2006－EVA－18　(1)
2006／8／1

分散ハッシュテーブル型P2Pシステムにおけるブルームフィルタを用いた高速連言検索手法の評価

田頭 茂明　小畠 功士　藤田 聡

広島大学 大学院工学研究科

分散ハッシュテーブル型 P2P システム (P2P DHT) は，分散ネットワーク上に存在する共有資源の管理とキーワード検索とをピア・ツー・ピア (P2P) で効率よく実現するための代表的な手法の一つである．本稿では，P2P DHT 上で AND 検索を効率よくサポートするための新しい手法の提案と評価を行う．提案手法では，ブルームフィルタ (Bloom filter) を用いて，各ピアがローカルにキャッシュする過去の検索結果の情報を伝搬し，ピア間でそれらのキャッシュを効果的に共有する．提案手法の有効性を，シミュレーションにより実験的に評価した．実験の結果，システムに対する問合せ回数で約 66%改良できることを確認した．

Evaluation of a Fast Method for Supporting Conjuctive Quries Using Bloom Filters in P2P DHT

Shigeaki Tagashira　Koji Kobatake　Satoshi Fujita

Graduate School of Engineering, Hiroshima University

P2P DHT (Peer-to-Peer Distributed Hash Table) is one of typical techniques for realizing an efficient management of shared resources distributed over a network, and a keyword search over such networks in a fully distributed manner. In this paper, we propose a new method for supporting conjunctive queries in P2P DHT. The proposed method is based on a sharing of global information on past trials by locally caching the search result for conjunctive queries. The idea of the method is to reduce the number of meaningless inquiries by registering cached conjunctions to the bloom filter, and by circulating it among all peers. The effect of the proposed method is experimentally evaluated by simulation. The result of experiments indicates that by using the proposed method, the number of inquiries is reduced by 66%, compared with a method which does not use the bloom filter.

1　はじめに

P2P システムではネットワーク上に存在する情報 (コンテンツ) が分散的に蓄積されるため，それらのコンテンツに対する適切な**インデックス**，すなわち，どのようなコンテンツがどこに存在するかを示す情報が，ユーザに対して利用可能な形で提供されている必要がある．P2P システム上でインデックスを提供するための方法として，分散ハッシュテーブル型 P2P システム (P2P DHT)[1, 2] が提案されている．

P2P DHT では，システム上に蓄積されている各コンテンツに関するインデックス情報が，そのコンテンツに付与された**識別名**によって分散管理される．識別名はコンテンツのファイル名等であってもよいが，通常は，そのコンテンツと関連のある複数のキーワードをそれぞれ識別名として登録し，インデックス情報のキーワードによる検索を実現する方法が用いられる (逆 DHT 法 [3])．したがって本稿では，(一般性を失うことなく) キーワード検索が可能な P2P DHT を前提とする．そのようなキーワード検索における重要な問題点の一つに，同一のキーワードにマッチするコンテンツ数が多くなると所望のインデックス情報がなかなか得られなくなってしまう点がある．この問題は，条件付き検索を行うことである程度解決することができるが，これまでに提案されているほとんどの P2P DHT は 2 語以上のキーワードを含む検索には対応していない．

本稿では，代表的な条件付き検索法である AND 検索を P2P DHT 上で効率良く実現するための新しい手法の提案と評価を行う．提案手法は，各ピアが過去に行った AND 検索の結果をローカルなディスク上にキャッシュす

るシステムを対象としている (リザルトキャッシュ[6]). このような環境において，提案手法ではキャッシュされた連言に関する情報が書き込まれたブルームフィルタ (Bloom filter)[3, 4] をピア間で交換・共有することで，ある連言がキャッシュ上に存在していないことを，ほかのピアに対する問合せを行うことなく確かめることができる．

本稿では，提案手法の効果をシミュレーションにより実験的に評価する．実験の結果，従来手法と比べて問合せ回数を約66%減少させることができた．なお本稿の実験では基盤となるP2P DHTとしてCAN (Content Addressable Network) を想定しているが，提案手法はP2P DHT全般に対して適用可能である．

2 モデル

2.1 P2P システム

本稿で想定するP2Pシステムのモデルは以下の通りである．各ピアが複数のコンテンツを保持しており，それらのコンテンツがシステムに参加しているすべてのピアによって共有されている状況を考える．各コンテンツにはそのコンテンツと関連のある複数のキーワードが付与されており，そのキーワードを，それぞれコンテンツの識別名として登録することができる．

本稿では，どのような識別名のコンテンツがどのピアによって保持されているかを示す情報のことを**インデックス**と呼ぶ．具体的には，各インデックスは識別名と値の二項組であり，値には，そのコンテンツを保持しているピアの名前やIPアドレスなどが入る．したがって，もし与えられた識別名からその識別名に対応するコンテンツの保持者が引き出せるような “インデックスのテーブル” があらかじめ用意されていれば，所望のコンテンツを保持しているピアの名前とアドレスをそのテーブルを用いて効率良く探り当てることができることになる．以下で述べるP2P DHTは，そのようなインデックステーブルをP2Pシステム上で分散的に実現するための一つの方法を与えている．

2.2 P2P DHT

P2P DHTはDHT(分散ハッシュテーブル) を用いたインデックス管理手法である．ここで，ハッシュテーブルとはインデックスを格納するための論理的な空間のことを指し，DHTとは，そのようなハッシュテーブルを分散的に実現する手法である．

各インデックスのハッシュテーブル上のポイントへの割り当ては，識別名空間からハッシュテーブルへの適切なハッシュ関数を用いて行われる．P2P DHTに参加する各ピアは，ハッシュテーブル (仮想的な論理空間) の一部分に写像されたインデックス情報を管理している．各ピアは，自分の受け持ち部分と隣接する部分を管理しているピア (隣接ピア) の情報も保持しており，隣接ピア同士で論理的な双方向リンクを形成している．よって，ハッシュテーブル上の任意のポイントを管理しているピアへのメッセージ送信は，隣接ピアの中でそのポイントに一番近いピアに向けてメッセージをバケツリレー式に転送し続けることで行われる．

P2P DHTでは，インデックスの登録と検索が，それぞれ基本的なオペレーションとして定義されている．インデックス検索は，識別名からハッシュ関数によって計算されるポイントに向けて検索要求メッセージ (クエリ) を送ることで実現され，一方インデックスの登録は，そのコンテンツが生成された時に，ハッシュテーブル上の対応するポイントに向けて登録要求メッセージを送ることで実現される．

2.3 検索モデル

本稿では，各コンテンツに付与される識別名のことを**キーワード**と呼び，キーワード全体の集合を $S = \{a_1, a_2, \ldots, a_i, \ldots\}$ と記すことにする．システム中に蓄積されているコンテンツの中でキーワード a_i が付与されているコンテンツの集合を $C(a_i)$ と記す．したがってP2P DHTでは，キーワード a_i をハッシュ関数によって写像して得られるハッシュテーブル上のあるポイントの上に，$C(a_i)$ に属する各コンテンツがそれぞれどのピアによって保持されているかというインデックス情報がすべて格納されていることになる．

以下では，ユーザから提示されるすべてのクエリは，複数のキーワードの連言 (conjunction) として与えられるものとする．よって各クエリは，それを構成するキーワードの集合によって一意に特定することができ，以下では，k 個のキーワード $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ を含むAND検索要求のためのクエリを，$Q = \{a_{i_1}, a_{i_2}, \ldots, a_{i_k}\}$ のように集合として表現することにする．ここでこのクエリ Q が，集合

$$C(a_{i_1}) \cap C(a_{i_2}) \cap \cdots \cap C(a_{i_k}) = \bigcap_{a \in Q} C(a)$$

に属するすべてのコンテンツのインデックス情報を要求しているという点に注意されたい．

以下では，ユーザから提示される各クエリに含まれるキーワードは，キーワード集合 S から一定の偏りをもって選択されるものとする．さらに，各コンテンツに対するキーワードも，同様の偏りで選択され付与されるものとする．この仮定は，検索要求に含まれるキーワードには一種の人気度があり，人気のあるキーワードは非常に多くのクエリに含まれるが，そうでないキーワードはほとんどのクエリにあらわれない，という状況を想定している．本稿ではそのような偏りの具体的なモデルとして，ジップの法則 (Zipf's Law) を用いることとした[1)]．ジップ

[1)] どのような偏りであるかが，方式の基本的なデザインには影響を与えないことに注意されたい．分布の影響は，方式の中で用いられるパラメータのチューニングの際に大きくあらわれることになる．

の法則とは，“最も人気のあるキーワードの出現頻度が p のとき，i 番目に人気のあるキーワードの出現頻度が p/i になる” という法則であり，文字列を取り扱う様々な分野で観測されている一種の経験則である．以下では一般性を失うことなく，すべてのクエリの集合の中で i 番目に出現頻度の高いキーワードが a_i であるとする．また同一のクエリ内に含まれるキーワード間の相関関係については，本稿ではまったく考慮しないものとする．したがって 4 節の実験でクエリを構成する際には，必要な個数のキーワードをジップの法則にしたがって独立に選択していくことになる．

2.4 リザルトキャッシュを用いた連言検索

提案手法では，P2P DHT 上で，リザルトキャッシュを用いて AND 検索をサポートするシステム [5] を対象としている．リザルトキャッシュ(result cache) とは，システム上で過去に得られた検索結果をキャッシュ上に一時的に保存することで，処理に時間のかかる問合せの無駄な繰り返しを防ぐ技術である [6]．P2P 上の AND 検索処理にリザルトキャッシュを用いることの利点は以下の二点である．一つはシステムへの問合せ回数の減少である．もう一つは，クエリに対して返されるインデックス量 (返送インデックス量) が，連言を重ねれば重ねるほど減少していくことである．

リザルトキャッシュを用いた AND 検索は，基本的には，通常の P2P DHT で行われているような単一のキーワードに関する情報のハッシュテーブル上への登録に加えて，検索結果をキャッシュしているピアに関するインデックス情報を，対応するクエリに対してハッシュ関数をかけて得られたポイントに格納するようにすることで実現することができる．ただし本研究が対象としてシステムでは，リザルトキャッシュを用いた AND 検索の効率を向上させるため，下記の 2 点に留意している．

システム全体のコストパフォーマンスを向上させるため，各ピアがキャッシュする連言の長さを適切な値以下に制限する．以下ではこの値のことを**最大連言長**と呼び，記号 L で表現する．最大連言長を適切に設定することで，キャッシュに対するヒット率を低下させることなく，メモリの利用効率の大幅な向上が期待できる．

最大連言長が設定された P2P システム上の AND 検索では，もし与えられたクエリの長さが L 以下であれば，そのクエリに対する検索結果がキャッシュされている可能性があるため，クエリ全体に対する問合せをシステムに対して直接試みてみる価値がある (もちろん必ずキャッシュされているという保証はない)．その一方で，L よりも長いクエリについては，クエリ全体に対する検索結果がキャッシュされている可能性がまったくないことから，与えられたクエリを適切な長さの連言の集合に分割し，それらの連言たちに対する検索結果をすべて得た後でそれらを統合する，という段階的な方法をとることになる．そのため提案手法では，キャッシュ対象をユーザから出されたクエリに限定せず，もとのクエリの一部としてあらわれる連言のそれぞれに対しても，(必要があれば) 適宜キャッシュしていくことにする．

3 提案手法

3.1 ブルームフィルタ

提案手法では，検索の候補となる連言のうち，どのような連言がキャッシュされていないのかを，他のピアに対して検索要求を出す前に，ブルームフィルタを用いて各ピアに局所的にチェックさせることを提案する．要素の全体集合を U とし，集合 U の任意の (有限) 部分集合を S とする．ブルームフィルタ (Bloom filter: BF) の目的は，集合 U のある要素 z が集合 S に含まれているか否かを，長さ m ($<< |U|$) のビット列で可能な限り正確に表現することである [3, 4]．以下では，長さ m のビット配列を B であらわし，これをブルームフィルタと呼ぶ．また以下の議論では，集合 U から集合 $\{1, 2, \ldots, m\}$ への k 個の異なるハッシュ関数 $h_1, h_2, \cdots, h_k$ の存在を仮定する．

集合 S を B に “記録” するための手続きは以下の通りである：

Step 1: B のすべてのビットを 0 に初期化する．
Step 2: 集合 S の各要素 z に対して，以下のことを行う：

1. k 個のハッシュ値 $h_1(z), h_2(z), \ldots, h_k(z)$ を求める．
2. 各 i ($1 \leq i \leq k$) に対して，もし $B[h_i(z)] = 0$ ならば $B[h_i(z)]$ の値を 1 に更新する．

一方，ある要素 $y \in U$ が S に含まれているかどうかを，S を “記録” したビット配列 B を用いて判定する手続きは以下の通りである：

Step 1: k 個のハッシュ値 $h_1(y), h_2(y), \ldots, h_k(y)$ を求める．
Step 2: もしすべての i ($1 \leq i \leq k$) に対して $B[h_i(y)] = 1$ ならば，$y \in S$ である．そうでなければ，S は y を含んでいない．

ブルームフィルタには偽陽性 (false positive) がある．偽陽性とは，実際は y が S に含まれていないにも関わらず “$y \in S$” と誤判定してしまう性質のことである．偽陽性の誤判定をしてしまう確率のことを偽陽性率 (false positive rate) という．ブルームフィルタの偽陽性率は，n(S の要素数), m, k などのパラメータに依存していることが，これまでの研究からわかっている [3, 4]．また集合 S_1, S_2 ($\subset U$) に対する同一ハッシュ関数に基づくブルームフィルタ B_1, B_2 が与えられたとき，和集合 $S_1 \cup S_2$ に対するブルームフィルタは，B_1 と B_2 のビットワイズな論理和をとることで容易に得ることができる．

3.2 リザルトキャッシュへの応用

ブルームフィルタを提案手法に応用することで，ある連言がリザルトキャッシュ上に存在していないことを，ほかのピアに対する問合せを行うことなく，局所的な計算のみによって確かめることができる．したがって，もし偽陽性率が十分低く，かつ十分高い精度をもったブルームフィルタを用いることができれば，無駄な問合せを大幅に減少させることができるはずである．以下では，前述の全体集合 U を可能な連言全体の集合とし，S を現在キャッシュ上に存在する連言の集合とする．また適切な k 個のハッシュ関数を用いて S を記録して得られた長さ m のビット列を，B であらわすことにする．

共有メモリをもたない通常の P2P 環境では，各ピアがそれぞれプライベートなブルームフィルタのコピー (以下，単にコピーと呼ぶ) を所有し，各ピアが実行した連言のキャッシュ履歴は，各自のもつコピーに記録された後で，ほかのピアに対して順次伝播されていくことになる．提案手法では，そのようなコピーの伝播に，適切に構成された根付木状のオーバーレイネットワーク T を用いる．提案手法におけるコピーの伝播は，以下のような方針で進められる：

- ピア u は，ほかのピアへの情報伝達用に用いられるコピーのほかに，自分が実行した履歴のみを反映したオリジナルのコピー B_u を保存する．コピー B_u は，ピア u が自身のキャッシュ上に保存した連言集合が変化したときに u 自身の手で更新される．特に，保存していた連言がキャッシュから除去された際には，ピア u は B_u を 0 ベクトルに初期化し，その内容を現在のキャッシュ上の連言集合から再構成する．
- ピア u は，ネットワーク T 上の各接続リンクごとに，そのリンクの先にあるピア集合全体で保存されている連言集合を反映したコピーを用意する．そのようなコピーを $B_{u,1}, B_{u,2}, \ldots, B_{u,k}$ としたとき，ネットワーク全体で保存されている連言集合は，u 自身の局所的な計算により，$B_u \vee \left(\vee_{i=1}^{k} B_{u,i}\right)$ のように求めることができる．
- コピー B_u が更新されたとき，その更新情報は，ネットワーク T 上のすべての隣接ピアに向けて転送される．ただし更新情報の転送は情報更新後すぐに開始するのではなく，一定間隔で刻まれるクロックに同期して開始される．
- コピー $B_{u,i}$ に対応する隣接ピア v から更新情報 B^* を受け取ったとき，ピア u は $B_{u,i}$ を B^* で置き換えるとともに，v 以外の隣接ピアに向けて，その更新を反映したコピーを転送する．具体的には，コピー $B_{u,j}$ に対応する別の隣接ピア w に対しては，置き換えられた $B_{u,i}$ を用いて $B_u \vee (\vee_{h \neq j} B_{u,h})$ を計算し，その結果を更新情報として転送する (同様の処理を v, w 以外のすべての隣接ピアに対しても行う)．

3.3 手続き

本節では，上述の基本アイデアにしたがって AND 検索を行うための具体的な手続きを示す．以下では，1) リザルトキャッシュを用いた AND 検索処理と，2) リザルトキャッシュの更新処理，そして 3) ブルームフィルタの更新手続きについて順に説明していく．なお以下の説明では，ユーザから与えられたクエリを Q とし，検索処理の途中で生成される連言 (Q の部分集合) を p, q などの記号を用いてあらわす．また，ブルームフィルタに関する記法は，前節の記法に準じるものとする．

3.3.1 AND 検索処理

クエリ Q を提示したピア上で実行される AND 検索の具体的な手続きは以下の通りである．

procedure SEARCH(Q)

1. $q \leftarrow Q$ とする．
2. 変数 j を $\min\{L, |q|\}$ に初期化する．ただし，L は最大連言長とする．
3. 連言 q の長さ j のプレフィックスを p とする．
4. ブルームフィルタを用いて，$C(p)$ がハッシュテーブル上に保存されている可能性があるかどうかをチェックする．保存されている可能性があるときのみ，連言 q とともにシステムに対して問合せる．キャッシュされていないと判断した場合はすぐ下の 4a へ．
   (a) $C(p)$ が保存されていないとき: $j = 1$ ならば検索失敗で終了．そうでなければ，j の値を 1 減らして 3 へ戻る．
   (b) $C(p)$ が保存されているとき: もし $C(p)$ のインデックスとともにシステムから返された集合 $\hat{S} = \{q' \in S^*(p) \mid q \cap q'\}$ の要素が p のみであれば，p に関するインデックスを獲得して保存する．そうでなければ集合 $\hat{S}$ の中で最も長い連言を新たに p とし，p に関するインデックスをキャッシュ先から獲得して保存する．
5. $q \leftarrow q \setminus p$ とする．もし $q = \emptyset$ ならば，それまでに獲得した集合の積をとったものを検索結果として出力して終了．そうでなければ 2 へ戻る．

この手続きの終了条件は，検索に失敗したとき ($j = 1$ で $C(p)$ が保存されていないとき) か成功したとき (q が空になったとき) の 2 通りであるが，そのいずれの場合も，検索途中で獲得したインデックス集合が，クエリを提示したピア上に局所的に保存されていることに注意されたい (クエリ中の最初のキーワードに対する検索結果が空である場合を除く)．そのようなインデックス集合を獲得順に $C_1, C_2, \ldots, C_\ell$ とし，対応する連言をそれぞれ $q_1, q_2, \ldots, q_\ell$ とする (手続きの記述よりあきらかに，$\bigcup_{i=1}^{\ell} q_i = Q$ である)．なお以下の説明では，獲得された連言の集合を $\mathcal{Q} = \{q_1, q_2, \ldots, q_\ell\}$ とおく．

3.3.2 リザルトキャッシュの更新処理

リザルトキャッシュの更新は，前述の検索手続きの終了直前に実行される．更新処理ではまず最初に，連言集合 $\mathcal{Q}$ を次の条件を満たすように部分集合 $\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_x$ に分割する：

条件 1: 各 $1 \leq i \leq x$ について，$|\bigcup_{q \in \mathcal{Q}_i} q| \leq L$ であること．

条件 2: 各 $1 \leq i < x$ について，部分集合 $\mathcal{Q}_{i+1}$ に含まれる最も小さい添字をもつ連言を q_j としたとき $|\bigcup_{q \in \mathcal{Q}_i} q| + |q_j| > L$ であること．

この分割が，サイズ L のビンの集合に $\mathcal{Q}$ に含まれるアイテムたちを貪欲に詰め込んで得られる，一種のビンパッキングになっていることに注意されたい．

リザルトキャッシュの更新は，各部分集合 $\mathcal{Q}_i$ ごと独立に．次の手続きに従って行われる：

1. $\mathcal{Q}_i = \{q_{i_1}, q_{i_2}, \ldots, q_{i_y}\}$ とする．
2. 各 $1 \leq j \leq y$ に対して $\tilde{q}_j = q_{i_1} \cup \cdots \cup q_{i_j}$ とする．
3. 各 $1 \leq j \leq y$ に対して $C(\tilde{q}_j)$ を計算し，その結果を自ピアのリザルトキャッシュ上に局所的に保存するとともに，そのことをハッシュテーブル上に連言 $\tilde{q}_j$ をキーとして登録する ($\tilde{q}_1$ を登録するときのみ，$\tilde{q}_2, \ldots, \tilde{q}_y$ がハッシュテーブル上に登録されていることを $\tilde{q}_1$ のインデックス情報として追記することに注意)．

3.3.3 ブルームフィルタの更新処理

最後に，各ピアが持つブルームフィルタの更新処理について説明する．更新処理は，大別すると以下の 2 つの処理に分けることができる：

1. キャッシュ内に保持する連言集合を修正したときのブルームフィルタ B_u の更新．
2. リンク先のピアからブルームフィルタの更新情報が送られてきた場合の処理．

前者の処理は，ピア u が保持している連言集合に変化があった場合に直ちに実行される．具体的には，以下の手続きが行われることになる：

1. 連言を追加したとき：B_u に，追加された連言を記録する．
2. 連言を削除したとき：B_u を 0 ベクトルに初期化し，削除後の連言集合から B_u を再構築する．

一方後者の更新処理は，リンク先のピアからブルームフィルタの更新情報 B^* が送られてきた場合に，対応するコピー $B_{u,i}$ を最新のものに更新するだけでよい．

なお各ピアは，隣接するピアに向けてブルームフィルタの更新情報を一定間隔で送信する．その際，リンク先のピア v へは，ピア v 以外のブルームフィルタの論理和をとったものが送信される．具体的には，ピア v に対応するコピーを B_j とすると，送信されるコピーは $B_u \vee (\vee_{h \neq j} B_{u,h})$ となる．

4 評価

本章では，提案手法の効果をシミュレーションにより評価する．以下では AND 検索手法の評価項目として，返送インデックス量とシステムへの問合せ回数に着目する．ただし返送インデックス量は 1 クエリあたりの返送データ量によって評価するものとし，問合せ回数はクエリあたりの検索要求の送信回数で評価される．

4.1 シミュレーションモデル

本章の評価では，具体的な P2P DHT として二次元 CAN[1] を想定する．あらかじめ固定された数のピア (以下の実験では 256 に固定) が CAN 上にすでに参加しており，ピアの追加や離脱が起こらないような状況を考える．またすべてのコンテンツはあらかじめピアたちによって共有されており，コンテンツの追加や削除も発生しないものとする．システム上の各ピアはネットワークを介して高速に通信できるものとし，CAN 上のピア間通信に要する時間は，ホップ数とデータ量にそれぞれ比例するものと仮定する．

シミュレーションでは，各ピアがそれぞれ独立にクエリを提示し，そのクエリ提示間隔は平均 $\lambda = 20$ のポアソン分布に従うものと仮定する (以下では 1 単位時間を実時間の 15 秒と対応させている)．また 1 回のシミュレーションで提示されるクエリ数を 5000 回にそろえる．

各ピアが保持するキャッシュのサイズは，以下のシミュレーションではパラメータとして与えられる (ここで，キャッシュサイズの単位はインデックス数である)．キャッシュ置換アルゴリズムは LRU(Least Recently Used) を使用する．またハッシュテーブル上に登録された情報の正しさを維持するため，自身のキャッシュ上からある連言に関するインデックス情報を除去したピアは，そのことをハッシュテーブル上の情報に正しく反映させているものとする．ここでハッシュテーブルの情報に反映させるためのメンテナンスコストは，無視できるほど小さいものと仮定する (ブルームフィルタの交換にかかるコストは別途評価する)．

各クエリ中に含まれるキーワードは，次のような方法で選択される．キーワード集合 S のサイズを 2,500 とする．まず，各ピアが保持するコンテンツ数を 100，各コンテンツに付与されるキーワード数を 20 とし，各クエリに含まれるキーワード数を，確率 1/4 で停止するベルヌーイ試行によってそれぞれ独立に決定する．(ただしあまり長いクエリは現実的ではないことから，便宜上，連言長が 10 を超えると，同様の試行をキーワード数 2 から繰り返すことにする)．次に，決定されたキーワード数が X のとき，X 個のキーワードがジップ (Zipf) の法則にしたがって集合 S から重複することなく独立に選択され，そのコンテンツもしくはクエリに付与されていく．

また本稿では，ジップの法則によって実現されるキーワード選択の局所性に加えてクエリ選択の局所性にも注

表 1: 3 つの手法の比較.

	返送インデックス量 [個]	問合せ回数 [回]
NR 法	23625.652	3.4532
RC 法	11285.268	7.4574
BF 法	17445.209	3.0188

目し，以下のような方法で局所性のあるクエリ選択を実現する．クエリ選択の局所性はワーキングセットの概念を用いて実現される．ここでいうワーキングセットとは，集合 S 中の上位 10%のキーワードたちのみから前述の方法にしたがって構成される特別なクエリの集合のことであり，以下の実験では，ワーキングセット中のクエリ数を 500 に固定する．各ピアからシステムに対して提示されるクエリは，30%があらかじめ構成されたワーキングセットからランダムに選択され，残りの 10%が S 全体からキーワードを拾っていく前述のような形で構成される．

4.2 実験結果

4.2.1 ブルームフィルタの効果

まず最初に，AND 検索をサポートせず DHT だけを用いた従来手法 (以下，NR 法) とリザルトキャッシュだけを用いた提案手法 (以下，RC 法) と RC 法にブルームフィルタを加えた手法を BF 法を比較することで，ブルームフィルタの効果を評価する．結果の一例として，ブルームフィルタの交換開始インターバルを 5 分に設定したときの性能の比較を表 1 に示す 1 クエリあたりの返送インデックス量と問合せ回数をそれぞれ評価した．RC 法における最大連言長を 6，キャッシュサイズを 5000 としている．表よりあきらかに，BF 法の返送インデックス量は RC 法と比べると増加しているが，その問合せ回数は，他のいずれの方法よりも少なく抑えられていることがわかる．すなわちブルームフィルタを用いることで，ハッシュテーブル上に登録されていない連言に対する無駄な問合せが大幅に減少することがわかる．以下では，BF 法の効果をより詳細に調べるため，以下の各項目について実験的に評価することにする：

(i). ブルームフィルタが十分大きく，各ピアが自分のコピーに対して行った変更が他のピアに対して瞬時に伝わるという理想的な状況のもとで，キャッシュサイズの変化が方式全体の性能にどのような影響を与えるのか

(ii). ブルームフィルタが十分大きいとき，ブルームフィルタの交換開始インターバルの変化が方式全体の性能にどのような影響を与えるのか

4.2.2 キャッシュサイズの影響

理想的な状況でキャッシュサイズのみを変化させたときの様子を図 1 に示す．図の横軸はキャッシュサイズであり，(a) と (b) の縦軸はそれぞれクエリあたりの返送インデックス量と問合せ回数である．また図中の曲線は，それぞれ設定された最大連言長 L に対応しており，例えば “level2” は L の値が 2 に設定された場合に対応している (以降の図においても同様)．まず図 (a) より，返送インデックス量は，キャッシュサイズの増加に伴って一定の値に収束することがわかる．また図 (b) より，問合せ回数についても返送インデックス量と同様の傾向が認められることがわかる．さらにどのキャッシュサイズに対しても，L が大きくなるほど少なくなるという傾向を示している．この結果は，(理想的な) ブルームフィルタには，L の増加によるデメリットを抑え，そのメリットを引き出す効果があることを意味している．

4.2.3 交換開始インターバルの影響

最後にブルームフィルタの交換開始インターバルが性能に与える影響について調べる．図 2 に結果を示す．ここで図の横軸は交換開始インターバル (分) をあらわす．まず (a) より，返送インデックス量は，交換開始インターバルが長くなるほど多くなることがわかる．これは，長いインターバルでは，連言がキャッシュされているにもかかわらず存在しないと誤判定してしまうケースが発生し，キャッシュが有効に用いられなくなるためであると考えられる．また，最大連言長 L が長くなるほど，その影響が大きくなることもわかる．また問合せ回数についても，それと同様の理由から，長いインターバルでは回数が増加する傾向にある．

4.2.4 ブルームフィルタのコスト

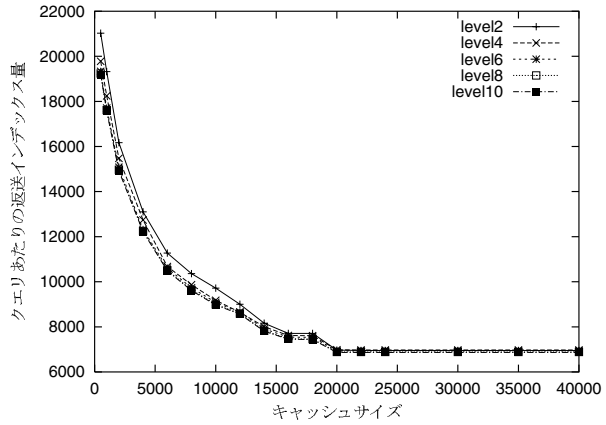
BF 法では RC 法に比べて，ブルームフィルタを交換するためのコストが余分にかかる．以下では，次の 3 通りの伝播方法に着目し，それぞれの伝播方法を用いたときに方式全体で必要となる転送データ量を調べる：

方法 1: 毎回の交換開始タイミングで，フィルタ全体を転送する．

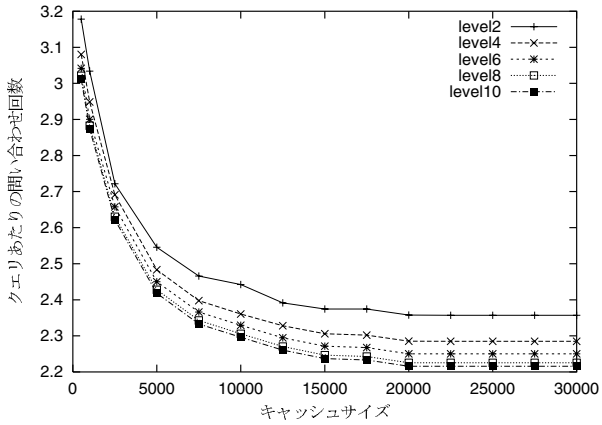
方法 2: 前回送信したフィルタと内容が変わっているときのみ，毎回の交換開始タイミングで，フィルタ全体を転送する．

方法 3: 前回送信したフィルタと内容が変わっているときのみ，毎回の交換開始タイミングで，差分情報のみを転送する．

ここで転送データ量は，返送インデックスにブルームフィルタの伝播にかかる転送データ量を加えた値とする．なお以下では，1 インデックスあたりのサイズを 100 バイトとし，交換されるブルームフィルタのサイズを 20000

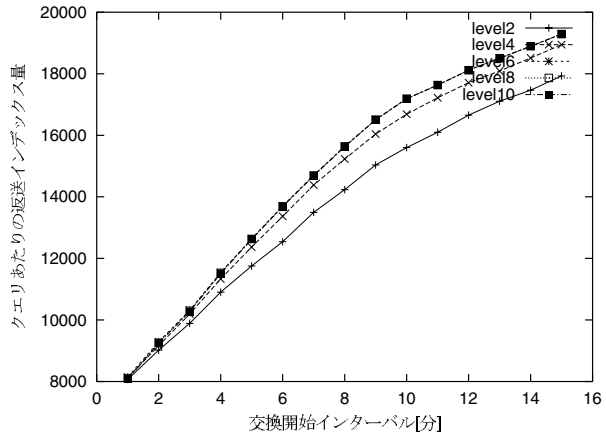


(a) クエリあたりの返送インデックス量.

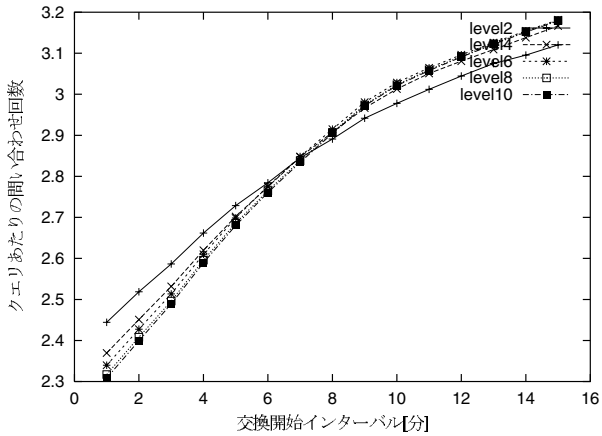


(b) クエリあたりの問合せ回数.

図 1: キャッシュサイズの影響.



(a) クエリあたりの返送インデックス量.



(b) クエリあたりの問合せ回数.

図 2: 交換開始インターバルの影響.

バイトとする．また，最大連言長を 6 に固定し，十分なサイズのキャッシュの存在を仮定する．

結果を図 3 に示す．図の横軸は交換開始インターバル (分) である．図中の各曲線はそれぞれの手法に対応しており，“NR” は NR 法，“RC” は RC 法，“方法 1” から “方法 3” は BF 法のうち上記の方法 1 から方法 3 までにそれぞれ対応している．図より，RC 法，BF 法ともに，転送データ量が NR 法に比べて減少していることがわかる．また方法 1 では，インターバルが短いときには転送データ量は減少し，逆に長いときは増加していることがわかる．したがって方法 1 では，インターバルが短いときはブルームフィルタを転送するためのデータ量の影響が支配的であり，長いときは返送インデックスにかかる転送データ量がブルームフィルタの伝播にかかる転送データ量に対して支配的となっていることになる．

一方，方法 2 と方法 3 では，インターバルが短い方が転送データ量が減少しており，返送インデックスにかかる転送データ量が支配的となっていることがわかる．図 2(a) にも示したように，返送インデックス量は交換開始インターバルの影響を強く受けており，交換開始インターバルを長くすると，転送データ量は RC 法と比べてかなり増加する．しかし交換開始インターバルを短くして方法 3 を用いることで，ブルームフィルタのコストを抑えつつ，問合せ回数を軽減することが可能となっている．

5 関連研究

本章では，P2P DHT 上でキーワード検索を効率的に行うための従来手法について概観する．

コンテンツ名から仮想論理空間への関数として定義されていたハッシュ関数を，各コンテンツに与えられたキーワードからの関数へと拡張する方法 (逆 DHT 法) は，Vahdat によってはじめて提案された [3]．2 章でも述べたように，本稿で対象とするシステムも，この逆 DHT に基づいて設計されている．

ブルームフィルタを用いて P2P 上を流れる転送インデックス量を圧縮する方法は，Vahdat によって提案されている [3]．ただしこの手法は，単一のキーワードに対する検

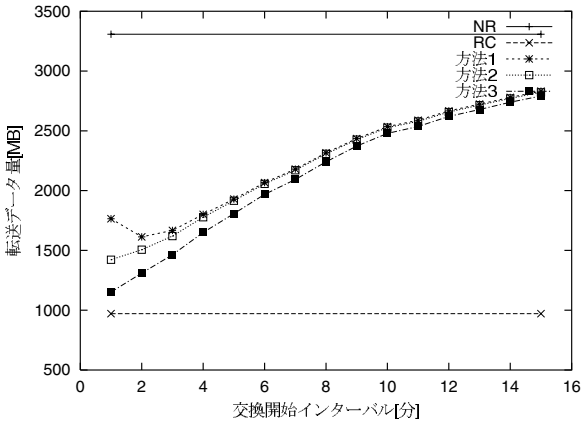


図 3: 交換開始インターバルを変化させたときの転送データ量.

索結果を転送する際の転送コストの削減を目指したものであり，本稿で提案しているブルームフィルタの利用方法とは異なっている．

連言の検索を効率よく実現するための方法として，これまでにいくつかの手法が提案されている．まず Bhattacharjee は，ビューツリー (view tree) を提案した [6]．ビューツリーは P2P DHT 上でリザルトキャッシュを実現するための手法の一つであり，キャッシュした連言を木構造状のオーバーレイネットワーク上で階層的に管理することで，AND 検索を効率的にサポートすることを目指している．しかし，この手法には以下の問題点がある：1) キャッシュされている連言が木の中のどこに存在しているかを明示的に管理していない．2) 木の各頂点が自分のすべての子頂点を正確に把握しなければならない．3) 頻出する連言とそれ以外とを区別していない．

Liu はキーワード融合 (keyword fusion) という手法を提案している [7]．キーワード融合の基本的なアイデアは，連言検索のクエリ中で頻出するキーワード (common keyword) をうまく切り出すことで，そのような頻出キーワードを管理しなくてはならないピアの負荷を軽減することである．しかし，“頻出キーワード単独での検索では，それに一致する全コンテンツを探し出すことが不可能になることがある” という問題も残る．

Gnawali は，連言検索をサポートするための方法の一つとして，キーワードセットという考え方を提案している [8]．しかしこの手法では，各コンテンツに付与されているキーワードから作成されるすべての連言を登録することになるために，検索されない連言についても登録が行われ，そのための無駄な作業やインデックス量の増加が問題となる．

6 おわりに

本稿では，P2P DHT 上で効率的な AND 検索を実現するための手法を提案し評価を行った．具体的には提案手法では，各ピアがどのような連言をキャッシュしているのかという情報をブルームフィルタに書き込み，その情報をピア間で共有することで，システムに対する無駄な問合せを削減している．本稿では，提案手法の効果をシミュレーションにより実験的に評価した．その結果，問合せ回数をリザルトキャッシュのみを適用する手法と比べて約 66%，減少させることができた．ただし，ブルームフィルタの偽陽性や伝播遅延により，返送インデックス量は，リザルトキャッシュのみを適用する手法と比べて約 7%増加した．今後の課題として，ピアの参加や離脱が頻繁に発生する動的なネットワーク上に提案手法を応用することや，より現実の環境に近いネットワークモデル上でシミュレーションを行うことが挙げられる．

参考文献

[1] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content Addressable Network. In *Proceedings of the ACM SIGCOMM 2001*, pages 161–172, August 2001.

[2] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of the ACM SIGCOMM 2001*, pages 149–160, August 2001.

[3] P Reynolds and A. Vahdat. Efficient Peer-to-Peer Keyword Searching. In *Proceedings of the 4th ACM/IFIP/USENIX International Middleware Conference*, pages 21–40, June 2003.

[4] B. Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, July 1970.

[5] 小畠，竹本，田頭，藤田, P2P システム上でのリザルトキャッシングを用いた条件付検索手法, 電子情報通信学会 IN 研究会，pages 7-12, 2004.

[6] B. Bhattacharjee, S. Chawathe, V. Gopalakrishnan, P. Keleher, and B. Silaghi. Efficient peer-to-peer searches using result-caching. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, pages 225–236, February 2003.

[7] L. Liu, K. D. Ryu, and K. W. Lee. Keyword Fusion to Support Efficient Keyword-based Search in Peer-to-Peer File Sharing. In *Proceedings of the 4th International Workshop on Global and Peer-to-Peer Computing*, pages 269–276, April 2004.

[8] O.Gnawali. A Keyword-Set Search System for Peer-to-Peer Networks. *Master's thesis, Massachusetts Institute of Technology*, June 2002.