LM-010

複数キーワード検索に対応した分散ハッシュ型P2P ネットワーク
Multi-Keyword Search for DHT P2P Networks

佐藤 一帆 † 松本 倫子 ‡ 吉田 紀彦 ‡
Kazuho Sato Noriko Matsumoto Norihiko Yoshida

1. はじめに

P2P（Peer-to-Peer）ネットワーク上で効率の良い検索を実現する手段として，分散ハッシュテーブル（Distributed Hash Table, DHT）を用いたさまざまな手法が提案されている．DHT 型 P2P では，ネットワークに参加するノードに対しノードのハッシュ値によって ID を割り当て，ある規則に従ってそれぞれのノードに他ノードの所在情報を管理させることでネットワークを構造化する．これにより，検索メッセージを目的のノード ID が割り当てられたノードに誘導することができる．

ところが，実際に DHT 型 P2P によってコンテンツを管理する場合，コンテンツに複数のキーワードを設定することが難しいという問題が起きる．DHT 型 P2P 上にコンテンツを追加するには，コンテンツのある値（タイトル，コンテンツのビット列など）のハッシュ値からコンテンツ ID を設定し，コンテンツ ID に一番近い ID を持つノードにコンテンツを管理させる．コンテンツを検索するには，検索するコンテンツ ID を求め，一番近い ID を持つノードに検索メッセージを誘導する．つまり，検索には目的のコンテンツのコンテンツ ID が必要である．このコンテンツ ID はハッシュ値を使って求めるので，ID を求めるためのハッシュ関数への入力が厳密に一致していなければならない．このため，複数のキーワードに対応することが困難となる．例えば，あるコンテンツに関する 2 つのキーワード A, B からハッシュ関数を用いて $Hash(A+B)$ という値をコンテンツ ID に設定することを考える．このとき，検索キーワード A, B を使って，$Hash(A), Hash(B), Hash(B+A)$ などの値で検索を行っても，ハッシュ関数の性質によりそれらは $Hash(A+B)$ と全く関係がない値であり，コンテンツを発見することはできない．このように，コンテンツに複数のキーワードを設定していても，キーワードの組み合わせに柔軟に対応することは難しい．

そこで我々は，データベースなどで使われている Bloom フィルタを用い，DHT 型 P2P において複数キーワード検索を実現する手法を提案する．対象とする DHT 型 P2P のプロトコルとして，構造が比較的単純で扱いやすいことから，Chord を選択した．

2. 基礎となる技術

2.1 Chord

Chord [1] は DHT 型 P2P の代表的なプロトコルの 1 つである．比較的単純な構造ではあるが，耐障害性が高く，少ないホップ数で検索可能である（ノード数 N のとき検索ホップ数は $O(\log N)$）．Chord は，ノード ID に従ってノードを円状に配置した構造を持っており，ID のサイズは 2^m（通常 $m = 160$）である．

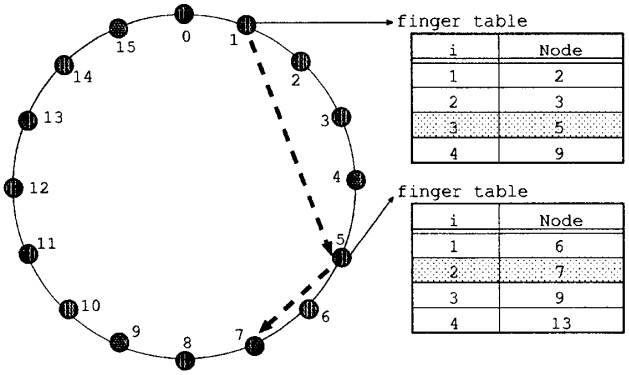

図 1: Chord における ID=7 の検索例)

Chord では効率的な検索を実現するため，finger table と呼ばれる表を保持する．これには，自ノード ID から $2^{i-1}(1 \leq i \leq m)$ だけノード ID の離れた m 個のノードの所在情報が登録される．検索メッセージを受け取ったノードは，各々が保持する finger table の中で目的のコンテンツ ID に近いノード ID を持つノードに検索を転送していくことで，目的のコンテンツを管理するノードへ検索メッセージをルーティングする．図 1 は，$m = 4$ の Chord において ID=7 のノードを検索する例である．

2.2 Bloom フィルタ

Bloom フィルタ [2] は，ハッシュ関数を用いて，ある要素が存在することを一定のビット列で表現するデータ構造である．

Bloom フィルタを構成するには，まず n ビットのビット列を用意し，全ビットを "0" に初期化する．次に，0 から $n-1$ の値を返す k 個のハッシュ関数により，要素のハッシュ値を求める．そして，各々のハッシュ値に該当する位置のビットを "1" に変える．Bloom フィルタを用いて要素の有無を判断するには，検索したい要素のハッシュ値に対応する位置の Bloom フィルタのビットを調べる．対応する全てのビットが "1" であれば，要素が存在すると判断できる．ただし，ハッシュ値が衝突することにより，要素が存在しないにもかかわらず，要素が存在すると判断する可能性がある．逆に，要素が存在するにもかかわらず，要素なしと判断することは起こり得ない．

Bloom フィルタには，複数の Bloom フィルタの論理和をとることで，それぞれの Bloom フィルタに含まれる全ての要素を表現する Bloom フィルタが得られるという特徴がある．このとき，要素数に関わらず Bloom フィルタのビット長は変化しない．

本研究では，Bloom フィルタを用いてコンテンツのキーワードを表現する．Bloom フィルタを使うことで，キーワード数に関わらず同じビット幅でキーワードを表

†リコー Ricoh
‡埼玉大学 Saitama University

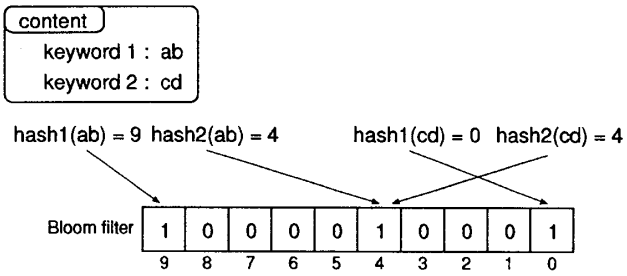

図 2: コンテンツの Bloom フィルタ生成例

現でき，また複数のコンテンツのキーワードをまとめて表現することが可能となる．図 2 は，コンテンツのキーワードを Bloom フィルタによって表現する例である．

3. 関連研究

DHT 型 P2P 上で複数キーワード検索を解決するための手法は既にいくつか提案されている．ここでは，それらの手法を紹介する．

[3] などで用いられている，inverted index を使った手法は，各キーワードのハッシュ値に従い，コンテンツ情報を複数のノードに追加する．検索を行うには，検索キーワードそれぞれに対応するコンテンツのリストを DHT 上で検索し，それらの積をとる．

keyword-set search[4] は，キーワードの組み合わせを求め，それら全てのハッシュ値から複数のコンテンツ ID を生成し，それぞれを DHT 型 P2P 上に追加する．つまり，一つのコンテンツに複数のコンテンツ ID が与えられる．検索時には，指定した複数のキーワードの組み合わせのハッシュ値を求め，DHT 上で検索を行う．

当研究室で過去に考案した手法は，コンテンツのキーワードを Bloom フィルタで表現している [5]．この手法では，コンテンツのキーワードから生成した Bloom フィルタをコンテンツ ID として扱い，DHT 型 P2P 上で管理する．また，キーワードの組み合わせに対応するため，Bloom フィルタの “1” ビットの全ての組み合わせもコンテンツ ID として扱う．これにより，どのようなキーワードの組み合わせにも対応できる．

これら 3 つの手法は，いずれも複数キーワードに関して，複数の ID を得ることで，複数キーワード検索を可能にしている．この方法は，同じキーワードを持つコンテンツは同じ ID を得るため，あるキーワードに関連する全てのコンテンツを探すのに有利である．その反面，キーワード数が増えると，コンテンツの追加・検索の負荷が増大するという問題がある．

Samir らの提案した手法 [6] では，あらかじめキーワードとして登録する属性（提供者・ファイルの種類・サイズ等）を決めておき，それらのハッシュ値を結合して，コンテンツ ID を生成する．検索時には，目的のコンテンツの属性から検索するコンテンツ ID を作成する．このとき，すべてのキーワードを指定する必要はなく，その場合には検索時に検索メッセージが分岐する．この手法では，検索キーワードを増やすと検索の負荷は小さくできるが，自由にキーワードを設定できない．

4. 提案手法

従来の研究では，複数のキーワードから一つもしくは複数のコンテンツ ID を生成し，それぞれの DHT 型 P2P のプロトコルに従って，コンテンツ ID に対応するノードにコンテンツを追加する．コンテンツを検索するには，検索キーワードからコンテンツ ID を生成し，それぞれの DHT 型 P2P のプロトコルに従って検索を行う．一方，本研究ではコンテンツの追加・削除を行う際に，従来の “コンテンツ ID” を使わない．コンテンツの追加に関して制約はなく，ネットワーク中のどのノードに追加しても良い（ただし負荷分散のため，配置が分散されることが望ましい）．また，コンテンツのキーワードは Bloom フィルタを用いて表現し，DHT 型 P2P のネットワーク構造に従って Bloom フィルタの集約を行うことで，Bloom フィルタの比較による検索を可能にした．

“コンテンツ ID” によるコンテンツの追加・検索を行わない利点は，キーワードの人気度の差による負荷集中を回避できる点にある．キーワードに従ってコンテンツ ID を生成する方法では，キーワードとして使う単語の人気度の差により，コンテンツに与えられる ID は一様に分布しない（本来の DHT は，ハッシュ値を利用してコンテンツ ID を決定するため，ID が一様に分布するはずである）．このため，コンテンツの追加・検索の処理に偏りが生まれ，負荷分散が機能しなくなる可能性がある．一方，本手法ではコンテンツをどのノードに追加しても良いので，コンテンツの追加が特定のノードに集中することは容易に回避できる．

Bloom フィルタの比較による検索を行う利点は，複数のキーワードを同時に扱うことができることにある．このため，キーワード数の増加によるコンテンツ追加・検索負荷の増大といった問題が起こらない．

以下，DHT 型 P2P の一つである Chord に従って詳細を説明する．

4.1 キーワードの管理

コンテンツを管理するノードは，管理しているコンテンツの全てのキーワードから，Bloom フィルタを生成する．これにより，ノードが管理しているコンテンツのキーワードを表現する Bloom フィルタができる．これを Node Bloom フィルタ（以下 NBf）と呼ぶことにする．各ノードの NBf を参照することで，キーワード検索を行うことができるようになる．しかし，検索を行う毎に全ノードの NBf を参照するのは効率が悪いため，各ノードの NBf を論理和によって集約することで，DHT 型 P2P の探索経路にどのようなキーワードがあるかを判断できる新たな Bloom フィルタを作成し，検索を効率化する．DHT 型 P2P の構造に従って Bloom フィルタを集約するため，DHT 型 P2P と同様に効率の良い検索が可能となる．

Chord では，finger table 内の各ノードについて，それぞれの探索経路の範囲内の全ノードの NBf の論理和をとる．図 3 は，Node0 の各探索経路の範囲を示しており，Node0 はこれらの探索経路の NBf の論理和から，新たな Bloom フィルタを作成する．こうして得られる Finger Bloom フィルタ（以下 FBf）は，各探索経路の全てのキーワードを含む．FBf を作成するに当たって，

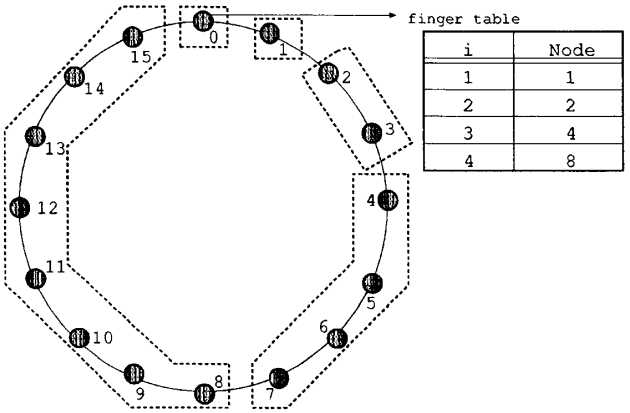

図 3: Node0 の finger の探索範囲

各ノードは finger table 内のノードと通信するだけでよく，以下のような処理を行う．ここで，finger[i] は finger table 内の i 番めのノード，FBf[i] は finger[i] についての FBf を表している．

$$\begin{aligned} \mathrm{FBf}[i] = \mathrm{finger}[i].\mathrm{FBf}[0] &+ \mathrm{finger}[i].\mathrm{FBf}[1] \\ &+ \cdots \\ &+ \mathrm{finger}[i].\mathrm{FBf}[i-1] \end{aligned}$$

4.2 検索方法

検索キーワードから検索 Bloom フィルタを生成し，これに従って検索メッセージをルーティングする．各ノードが DHT の構造に従って集約した Bloom フィルタと検索 Bloom フィルタとを比較し，検索 Bloom フィルタに当てはまる Bloom フィルタに対応するノードにのみ検索を転送することで，基礎となる DHT と同様の探索経路を使った検索が可能となる．ただし，探索範囲が重複しないよう，検索メッセージを転送する際に探索範囲を指定する必要がある．

Chord 上で検索を行うには，FBf と検索 Bloom フィルタを比較し，FBf の中に検索 Bloom フィルタの全要素（“1” ビット）が含まれていれば，対応する finger に検索を転送する，この動作を繰り返すことにより，Chord の検索と同様，各ノードの finger table に検索を転送しながら，検索 Bloom フィルタに一致する Node Bloom フィルタを持ったノードに検索を転送できる．図 4 は，Node0 から，検索 Bloom フィルタ “110000001” に合致する NBf を持った Node5 に検索がルーティングされる様子を示している．ここで，Node4 の検索動作においては，探索範囲の重複を避けるために “finger=12” の FBf は比較しない．

5. シミュレーション

5.1 キーワード検索

提案手法による複数キーワード検索の動作を確認するため，シミュレーションによる検証を行った．ノード数 10000，コンテンツ数 10000，1 コンテンツに与えるキーワード数を 10 とし，キーワード検索を行った結果を図

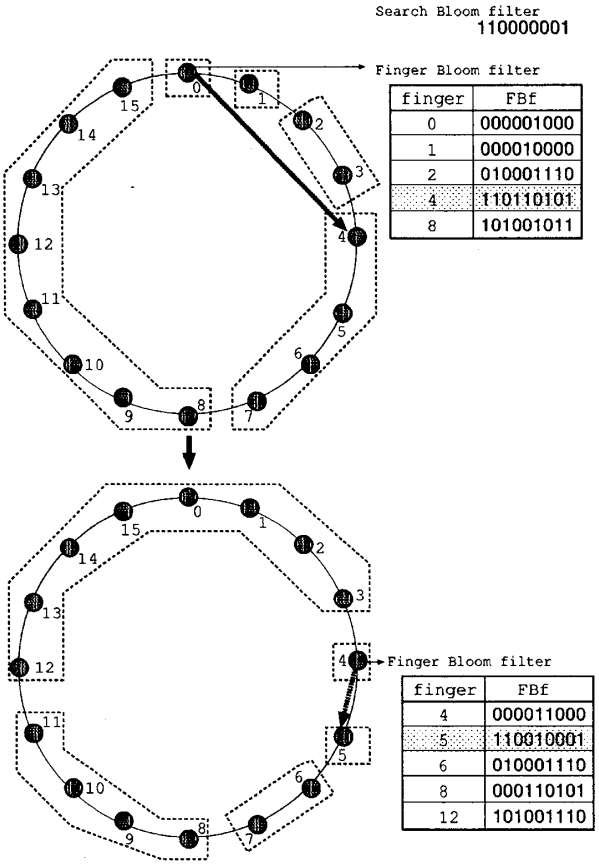

図 4: Bloom フィルタを使った検索例

6 から図 8 示す．キーワードを表現するために使用する Bloom フィルタのビット長は 1000 であり，ハッシュ関数は 3 個用意している．また，コンテンツに与えられるキーワードには図 5 に示したような人気度が設定されている．検索キーワードは，ランダムに選んだコンテンツからキーワードを指定した数だけ取り出して使用した．

図 6，7 に示すように，キーワードを増やすことで，検索コンテンツ，検索範囲を絞って検索できることが確認できた．また，図 8 で示すように，キーワード数による検索パス長の変化は見られなかった．なお，検索パス長とは，目的のコンテンツを管理するノードに検索が辿り着くまでに，検索メッセージがいくつのノードを転送されたか，というホップ数を表す．

5.2 ネットワーク規模の変化による検索パス長の変化

図 9 は，上の実験と同様の条件で，ノード数を変えながら検索パス長を測定したものである．グラフの横軸を対数としている．この結果から，検索パス長の変化はノード数に対して対数的に変化していることがわかる．これは，基礎となる Chord の検索パス長 ($O(\log N)$) と同様の検索パス長の変化である．図 8，図 9 の結果から，提案手法がキーワード数に関係なくスケーラブルな検索が可能であることがわかる．

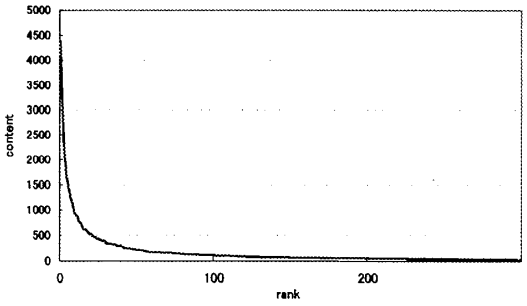

図 5: キーワードの人気度（上位 300 個）

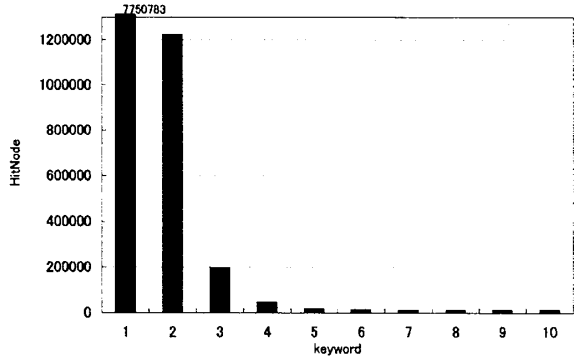

図 6: 検索ヒットノード数

6. まとめ

DHT 型 P2P において複数キーワード検索を実現するため，キーワードを Bloom フィルタで表現し，Bloom フィルタによって検索メッセージをルーティングする手法を提案した．Chord においてこれを実現するには，finger table に従って Bloom フィルタを集約した FBf を用意し，検索 Bloom フィルタと FBf とを比較しながら検索メッセージをルーティングする．

シミュレーションを行った結果から，提案手法によって複数キーワード検索が実現でき，そのホップ数がキーワード数に関係なくノード数の対数に比例することがわかった．ノード数が増加しても，検索パス長の増加は抑えられるため，大規模なネットワークへの適用が見込まれる．

今後の課題としては，検索メッセージ数を減少させ検索をさらに効率化すること，Bloom フィルタのパラメータ（ビット長，ハッシュ関数の数）の決定方法の考案，他の DHT 型 P2P への提案手法の適用などが挙げられる．

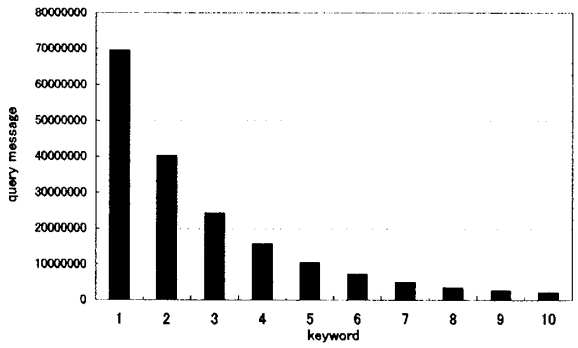

図 7: 検索メッセージ数

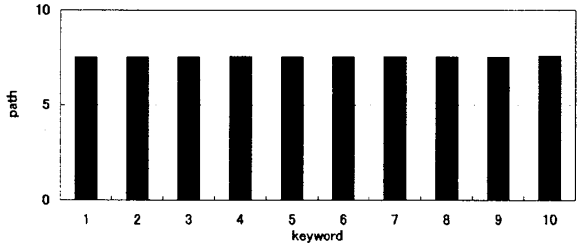

図 8: 検索パス長の変化

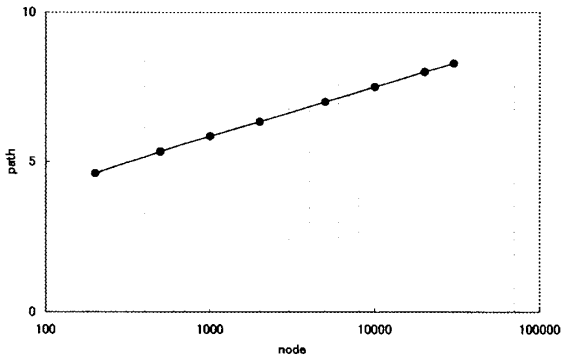

図 9: 検索パス長の変化

参考文献

[1] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek and H. Balakrishnan. "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications", Proc. ACM SIGCOMM 2001

[2] B. Bloom, "Space/Time Trade-Offs in Hash Coding with Allowable Errors" Communications of ACM, 13:7, 1970

[3] P. Reynolds, A. Vahdat, "Efficient Peer-to-Peer Keyword Searching", Proc. Middleware 2003

[4] O. D. Gnawali "A Keyword-Set Search System for Peer-to-Peer Networks" Master's Thesis, MIT, 2002

[5] 佐藤 崇，分散ハッシュ型 P2P ネットワークへの Bloom フィルタの適用，埼玉大学修士論文，2006

[6] S. Ghamri-Doudane, N. Agoulmine "Enhancing the P2P Protocols to Support Advanced Multi-keyword Queries" Proc. Networking 2006

[7] 佐藤 一帆，複数キーワード検索に対応した分散ハッシュ型 P2P ネットワーク，埼玉大学修士論文，2007