では，ピア数が 5,000 の場合に比べ検索結果数でのパフォーマンスは低下している．各データの複製数はピアごとに設定されるため，有用度が高いデータであっても，そのデータが複製されない可能性がある．すなわち，ピア数が少ないために，効率的なデータ配置が行われなかったと考えられる．

PN 方式を用いた場合，ピア数が少ない場合に対して有効であるが，ピア数を増加するに従って，そのパフォーマンスは低下した．本実験では，各ピアは同じ嗜好の問合せメッセージを送信するピアに対して join

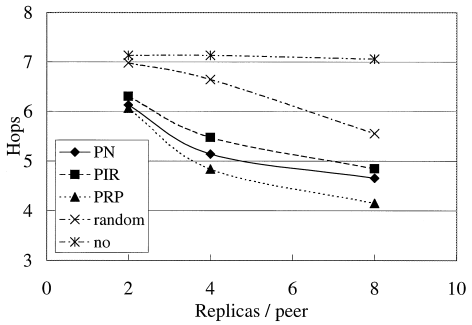


(a) 検索結果数
(b) レスポンスタイム

図 1 num_peer の変動による問合せ処理への影響
Fig. 1 An effect on query processing by varying num_peer.

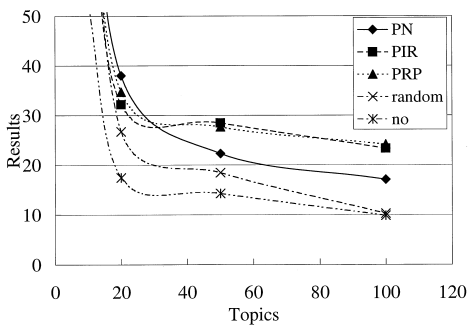


(a) 検索結果数

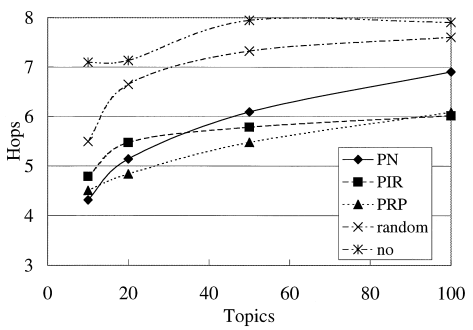


(b) レスポンスタイム

図 2 c の変動による問合せ処理への影響
Fig. 2 An effect on query processing by varying c.



(a) 検索結果数



(b) レスポンスタイム

図 3 num_topic の変動による問合せ処理への影響
Fig. 3 An effect on query processing by varying num_topic.

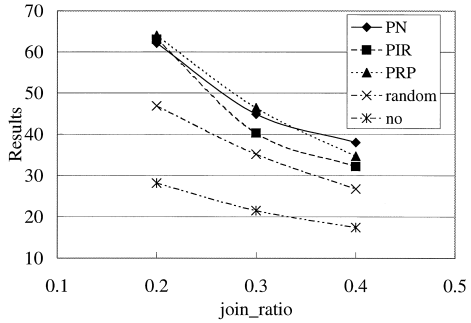
をしようとした．PN 方式を用いた場合，複製配置対象となるピアは隣接ピアであるため，各ピアが嗜好するデータが近い距離に位置するピア上に存在する傾向がある☆．これはピア数が少ない場合には非常に有効である．PN 方式では，各ピアから遠い距離に位置するピアに配置されているデータがそのピアによって嗜好される確率は低い．各データは局所的に配置されるため，ピア数の増加に従って，各ピアによって発見できるデータの数は減少する．よって PN 方式はスケーラビリティに問題があることが分かる．ランダム方式は，問合せメッセージを送信したピアから遠い距離に位置するピア上にも嗜好するデータがいくらか配置されるが，ピア数の増加にともない，嗜好するデータが配置されている確率は低くなるため，そのパフォーマンスは低下する．

PN 方式およびランダム方式に対し，PIR 方式および PRP 方式では，ピア数の影響をあまり受けなかった．その要因として，これらの方式では，PN 方式と比較した場合，各ピアから離れた距離に位置するピア上にも，そのピアによって嗜好されるデータが配置されるためと考えられる．PIR 方式は，ピア数の増減にかかわらず，PRP 方式よりも低いパフォーマンスであったが，両方式はともに同様の傾向を示した．ピア数の増加にともない，PRP 方式と PIR 方式のパフォーマンスの差は，わずかではあるが縮小する．PRP 方式を用いた場合，ピア p から離れたピア p_i 上に p によって嗜好されるデータが配置されている確率が高いが，p と p_i の距離が遠くなるに従って，p が嗜好するデータが p_i に配置されている確率は低くなると予想される．PIR 方式を用いた場合も同様であると予想されるが，各ピアから離れて位置するピアにそのピアによって嗜好されるデータが配置されている確率は，PRP 方式の場合よりも高いと予想できる．

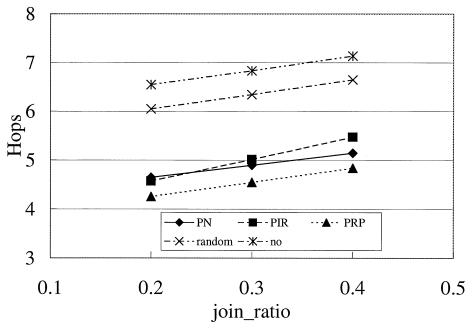
図 2 は 1 ピアあたりの複製数の問合せ処理への影響を示す．1 ピアあたりの複製数を増加させた場合，システム全体のデータ数は増加する．システム上に多くの複製が存在する場合，提案した 3 方式ともそのパフォーマンスが優れていることが分かった．ただし，PN 方式を用いた場合，各ピアと近い距離に位置するピアに多くの同一の複製が配置されている．そのため，PN 方式を用いた場合のレスポンスタイムは PRP 方式よりも劣る．

図 3 はデータのカテゴリ数の変動による問合せ処理

☆ 本研究ではピア間の距離をホップ数で表す．

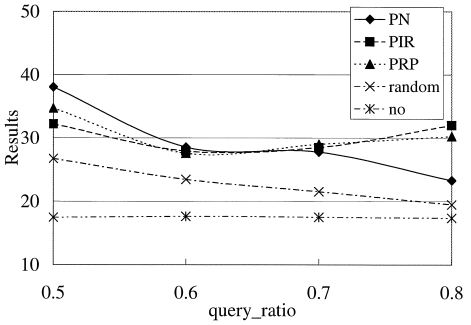


(a) 検索結果数

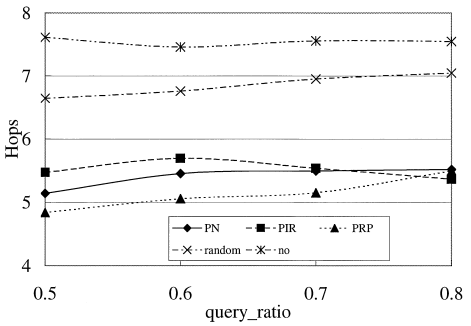


(b) レスポンスタイム

図 4 *join_ratio* の変動による問合せ処理への影響
Fig. 4 An effect on query processing by varying *join_ratio*.



(a) 検索結果数



(b) レスポンスタイム

図 5 *query_ratio* の変動による問合せ処理への影響
Fig. 5 An effect on query processing by varying *query_ratio*.

への影響を示す．データのカテゴリ数を増加させることによって，1 カテゴリあたりのデータ数は低下する．PN 方式およびランダム方式は，カテゴリ数が少ない場合に，優れたパフォーマンスを発揮するが，カテゴリ数の増加にともない，そのパフォーマンスは低下した．これに対し，PIR 方式および PRP 方式では，PN 方式およびランダム方式と比較して，そのパフォーマンスはあまり低下しなかった．PN 方式では，カテゴリ数の増加にともない，隣接ピアが同じ嗜好を持つ確率が低下するため，各ピアから近い距離に位置するピアに配置されているデータがそのピアによって嗜好される確率は低下してしまう．これに対し，PIR 方式および PRP 方式では，各ピアから離れたピア上にも嗜好するデータが配置されている確率が PN 方式よりも高いと予想できる．このことにより，PN 方式を用いた場合に比べ，PIR 方式および PRP 方式を用いた場合ではカテゴリ増加にともなう影響を低減することが可能になる．またデータのカテゴリ数が多い場合では，PIR 方式と PRP 方式でのパフォーマンスの差はあまり見られなくなった．この状況下では，両提案方式が実際には同様のピアを選択し配置していると考えられる．

図 4 にピアによる join/leave の実行頻度による問合せ処理への影響を示す．この実験では，*join_ratio* を変動させた場合，この頻度と同様に *leave_ratio* を変動させている．*join_ratio* の頻度が低いほど安定したネットワークになり，この頻度が高いほど不安定なネットワークになる．*join_ratio* の頻度が低い場合，提案した 3 つの配置方式ともランダム方式よりも良いパフォーマンスを示した．PIR 方式および PRP 方式は *join_ratio* の頻度にかかわらず，同様な傾向を示し，わずかではあるが，PIR 方式に比べ PRP 方式は不安定なネットワークでのパフォーマンスが優れている．PN 方式を用いた場合，安定したネットワーク環境下では他の提案方式とあまり変わらない結果を示したが，高頻度の *join_ratio* に対して他の提案方式よりもパフォーマンスの低下を抑えることが可能であった．PN 方式を用いた場合，あるピア p の隣接ピア p_i が leave したとしても，p_i の隣接ピアには p_i が保有しているデータの複製を保有している可能性が高く，*join_ratio* の影響を軽減することが可能である．

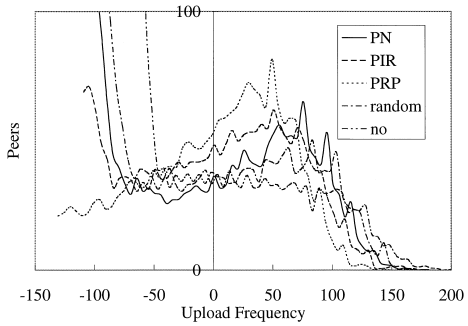
query_ratio の変動が問合せ処理へ与える影響を**図 5** に示す．データ更新の頻度が高い場合．データの再配置の処理を行うための遅延が生じるため，非同期であ

るデータが増加し，システムへの悪影響を与える．しかしながら，データ更新時のデータ再配置の遅延の発生にもかかわらず，$query\_ratio = 0.5$ のとき，No-Replication を用いた場合以外の方式では多くの検索結果を得ることができた．本研究では，各ピアはいったん複製を他のピアへ配置した後に，データが更新されるまで，データの再配置を行わない．そのため，ピアの leave にともないシステム全体のデータ数は減少してしまう．一度 leave したピアが再び join を行うとき，以前の隣接ピアと再び隣接するとは限らない．そのため PN 方式では，データ更新の頻度が低い場合では，ピア p から近くに位置するピアは p が嗜好するデータを保有する確率は低下する．また，データ更新頻度が低くなるほど，PN 方式でのデータ分布はランダム方式のデータ分布に近づくと考えられる．そのため，データ更新がほどよく発生することにより，問合せ処理のパフォーマンスは向上する．PIR 方式ではデータ更新の頻度が低い状況下でも，他の複製配置方式と比較して，その影響を受けにくいという結果を得た．PIR 方式では，各ピアから離れた距離に位置するピアにも複製を配置するため，データ分布はデータ更新の頻度にあまり影響されないと予想できる．PRP 方式でも同様の傾向になると考えている．

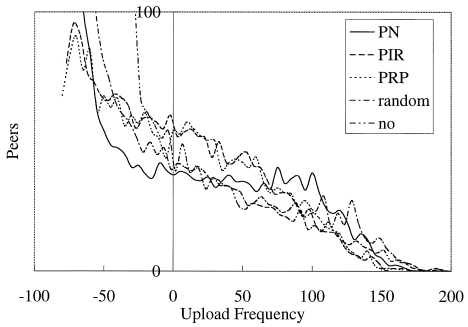
3.2.2 ロードバランシング

図 6 は各ピアのアップロード頻度を示している．横軸は各ピアにおける問合せ処理に基づくデータ転送の要求を受けることで生じるアップロードの回数をアップロード頻度の偏差で正規化した値を示している．縦軸はそのアップロード頻度を持つピアの数を示している．つまり，平均値（正規化されたアップロード頻度 = 0）にピアが集中しているほど良いロードバランスを行うことを示すことになる．

図 6 に示されるように，ピア数が 10,000，データのカテゴリ数が 20 のとき，PIR 方式および PRP 方式では各ピアに均等にアップロードの要求があり，優れたロードバランシングを行うことが分かった．PN 方式では，ほぼ特定されたピアへ複製を配置しようとする．PIR 方式および PRP 方式では，限定されたピア集合の中からピアを選択し，そのピアへ複製を配置しようとする．その結果 PIR 方式および PRP 方式は PN 方式に比べ広範囲に分布するようにデータを配置することが可能になる．ランダム方式では，PRP 方式および PIR 方式の場合よりも広範囲に複製を分布させるが，配置するデータとまったく関連しないピア上へ配置することもある．よって，ランダム方式のロードバランシングの状況は良くない．



(a) $num\_peer = 10000$, $num\_topic = 20$



(b) $num\_peer = 10000$, $num\_topic = 100$

図 6 アップロード頻度
Fig. 6 Upload frequency.

データのカテゴリ数を増加させた場合，PIR 方式および PRP 方式ではアップロード頻度の低いピア数が明らかに増加したが，他の配置方式よりも優れたロードバランスをが得られた．また他の変数を変動させた場合に比べ，PIR 方式と PRP 方式のパフォーマンスの差はほとんどなくなった．その要因として，両配置方式とも選択した複製配置対象のピアは同じである可能性が高いためであると考えられる．

4. おわりに

協調作業を P2P システム上で行うためには効率的なデータ共有を行う環境が求められている．そこで本研究では，P2P システム上でのデータ共有を行うために効率的なデータ配置方法を考案した．さらに，データ配置のシステムへの効果を確かめるため，各種実験を行った．これらの実験では問合せ処理およびロードバランシングの状況を計測し，この結果より提案したデータ配置方式の中でも PRP 方式は，効率的な問合せ処理およびロードバランシングを実現する．また実験結果より，長期間 leave の状況にあるピアは問合せ処理に対して弊害を与えることが分かった．そのため，今後は長期間 leave の状況にあるピアの対策を検討したいと考えている．

参 考 文 献

1) Aberer, K.: P-Grid: A self-organizing access structure for P2P information systems, *CoopIS 2001*, Trento, Italy (2001).
2) Chen, Y., Katz, R. and Kubiatowicz, J.: Dynamic Replica Placement for Scalable Content Delivery, *IPTPS'02*, Cambridge, USA (2002).
3) Cooper, B.F. and Garcia-Molina, H.: Bidding for storage space in a peer-to-peer data preservation system, *IEEE ICDCS 2002*, Vienna, Austria (2002).
4) Datta, A., Hauswirth, M. and Aberer, K.: Updates in Highly Unreliable, Replicated Peer-to-Peer Systems, *IEEE ICDCS 2003*, Rhode Island (2003).
5) FreenetProject. http://freenet.sourceforge.net/
6) Gnutella. http://www.gnutella.com/
7) GrooveNetworks. http://www.groove.net/
8) ICQ. http://web.icq.com/
9) Napster. http://www.napster.com/
10) Ng, W.S., Ooi, B.C., Tan, K. and Zhou, A.: PeerDB: A P2P-based System for Distributed Data Sharing, *IEEE ICDE 2003*, Bangalore, India (2003).
11) OpenNap. http://opennap.sourceforge.net/
12) Ratnasamy, S., Francis, P., Handley, M., Larp, R. and Shenker, S.: A scalable content-addressable network, *ACM SIGCOMM 2001*, San Diego, USA (2001).
13) Stoica, I., Rorris, R., Karger, D., Kaashoek, M. and Balakrishnan, H.: Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications, *ACM SIGCOMM 2001*, San Diego, USA (2001).
14) Sun, Q. and Garcia-Molina, H.: Partial Lookup Services, *IEEE ICDCS 2003*, Rhode Island, USA (2003).
15) Zhao, B., Kubiatowicz, J. and Joseph, A.: Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and Routing, Technical Report, U.C. Berkeley Technical Report UCB//CSD-01-1141 (2000).

(平成 15 年 9 月 25 日受付)
(平成 16 年 1 月 19 日採録)

(担当編集委員 石川 博, 市川 哲彦, 原 隆浩, 佐藤 聡, 土田 正士)

山田 太造(学生会員)

1977 年生. 総合研究大学院大学数物科学研究科博士後期課程在学中. データベースシステム, 分散処理, 情報検索等に興味を持つ. 日本データベース学会学生会員.

相原 健郎(正会員)

1969 年生. 1992 年横浜国立大学工学部卒業. 1997 年東京大学大学院博士課程修了. 博士(工学). 同年学術情報センター助手, 国立情報学研究所助手を経て, 2004 年 4 月より同研究所助教授. この間 2003 年より University of Colorado at Boulder 訪問研究員. 人工知能, 発想支援システムの研究に従事. 特に人間の創造性を伸ばすことを目的とする知的活動支援の理論, 方策に興味を持つ. 人工知能学会, 日本認知科学会, ACM 各会員.

高須 淳宏(正会員)

1984 年東京大学工学部航空学科卒業. 1989 年同大学院工学系研究科博士課程修了. 工学博士. 同年学術情報センター研究開発部助手. 1993 年より同センター助教授. 2000 年より国立情報学研究所助教授. 2003 年より同研究所教授. データ工学, 電子図書館, 機械学習の研究に従事. 電子情報通信学会, 人工知能学会, 日本データベース学会, ACM, IEEE 各会員.

安達 淳(正会員)

1981 年東京大学大学院工学系研究科博士課程修了. 工学博士. 東京大学大型計算機センター助手, 文部省学術情報センター研究開発部助教授, 教授を経て現在国立情報学研究所教授. 東京大学大学院情報理工学研究科教授を併任. データベースシステム, 分散処理システム, 情報検索, 電子図書館システム等の開発研究に従事. 電子情報通信学会, IEEE, ACM 各会員.