2008－DPS－137 (9)
2008／11／28

B木構造に基づくBloomフィルタの提案

若林 繁寿 佐藤 文明
東邦大学理学部情報科学科
5505109w@nc.toho-u.ac.jp, fsato@is.sci.toho-u.ac.jp

Bloomフィルタは、分散システムにおける情報検索方法の一つとして注目されている。Bloomフィルタは、分散ハッシュテーブル（DHT）とくらべて検索に複数のキーワードが使えるなど自由度が高い。特に構造を持ったBloomフィルタは、検索要求の転送回数が確定している点に特徴がある。従来の研究では、リング構造を持ちDHTの一つであるChordと同様の検索要求の転送方法を使ったBloomフィルタが提案されているが、リングのサイズに応じたフィルタ情報を保持する必要があるといった無駄があった。本研究では、管理すべきフィルタ情報を削減するために、木構造に基づくBloomフィルタを提案し、従来研究との情報量の比較を行った。

Bloom Filters Based on the B-Tree

Shigetoshi Wakabayashi Fumiaki Sato
Dept. of Information Science, Faculty of Science, Toho University

The Bloom filter has become attractive as one of the methods of information retrieval in the distributed system. The degree of freedom of the Bloom filter is high as two or more key words can be used for the retrieval compared with the distributed hash table (DHT). Especially, the Bloom filter with the structure has the feature that the forwarding times of query is fixed. In the research of the past, the ring of the Bloom filter which is like the Chord structure is proposed. However, the node in the ring must maintain many filters corresponding to the size of the ring. In this research, we proposed the Bloom filter based on the tree structure to reduce filter information which had to be managed, and size of information with this research was compared to the existing research.

1. はじめに

P2Pにおける情報検索では、分散ハッシュテーブルを使った方式が盛んに研究されている[1,2,3,4]。分散ハッシュテーブルを用いた方式は、問い合わせをフラッディングするピュアP2Pに比べてネットワークの負荷が軽く、ハイブリッドP2Pにくらべてサーバの負荷が広く分散できる特徴がある。しかし、複数のキーワードでの検索がしにくく、範囲検索が難しい、メンテナンスのコストがかかる、といった問題点がある。これに対して、Bloomフィルタを利用して、情報を検索する方法が従来から研究されている[5]。

Bloomフィルタ[6]とは、キーワードに対して複数のハッシュ関数を掛けて得られた値をビット列の位置とみなして、その位置のビットを１にすることで得られたビット列である。Bloomフィルタは、そのサイトに蓄積されたコンテンツのすべてのフィルタをOR演算することで、サイト全体のフィルタを集約することができ、その集約されたフィルタと、検索用フィルタのAND演算を行うことで、そのサイトに情報が含まれている可能性があるかないかを決定できる。

Bloomフィルタを使って、検索要求を転送することで効率的に分散システム上の資源を検索できる。ネットワーク結合されたサイトのBloomフィルタによって問い合わせの転送方向を変える方式に、減衰Bloomフィルタがある[7]。減衰Bloomフィルタは、Bloomフィルタのテーブルを管理し、論理リンクで接続された隣接ノード同士がBloomフィルタのテーブルを交換することで、目的とするフィルタが自ノードから何ホップ先にあるかを知ることができ、適切なリンクに検索要求を転送できる。しかし、この方法は、検索要求を転送する回数の上限を確定することができない。

Bloomフィルタにおける問い合わせの転送方式に、分散ハッシュテーブルの一方式であるChord[4]を使った方式が提案されている[8]。この方式では、検索ホップ数がChordと同じ、O（$\log_2 N$）となっている。しかし、この方式ではハッシュのビット数をMとすると、各ノード

がM個のBloomフィルタを管理する必要がある。

本研究では、m分木のB木に基づく新しいBloomフィルタの検索方法を提案する。この方式では、検索要求の転送回数が$O(\log_m N)$になるとともに、管理するBloomフィルタの量が各ノードあたり$O(m\log_m N)$となる。

2. 関連研究

2.1 Bloom フィルタとは

Bloomフィルタは、ある要素がサイトに存在することをハッシュ関数と一定のビット列を用いて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1 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。

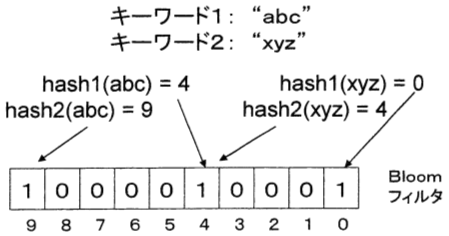

図1 Bloomフィルタの構成例

2.2 Chordの構造を持つBloomフィルタ

通常の分散ハッシュテーブル（DHT）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。

以下、DHT 型P2P の一つであるChord に基づく従来研究[8]の詳細を説明する。

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2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、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 番目のノード、FBf[i] はfinger[i] についてのFBf を表している．

$$FBf[i] = finger[i].FBf[0] + finger[i].FBf[1] + \cdots + finger[i].FBf[i-1]$$

検索では、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。

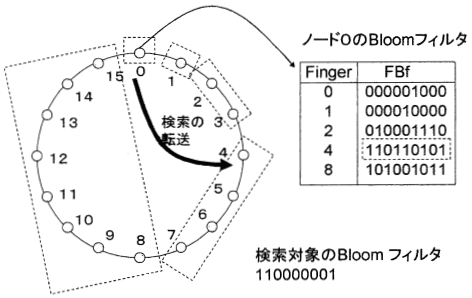

図２ Chord 構成の Bloom フィルタによる検索例

2.3 減衰Bloomフィルタ

減衰Bloomフィルタは、標準的なBloomフィルタをｄ段に組み合わせたものである。あるノードは他のすべてのノードに対する到達性に関する情報を減衰Bloomフィルタに格納する。このフィルタは、ｉ番目の階層では、ｉ＋１ホップで到達できるすべてのノードの情報がまとめられている。最初の階層は、自サイトの情報が格納されている。

表１　減衰Bloomフィルタの例

ビットの番号	0	1	2	3	4	5	6	7
Local Info.	0	1	1	0	0	0	0	0
First Hop	0	0	1	0	1	1	1	0
Second Hop	0	1	0	1	1	1	0	1

この減衰Bloomフィルタは隣接するノード同士で定期的に交換される。あるノードが近隣のノードからBloomフィルタを受信したとき、受信したフィルタのｉ番目の階層はビット同士のOR演算を行って、そのノードのｉ＋１番目の階層のフィルタとして計算される。図３は、その状況を表している。従って、第２階層では１ホップ先の、第３階層では２ホップ先の情報が集約されることになる。

この減衰Bloomフィルタを使った検索では、どのフィルタがどのリンクから受信したかが管理されるため、検索するフィルタがどの方向に存在するか、どの方向に最も近いフィルタがあるかが特定できる。その情報から、適切な検索要求の転送ができる。

この方法は、ネットワークの構造化を前提としていないため、構造化された方法と比較して維持コストは低いが、検索要求が何回転送されるかについて上限が定まらないという問題がある。

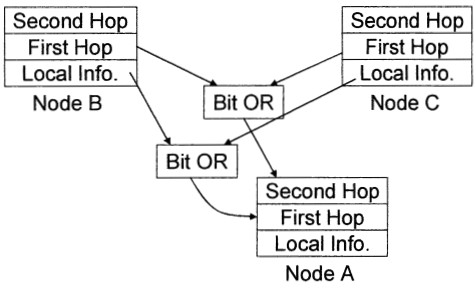

図３　減衰Bloomフィルタの情報交換

3. 木構造に基づく Bloom フィルタの提案

3.1 B木構造

この研究では、各ノード（P2Pインデックス情報を管理している実際のノード）は、分散型のB木によって管理されている。B木におけるノードは、物理的なノードとは異なるため、論理ノードと呼ぶ。B木における葉ノードは、物理ノードのIDが格納されているものとする。また、木の中間ノードには、木の接続関係（親ノード、子ノードへの分岐の状況を示すノードIDの配列など）や情報のバージョン番号が管理されている。

このB木の情報は、参加する物理ノードによって分散管理される。各物理ノードは、B木の部分情報を持っている。B木に変更が生じた場合、他のノードに変更情報を通知することで、全物理ノード間の一貫性を管理している。各物理ノードが持つ部分木の情報は、対応する葉ノードから、根ノードまで、木を遡った経路に存在する各論理ノードが持つ情報と、それらに隣接する兄弟ノードが持つ情報である。従って、ｍ分木を想定すると、物理ノード数をNとしたときに、およそ$\log_m N$の高さのB木ができる。各論理ノードには、ｍ個の分岐を管理するためのサイズｍの配列がある。そして、その各論理ノードに隣接する兄弟ノードの情報を持つと仮

定すると、各物理ノードは$m\log_m N$に比例する個数の情報を持つ。

なお、根ノード、及び各中間ノードは、下に連結された中間ノードの中の一番小さいIDを分岐の区切りとして、配列に保存して管理することとする。この配列中の一番小さいIDを持つ物理ノードを、代表ノードと呼び実際にその配列を管理する責任を持つものとする。

図４にB木によって管理された物理ノードと、その中の代表ノードの例を示す。

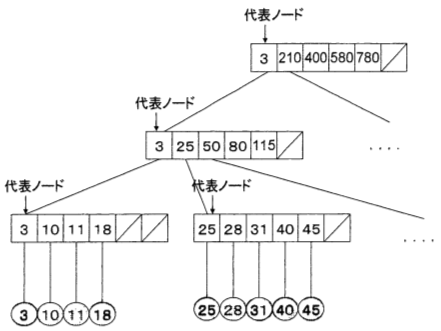

図４　B木によるノードの管理

3.2 木の管理方法

（１）参加方法

物理ノードが新たにＢ木に参加する場合、既に参加しているノードのどれかにアクセスして自身の ID を含む参加要求を送付するものとする。アクセスされたノードは、根ノードの代表ノードに要求を転送する。要求された根ノードの代表ノードは、要求の ID に基づいて、どの中間ノードに管理されるべきかを判断し、要求を下位の中間ノードに転送する。これを繰り返すことで、管理されるべき最も下位の中間ノードの代表ノードに参加要求が到達する。もし、その中間ノードの配列に空きがある場合、その中間ノードの配列に新規に ID が登録されることで、参加処理は終了となる。

もし、空きがない場合、中間ノードはＢ木におけるノードの分割作業を行う。分割作業は、そのレベルの中間ノードの分割が、上位のレベルの分割を引き起こすことがあり、最後には根ノードの分割と、その上位に新たな根ノードの作成が行われることがある。そのプロセスについては、中間ノードの代表ノードが持つ情報（根ノードまでの中間ノードの情報と、その兄弟ノードが持つ情報）のみで判定できる。

変更情報の通知は、変更が生じた最も上位のノードの代表ノードに依頼して実施してもらう。通知は、上位の代表ノードから、下位の代表ノードへとマルチキャストで行われる。なお、その情報更新において、ノードのバージョン番号が既に変更されている場合、更新が別の過程で並行して発生していることを意味しており、変更プロセスを中止する。自身が持つノードのバージョン番号が古いことが分かった場合、その代表ノードに対して、新しいノードの情報を転送するように依頼し、参加処理を再度実行する。

（２）脱退方法

物理ノードがＢ木から脱退する場合、そのノードが属する最下位の中間ノードの代表ノードに脱退要求を送付する。代表ノードは、中間ノードの配列から対象ノードが脱退した場合に、要素数がｍ／２未満にならないならば、ただ配列から対象ノードを削除するだけで脱退処理を終了する。

もし、要素数がｍ／２未満になる場合、隣接の兄弟ノードから配列の要素を移動する。もし、隣接ノードから配列の要素を移動すると、ｍ／２未満になる場合は、その隣接ノードと中間ノードを合併する。この合併操作は、必要に応じて根ノードまで遡り、場合によっては最上位の根ノードを削除する操作までを行う。このプロセスも、中間ノードの代表ノードが持つ情報によって、判定できる。

変更情報の通知は、ノードの参加処理での通知方法と同様に、変更が生じた最も上位のノードの代表ノードに依頼して実施してもらう。

3.3 Bloomフィルタの管理

各物理ノードでは、インデックス情報が管理されており、各インデックス情報はBloomフィルタを持っている。各ノードは、ノード内のBloomフィルタをすべて集約した集約Bloomフィルタを持っている。また、B木の構造に基づいて、上位の中間ノードは、下位のノードのBloomフィルタを集約した集約Bloomフィルタを持っている。最上位の根ノードには、全ノードのBloomフィルタを集約した全体のBloomフィルタがある。

B木の情報と同様に、Bloomフィルタの情報も各物理ノードに分散されている。つまり、各物理ノードは、B木の葉から根に至る経路上にある中間ノードに存在する集約Bloomフィルタと、その兄弟ノードの集約Bloomフィルタを持っている。　各ノードが持つべき情報を図５に示す。図５において、TRは根ノードのB木の情報、Ｔ１からＴｋは第１レベルのB木の情報、Ｔ１１からＴ１２はＴ１以下の第２レベルのB木の情報を表す。また、FRは根ノードのフィルタの情報、Ｆ１からＦｋは第１レベルの集約されたフィルタの情報、Ｆ１１からＦ１２はT1以下

の第２レベルの集約されたフィルタの情報、FAからFCはノードAからノードCのフィルタの情報である。

（１）葉ノードの集約Bloomフィルタの変更

葉ノードの集約Bloomフィルタに変更があった場合、集約Bloomフィルタの変更が伝搬する最上位の中間ノードの代表ノードに通知する。変更は、代表から関係する兄弟ノード、下位ノードの代表ノードにマルチキャストにより通知される。さらに、フィルタを更新するとき、あらかじめ更新する順番を決めておく。

（２）ノードの参加

B木への新規の参加があったとき、そのノードを収容する中間ノードの代表ノードは、集約Bloomフィルタの変更を計算する。もし、集約Bloomフィルタの変更がある場合、その変更が伝搬する最上位の中間ノードの代表ノードに通知し、そこからマルチキャストによって各ノードに伝搬する。

（３）ノードの脱退

B木からの脱退があったとき、そのノードを管理する中間ノードの代表ノードは、集約Bloomフィルタの変更を計算する。もし、集約Bloomフィルタの変更がある場合、その変更が伝搬する最上位の中間ノードの代表ノードに通知し、そこからマルチキャストによって各ノードに伝搬する。

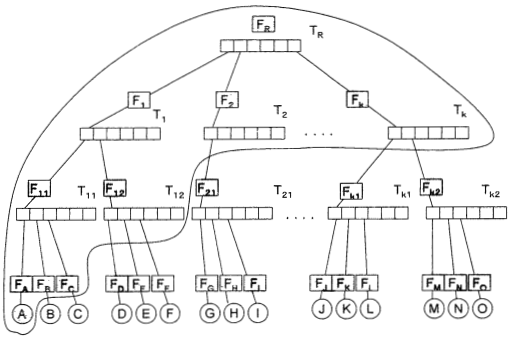

図５ ノードが持つべき情報の範囲(ノードAの場合)

3.4 検索方法

情報の検索では、まず検索用のキーワードから検索用の Bloom フィルタを作成する。検索用 Bloom フィルタを含む検索要求を、B 木のどこかのノードに送付する。検索要求が到達したノードでは、そのノードが保持する B 木の部分情報について、根ノードから比較を開始する。B 木の各ノードに付随する集約 Bloom フィルタと検索用の Bloom フィルタとの AND 演算を実施して、検索用 Bloom フィルタが残る（情報が見つかった）最も下位（つまり、部分木中の葉）のノードを見つける。そのノードが中間ノードであれば、その下の各中間ノードの代表ノードに問い合わせを送る。また、そのノードが葉ノードであれば、対応する物理ノードに直接検索要求を送付する。

転送された検索要求を受け取った中間ノードの代表ノードは、その中間ノード以下のノードに対して検索要求に含まれる Bloom フィルタを使っての照合を実施する。そして、最も下位のノードに到達するまで、繰り返し検索要求の転送を行う。

図６に検索要求の転送例を示す。ノード A に検索用のフィルタ０１０１０１が送られて来た場合、ノード A は自身が持つフィルタの情報から、Fk と F12 に検索が合致することが分かる。そのことから、その集約されたフィルタに関連するノード群を下位に持つ部分木の代表ノードであるノード D とノード J に検索要求が転送されることになる。

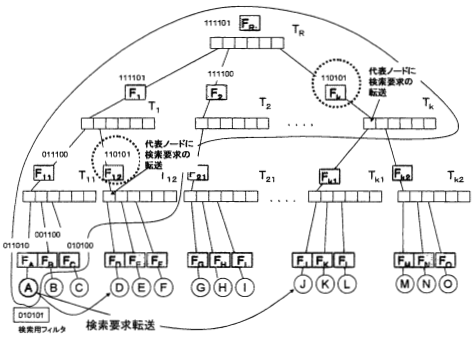

図６　ノードAへの検索要求の転送例

4. 評価

検索にかかるメッセージの転送回数について評価する。検索要求は、根ノードから順次集約 Bloom フィルタとの照合が行われる。つまり、この照合作業は最大で木の深さの回数だけ行われ、葉の位置にある物理ノードが検索される。従って、メッセージの転送回数は最大で、O（$\log_m N$）回となる。

また、Bloom フィルタの変更や追加、削除に要する作業の計算量を評価する。Bloom フィルタの変更については、個々のノードが計算する伝搬範囲は最大で $O(\log_m N)$であるが、情報の伝搬には最大 N に比例するメッセージが伝搬する。

管理する Bloom フィルタの情報量としては、木の高さ分の中間ノードがあることと、１つの中間ノード当たりｍ個の兄弟ノード分のフィルタを持つことから、O（$m\log_m N$）必要となる。例えば、N=10000、m=10 としたとき、

各ノードは 40 程度のフィルタを保持する必要がある。また、各ノードにおける Bloom フィルタの検索処理は、そのノードで管理されているフィルタ数に比例するコストとなるため、O（$m\log_m N$）となる。

一方で、[8]における検索要求の転送回数は、Chord における検索要求転送回数と同等であるので、O（$\log_2 N$）回となる。また、フィルタの変更は、変更があったノードが関連するノードに通知する場合、O（$\log_2 N$）回の通知が必要であり、最大ですべてのノードへの伝搬が発生しうるため、N に比例する通信コストが必要である。管理する Bloom フィルタの情報量は、ハッシュのビット数を M とすると、各ノードが M 個の Bloom フィルタを管理する必要がある（$M \geqq \log_2 N$）。Chord のハッシュ関数に広く使われている SHA1 を用いた場合、M=160 となり、各ノードは 160 程度のフィルタを管理することになる。各ノードでの検索処理にかかるコストは、やはり管理されているブルームフィルタ数に比例することから、ノード数 N の大小によらず O(M)となる。

表２　比較評価

	Chordベースの方式[8]	提案するB木に基づく方式
検索要求転送回数	O（$\log_2 N$）	O（$\log_m N$）
フィルタ変更の最大伝搬コスト	O(N)	O(N)
ノードが管理するフィルタ数	O(M)	O（$m\log_m N$）
ノードが管理する他ノードＩＤ	O(M)	O($m\log_m N$)
ノードでの検索コスト	O(M)	O（$m\log_m N$）

5. まとめ

本研究では、m 分木の B 木構造を持つ Bloom フィルタによって情報検索するための Bloom フィルタの管理方法を提案した。この方法では、ネットワークで結合されたサイトを、論理的な B 木構成に維持するための手間がかかるが、検索要求の転送回数の上限が確定するため、検索時間がむやみにかかることがない利点がある。

固定的なリング構造を持った Bloom フィルタの従来研究と比較して、我々の提案では木の高さが動的に変化する B 木構造を持つため、各ノードが持つ必要があるフィルタ情報の量が削減できる。これは、各ノードで行われる検索処理の速度の向上にも寄与すると思われる。

今後の課題は、B 木の中間ノードの管理に、代表ノードという集中型の管理方式を用いており、木の維持情報の伝搬や検索要求の伝搬に使用している。この代表ノードに対する負荷の集中がどれだけ性能に影響するかを評価する必要がある。また、このような代表ノードを固定しない方式などを今後検討していく。また、実際に試作システムを開発して、性能を評価していく必要がある。

参考文献

[1] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A Scalable Content-Addressable Network. In Proceedings of the ACM SIGCOMM 2001 Technical Conference, San Diego, CA, USA, August 2001.

[2] Antony Rowstron and Peter Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), pages 329-350, November 2001.

[3] Petar Maymounkov and David Mazieres. Kademlia: A Peer-to-peer Information Systems Based on the XOR Metric. In Proceedings of the IPTPS 2002, Boston, March 2002.

[4] Ion Stoica, Robert Morris, David Karger, Frans Kaashoek, and Hari Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In Proceedings of the ACM SIGCOMM 2001, San Diego, CA, USA, August 2001.

[5] Andrei Broder, Michael Mitzenmacher, Network Applications of Bloom Filters: A Survey, Internet Mathematics, 2002, pp.636-646

[6] B. Bloom. “Space/Time Tradeoffs in Hash Coding with Allowable Errors.” Communications of the ACM 13:7 (1970), 422—426.

[7] S. C. Rhea and J. Kubiatowicz. “Probabilistic Location and Routing.” In Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Volume 3, pp. 1248—1257. Los Alamitos, CA: IEEE Computer Society, 2002.

[8] 佐藤一帆, 松本倫子, 吉田紀彦, ” 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”, FIT2007.