B 木構造に基づく Bloom フィルタ 検索システムの実装と評価

堺竜太郎† 唐笠良太† 佐藤文明†

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 フィルタの特徴は，分散ハッシュテーブル(DHT)と比べて，検索に複数のキーワードが使えるといった柔軟性にある．また，構造化された Bloom フィルタでは，検索要求の転送回数が確定する点で応答速度が推定できるなどの利点がある．従来の構造化 Bloom フィルタでは，Chord と同様にリング型の方式が提案されてきたが，本研究では検索ホップ数が小さくなる木構造に基づく Bloom フィルタを採用する．特に，Bloom フィルタの木を構成する時に，特定の起点ノードを設けると検索要求が集中する問題が生じるため，特定のノードを起点としないように検索できるアルゴリズムを導入した．これらの提案内容を，シミュレーションと実装システムによって評価した．

Implementation and evaluation of Bloom filter search engine based on B tree

RYUTARO SAKAI† RYOTA TOGASA†
FUMIAKI SATO†

The Bloom filter has become attractive as one of the methods of looking up data in the distributed system. Since two or more keywords can be used in the method based on the Bloom filter, it is more flexible than the distributed hash table(DHT). Especially, the key aspect of the structured Bloom filter is that the number of query forwarding is fixed. In the previous research, the ring of the Bloom filter, which is like Chord ring, was proposed. In this paper, we propose the tree structured Bloom filter because the number of hops of query forwarding is smaller than ring based approach. In this paper, the delay based tree construction method are proposed. We also propose a lookup algorithm to which a specific starting point node is not required. To evaluate our method, response time for lookup and number of forwarding times of the query message are evaluated by simulation and implementation.

†東邦大学理学研究科
Toho University, Graduate School of Science

1.はじめに

現在ネットワーク上に検索をかけて，動画や音楽などの欲しい情報を得るといったことが日常的に行われている．しかし，検索サイトのように検索が集中するシステムには大規模なサーバを用意することが必要であり，メンテナンスのコストもかかる．それに対して，P2P を利用した検索システムではサーバは必要なく，Peer 全体が処理を負担する．

P2P における情報検索では，分散ハッシュテーブル(Distributed Hash Table, DHT)を用いた方式が多く提案されている[1，2，3，4]．分散ハッシュテーブルを用いたシステムは，問い合わせをフラッティングする Pure P2P よりもネットワーク負荷が少なく，Hybrid P2P と比べてサーバ負荷を分散できる特徴がある．

しかし，分散ハッシュテーブルを用いた情報検索では，複数のキーワードでの検索がしにくく，検索範囲が難しいといった問題点があった．それに対して，Bloom フィルタを用いた情報検索が提案されている[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る．その集約されたフィルタを，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．これにより，検索要求の送信先を効率的に選択することができる．

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．それによって，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，検索要求の転送方式に B 木構造に基づく検索方式が提案されている[8]．この方式では，ノードは自信の持つ情報から Bloom フィルタを作成し保持する．ノードは他の情報を持つノードと B 木構造を構築し，B 木構造の上位にあたるノードは下位のノードの Bloom フィルタを集約した Bloom フィルタを保持する．これにより検索要求が到着したとき，集約された Bloom フィルタを参照することで，検索要求の転送先を選択することができる．

しかし，従来の B 木を用いた Bloom フィルタの利用では集約された Bloom フィルタを持つノードに検索要求が集中してしまい，特定のノードの負担が大きかった．また，端末間の遅延時間を考慮してかったために，遅延時間を考慮した B 木構造を構築

することでさらに効率的な検索が実現できると考えられる．そこで本研究では，遅延時間を考慮した B 木構造の提案と，負荷の集中を減らす提案および実装を行い，評価を行った．

2.関連研究

2.1 Bloom フィルタ

Bloom フィルタは 1970 年に Burton H. Bloom が考案したデータ構造である．Bloom フィルタは，ある要素が存在することを一定のビット列で表現する．

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．対応するすべてのビットが“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，要素が存在するのにもかかわらず，要素なしと判断することは起こりえない．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すべ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，要素数にかかわらず 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，キーワード数にかか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 2.1 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．

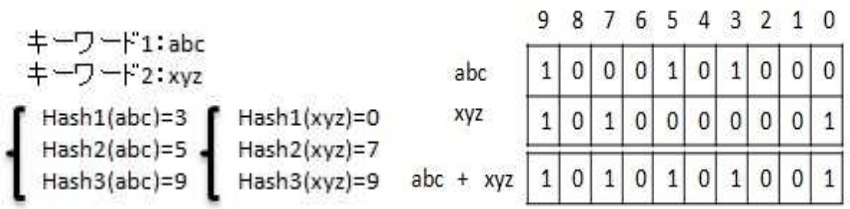


図 2.1　Bloom フィルタの構成例

2.2 木構造に基づく Bloom フィルタ

2.2.1 B 木構造

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は，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．したがって，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，根ノードおよび各中間ノードは，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 2.2 は 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である．

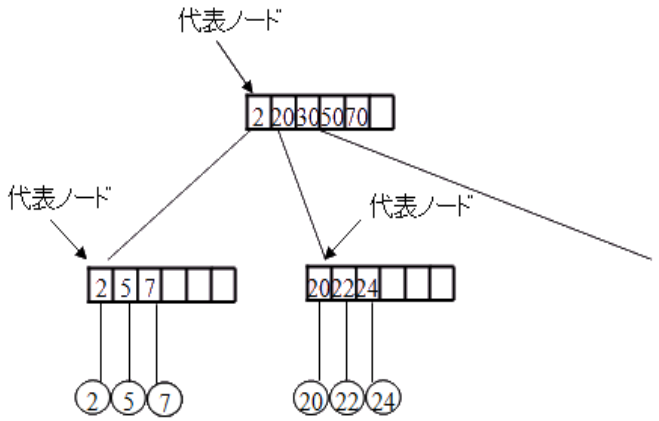


図 2.2　B 木によるノード管理

2.2.2　ノードの参加脱退方法

新規の物理ノードの参加方法は，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アクセスする．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，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．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．

既に参加しているノードの脱退方法は，脱退する物理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．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．もし要素数が m/2 未満になる場合は，隣接する兄弟ノードから配列の要素を移動する．隣接ノードから配列の要素を移動するとき 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．

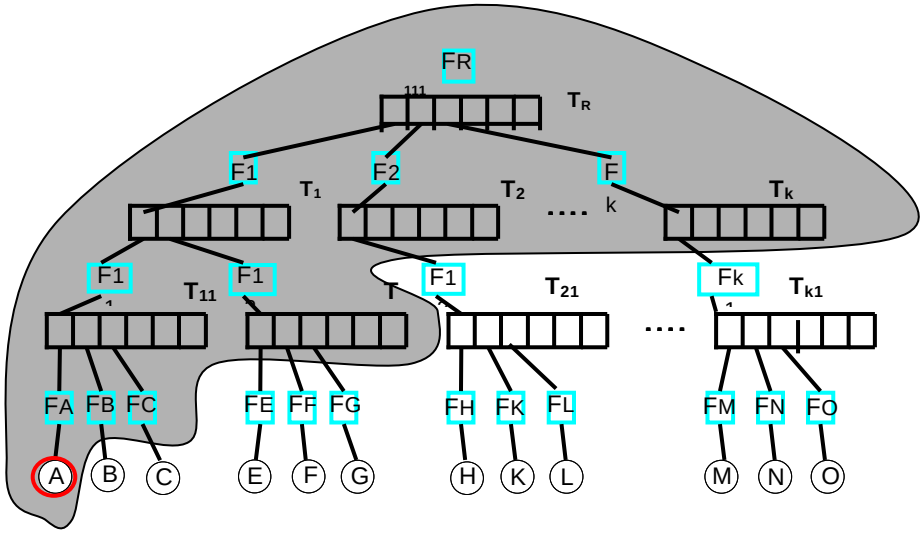


図 2.3　ノードAの持つべき情報の範囲

2.2.3　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，すべての Bloom フィルタを集約した全体の集約 Bloom フィルタを持っている．

B 木の情報と同様に，Bloom フィルタの情報も各ノードに分散されている．

2.2.4　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，繰り返し検索要求を行う．

3.提案手法

提案手法では，端末間の遅延時間を考慮した検索アルゴリズムの提案と，検索要求の集中を防ぐアルゴリズムを実装した検索システムの提案を行う．

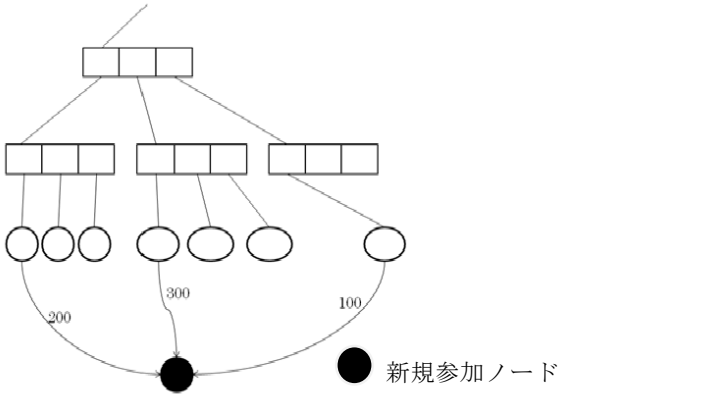
3.1　端末間の遅延時間を考慮した検索アルゴリズムの提案

3.1.1　新規ノード参加方法

提案方式では，新規ノードの参加する方法はすでに参加しているノードのどれかに参加要求を送付する．参加要求を受け取ったノードは，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下位の中間ノードの代表ノードと新規のノードとの遅延時間を求める．その後，その遅延時間に基づいて，遅延時間の一番短い中間ノードに参加要求を転送する．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．最も下位の中間ノードに参加要求が到達したら，代表ノードからの遅延時間を求め，参加しているノードと新規のノードとの

遅延時間を比較して，順番になるように新規ノードを追加する．

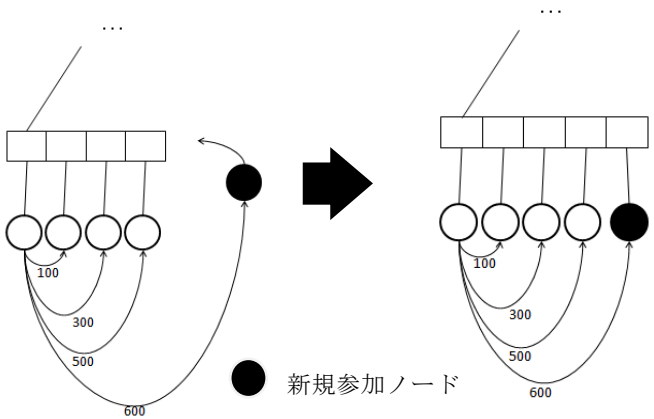
変更情報の通知は，変更が生じた最も上位のノードの代表ノードに依頼して実施してもらう．



数値:代表ノードからのそれぞれのノードの遅延時間

図 3.1　新規ノードと代表ノードとの遅延時間比較例

図 3.1 の場合，新規参加ノードは，一番右の中間ノードに参加要求を転送する．



数値:代表ノードからのそれぞれのノードの遅延時間

図 3.2　新規ノード参加例

図 3.2 のように，最も下位の中間ノードに要求がきたら，最も下位の中間の代表ノードからそれぞれのノードの遅延時間を求め，遅延時間の順番に並ぶように新規ノードを追加する．

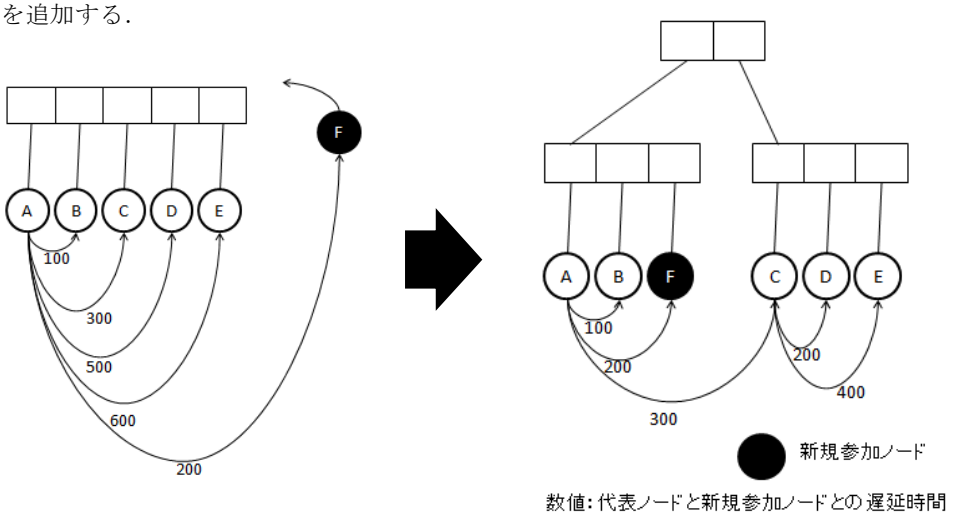


図 3.3　中間ノード分割の例（5 分木の場合）

中間ノードの分割方法は，分割する前に中間ノードが保持する子ノード m 個の中から新しく入る子ノードを代表ノードからの遅延時間が短い順に(m+1)/2-1 個選び，新しく入る子ノードと選ばれた子ノード，それ以外の子ノードで 2 つのグループに分ける．

図 3.3 の場合，代表ノードからの遅延時間はノード B，ノード F，ノード C，ノード D，ノード E という順番になるので，ノード A，ノード B，ノード F のグループとノード C，ノード D，ノード E のグループに分かれる．分かれた時に，ノード C が代表ノードになり，ノード C とノード D，ノード C とノード E の遅延時間を求め，遅延時間の短い順に並び替える．

3.2 検索要求の集中を防ぐ検索アルゴリズムの提案

プログラミング言語は C++で記述する．プログラムは複数のコンピュータ，または同じコンピュータ上でプロセスを複数起動し，それぞれのプロセス同士が互いに通信を行い B 木構造を構成する．また，評価を行う端末も作成する．この端末が B 木を構成するノードに検索要求を送信し，B 木構造における Bloom フィルタの検索の集中について評価する．

3.2.1 ノードの管理する情報

各物理ノードは自分の Key 値，IP アドレス，ポート番号，Bloom フィルタを保持する．また，物理ノードが B 木において根ノードに達するまでの中間ノード情報を保持する．各中間ノードは，中間ノードに属するノードの Bloom フィルタと，それを管理する物理ノードの情報を保持する．図 3.4 はノード K が持つべき情報の範囲を示している．

3.2.3 検索方法

検索をかける端末は検索用 Bloom フィルタを作成し，B 木を構成する端末へ送信する．検索用 Bloom フィルタを受信した端末は，自分の持つ Bloom フィルタとその上位の中間ノードが持つ Bloom フィルタを検索用 Bloom フィルタと比較し，検索用 Bloom フィルタの情報を含んだ端末へ要求を転送する．これを繰り返し，検索用と一致する Bloom フィルタを持つ端末の情報を検索の送信元へ送信する．

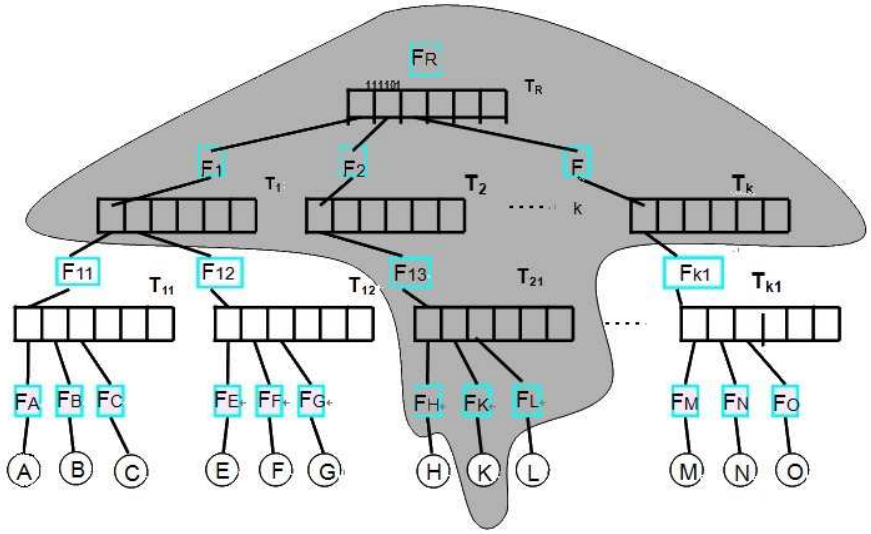


図 3.4 ノード K の持つべき情報の範囲

4.評価・考察

提案手法を用いた B 木構造における Bloom フィルタ検索システムの評価・考察をする．端末間の遅延時間を考慮したアルゴリズムの評価はシミュレーションで行う．

シミュレーションにはダイクストラ法を使用し，ノード間の重みを遅延時間[msec]として比較した．ネットワークデータを作るツール GT-ITM[11]を使用して，TS モデルのネットワークを作り，現実的なネットワークに近いデータを使う．今回のシミュレーションでは，トランジットノードの数を 4，スタブノードの数を 256 に設定した．GT-ITM ではノード間の重みがランダムに割り当てられる．

また，検索の集中を防ぐアルゴリズムでは，Bloom フィルタを用いた B 木構造を構成する端末を実装し評価した．

4.1　遅延時間を考慮した検索システムのシミュレーション

B 木構造のすべての物理ノードに Bloom フィルタを持たせたプログラムを使用し，検索要求時に遅延時間を計測するプログラム作成し，検索要求においての遅延時間を従来方式と提案方式で比較した．

計測方法は，すべてのノードの検索を行い，それぞれに発生する遅延時間を合計して，ノード数で割り，平均の遅延時間を出力する．ノード間の重みをランダムに設定し，数回この処理を行い，その出力された遅延時間の平均をとって比較した．ノード数は 20，50，100，200，500，1000 で固定し，それぞれの遅延時間の平均を計算した．　図 4.1 では 5 分木での従来方式と提案方式の遅延時間を比較した．図 4.2 では 10 分木での従来方式と提案方式の遅延時間を比較した．

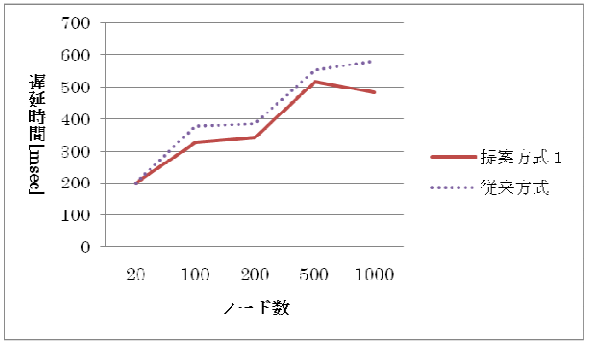


図 4.1　遅延時間の比較(5 分木)

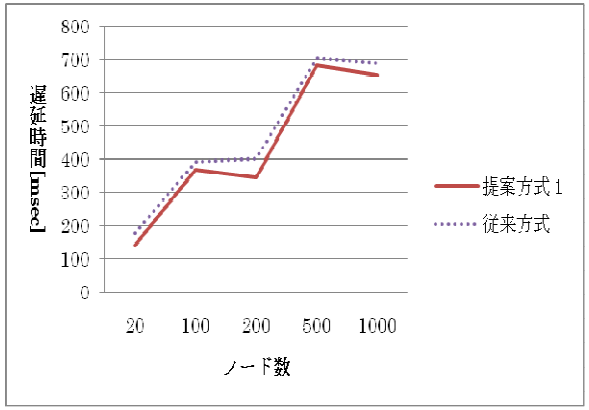


図 4.2　遅延時間の比較(10 分木)

4.1.2　考察

従来方式と提案方式を比較して，提案方式のほうが遅延時間が少ないことがわかる．また，提案方式ではm分木のmに関係なく従来方式より遅延時間が少なくなることが分かる．この計測結果から，提案方式は B 木構造に基づく Bloom フィルタの検索時に発生する遅延時間を削減できたことが分かる．

4.2 検索の集中を防ぐアルゴリズムを実装した検索システム

評価には複数のプロセスを同時起動して実験を行った．

各プロセスに Bloom フィルタをもたせ，参加を繰り返し Bloom フィルタを集約する B 木のデータ構造を作成する．作成された B 木構造を成す端末に無作為に 100 回検索用 Bloom フィルタを送信し，検索方法を従来方式と提案方式において，それぞれの端末が検索要求を受けた回数の合計，また全端末での平均，分散，標準偏差，最小値，最大値を記録した．図 4.3,4.4 では 3 分木，図 4.5,4.6 では 5 分木で計測した結果を示した．

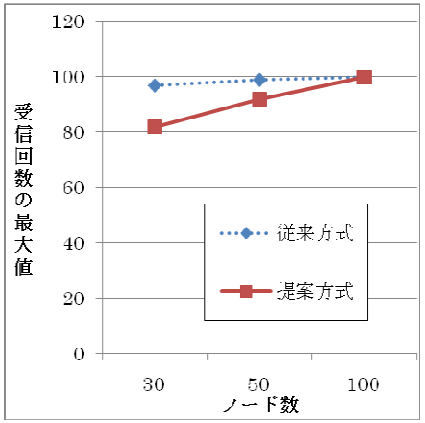


図 4.3　受信回数の最大値(3 分木)

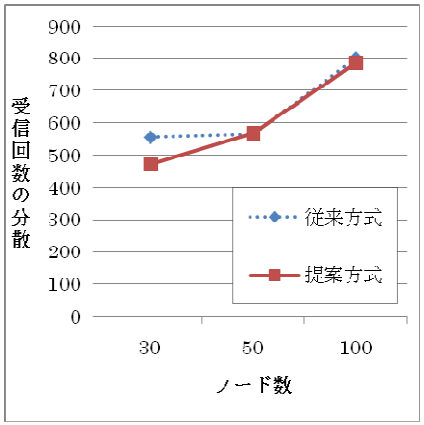


図 4.4　受信回数の分散(3 分木)

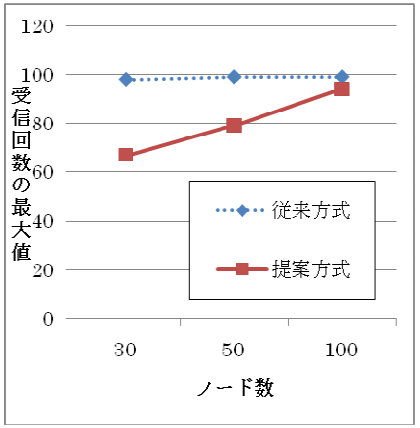


図 4.5　受信回数の最大値(5 分木)

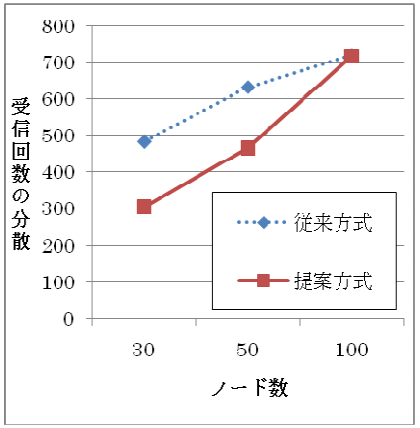


図 4.6　受信回数の分散(5 分木)

4.2.1 考察

従来方式と提案方式を比較して，図 4.3 と 4.5 を見ると，分木数にかかわらず 1 端末が受け取る検索要求の回数の最大値が提案方式によって減少していることが分かる．それに伴い，図 4.4 と 4.6 では 1 端末が受け取る検索要求の回数の分散が減少し，特定のノードに集中していた検索要求を分散させることができたことが分かる．また，ノード数が多いと従来方式と提案方式の結果の差が小さくなるのは，B 木による

Bloom フィルタの集約が進むことで，上位のノードにおいての情報の誤検出が従来方式と同程度起こってしまったためと思われる．

5.まとめ

この論文では，B 木構造に基づく Bloom フィルタにおいて，端末間の通信遅延時間を考慮した B 木構造の提案をし，シミュレーションにより提案方式が従来方式に比べて通信時間が短縮されたことを示した．また，検索要求を受け取った時に特定のノードに要求が集中してしまう問題に対して，各端末が上位の Bloom フィルタを管理し，検索要求を送信する端末を選択できるようにすることで，要求が集中しない方法を提案した．提案方式を実装することで，実際に代表ノードのような検索が集中しやすい端末に届く検索要求が減ることを示した．

今回の実装では，Bloom フィルタを 32 ビットで作成した．しかし，Bloom フィルタの集約が進むことで誤検出確率が増加してしまうため，端末数を増やすと効果が減少してしまった．

今後の課題は，誤検出確率の増加を抑える Bloom フィルタを実装することで，より効果的な結果が得られるように改善することである．

謝辞 本研究の一部は，日本学術振興会科学研究費，基盤研究(C)(22500071)の助成を受けたものである．

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

[6] 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.

[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] 若林繁寿, 佐藤文明, “Bloom Filters Based on the B-Tree” 社団法人 情報処理学会 研究報告 2008-DPS-137 (9) pp43-48