Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

ピアの近接性を考慮したスキップグラフの構築

牧川文紀[†1]　土屋達弘[†1]　菊野亨[†1]

キーの検索に特化したグラフであるスキップグラフは，ネットワーク上に点在するピアが持つキーを管理するために有効なグラフ構造である．しかし，通常のスキップグラフはピアの物理的な位置やピア間の通信時間などを考慮していない．そのため，部分的に通信時間が非常に大きいリンクがグラフ内に存在してしまう可能性がある．本論文ではスキップグラフを構築する際にピア間の通信時間を計測し，より通信時間が小さくなるようにグラフの構築を行う．また，グラフ全体のバランスを取るための手法も合わせて提案する．シミュレーションにより，提案した手法が通常のスキップグラフと比較して小さい検索時間でキーの検索ができることを確認した．

Constructing Skip Graphs with Proximity

Fuminori Makikawa,[†1] Tatsuhiro Tsuchiya[†1]
and Tohru Kikuno[†1]

A skip graph is a valuable overlay network for searching for keys in a peer-to-peer application. A problem with the construction algorithm for skip graphs is that it considers neither peers' physical location nor communication cost between peers. Because of this, a skip graph often contains links with considerably high communication time. In this paper, we propose a communication cost-aware construction to reduce the time required to search for keys. We also propose a method for dynamically keeping the balance of a skip graph. The results of simulations show that in the skip graph constructed by our approach, a peer can search for keys with lower time than in the original skip graph.

†1 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University

1. はじめに

ネットワーク技術の発達とともに，ネットワークに参加するピアは増加し続けている．パソコンに限らず，携帯電話や家電製品等も参加するような大規模ネットワーク型アプリケーションになると，一極集中型のサーバ・クライアント型のシステムではサーバの処理能力を確保することが難しくなってくる．そのため，そのようなアプリケーションではピア同士が相互に接続する P2P 型のシステムを構築する必要がある．

P2P アプリケーションは，オーバーレイネットワークと呼ばれる仮想のネットワークを構築し，そのネットワークを通じてメッセージのやり取りを行う．オーバーレイネットワークにはその構築手法から構造型オーバーレイネットワークと非構造型オーバーレイネットワーク[5),9)] があるが，キーの検索を確実に行うことができる構造型オーバーレイネットワークをここでは考える．

構造型オーバーレイネットワークを構築する手法にはいくつか種類があるが，その中にキーの検索に適した構造を作る手法として，分散ハッシュテーブル（DHT）とスキップグラフがある．DHT はキーをあるハッシュ関数によって変換し，ハッシュテーブルから対応するピアを探して検索を行う．代表的な DHT には CAN[7)]，Chord[8)] などがあり，いずれもピアの総数を n としたとき，キーの検索を $O(\log n)$ のホップ数で行うことができるオーバーレイネットワークを構築する．しかし DHT ではキーをハッシュ変換するため，キーの範囲検索を行うことには向いていない．

スキップグラフ[2)] はスキップリスト[6)] の考え方を分散システムに適用したオーバーレイネットワークであり，キーの検索にかかる時間は DHT と同様 $O(\log n)$ となる．またグラフがキー順に整列したリストから構築されているため，DHT で行えない範囲検索も行うことができる．スキップグラフの形状を改良した手法もいくつも提案されている．[1),3),4)]

しかし，キーの検索にかかる時間はホップ数だけではなくピア間の通信時間の影響も受ける．よって本論文では，ピア間の通信にかかる時間を考慮することによって，スキップグラフの利点であった検索ホップ数を保ったまま，より検索時間を小さくできるようなスキップグラフの構築手法を提案する．シミュレーションにより，通常のスキップグラフと比較してキーの検索をより少ない時間で行えることを確認する．

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

2. 用語説明

2.1 用　　語

本論文では以下の用語を用いて説明する．

- ノード：PC 1台など，物理的にネットワークを構築するエンドノード
- ピア：ひとつのキーとメンバーシップベクトルを保持し，スキップグラフを構築する単位．ひとつのノードに複数のピアがある状況も考えうる
- リンク：ピア同士がスキップグラフ上で別のピアと通信するための接続．TCP/IP 接続を設けている，あるいは接続相手の IP アドレスを保持していることをリンクを持っていると表現する
- リスト：複数のピアが，キー順に整列したもの．スキップグラフにおいては各ピアはリンクによって接続される
- スキップリスト：スキップグラフの元となったデータ構造．レベル 0 では全てのピアが昇順にリストを構築し，各レベルに含まれるピアの一定割合が一つ上のレベルに参加しリストを構築する
- スキップグラフの高さ：スキップグラフにおける最大レベル．それぞれのピアが自分が頂点とするスキップリストを構築するため，全てのピアの最大レベルの最大値をスキップグラフの高さとする
- 通信時間：リンクを持つピア間で必要となる通信時間
- 検索クエリ：キーの検索を行う際に発信されるメッセージ．目的のキーと検索開始ピアのアドレス情報を含み，スキップグラフで順次転送を繰り返される
- 検索時間：あるピアからあるキーを検索するために必要な通信時間の合計．検索クエリを中継するピアでの処理時間は考慮しない
- 検索ホップ数：あるピアがあるキーを検索するときに，検索クエリが転送される回数

なお，シミュレーションでは検索時間および検索ホップ数に関しての評価を行う．

2.2 スキップグラフ

スキップグラフは[2]，スキップリスト[6] を P2P 環境下での検索を行うために拡張したオーバーレイネットワークである．以下では，このスキップグラフを通常のスキップグラフと呼ぶ．それぞれのピアは検索に用いられるキーと，スキップグラフの構成のために用いられる，バイナリで表現されるメンバーシップベクトルを保持している．グラフはレベルと呼ばれる階層に分けられており，レベル 0 を最下層としている．レベル 0 では全てのピアがキー順に双方向リンクによってリストとして接続されており，レベル 1 では各ピアが 2 つのリストのどちらかに含まれている．同様に，レベル i で同じリストに含まれているピアは，レベル $i+1$ では 2 つのリストに分かれる構造になっている．このときレベル i のリストに含まれるピア数が 1 である場合は，そのリストに含まれるピアの最大レベルが i であるとしレベル $i+1$ は定義されない．メンバーシップベクトルの上位 i ビットが一致するピアはレベル i で同じリストに属している．図 1 は通常のスキップグラフの例である．メンバーシップベクトルをあらかじめランダムに決定しておくことで，ピア数が増加すればするほど各レベルの各リストに含まれるピア数が均一となる．このような場合，高い確率でグラフの高さは $O(\log n)$ となる．n はスキップグラフに参加しているピアの総数である．

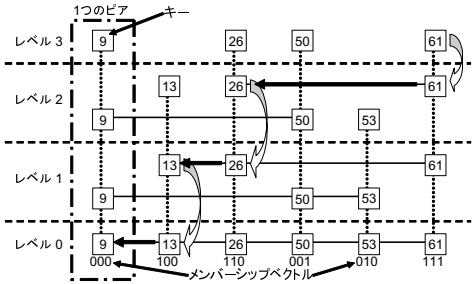

図 1　通常のスキップグラフ

2.2.1 スキップグラフにおける検索

スキップグラフでキーの検索を行う際は，上位レベルから順に検索クエリを転送していく．上位レベルの方が，目的のキーにより近いピアに検索クエリを転送できるからである．例えば図 1 において 61 のキーを持つピアがキー 9 を検索する際，まずレベル 2 においてキー 26 を持つピアに検索クエリを転送する．キー 26 を持つピアはレベル 2 ではキー 9 に近いキーを持つ隣接ピアを持たないため，レベルを一つ下げ，レベル 1 でキー 13 のピアに検索クエリを転送する．キー 13 のピアも同様に，レベルを一つ下げレベル 0 で検索クエリを転送し，検索クエリを受信したキー 9 のピアは検索開始ピアにメッセージを送信し，検索が完了する．キーの検索に必要なホップ数はスキップグラフの高さに比例して大きくなり，$O(\log n)$ となる．また，このような検索を行うため，目的のキーが存在しない場合でも最

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

も目的のキーに近い値を持つピアを得ることができる．

2.2.2 スキップグラフへの参加・離脱

新たなピアがスキップグラフに参加することを考える．このとき，既にスキップグラフに参加しているピアに，参加したいピアが持つキーの検索を要求し，レベル 0 で最もキーの値が近いピアを調べる．調べたピアに参加要求を送り，レベル 0 で適切な場所へ参加する．以降はレベル 1 から順に，メンバーシップベクトルの上位ビットが等しいピアをリストの左右にメッセージを送って発見し，順次上位レベルのリストへ参加していく．上位ビットが等しいピアが発見できなくなった段階で処理を終了する．キーの検索処理，およびレベル 0 から上位レベルへと順次参加していく処理のいずれも必要なメッセージ数は $O(\log n)$ となる．

スキップグラフからの離脱は参加時と逆に，最大レベルから順に左右のピアにメッセージを送信し，離脱を行う．ピアの離脱にかかるメッセージ数もグラフの高さに比例して大きくなり $O(\log n)$ となる．

2.3 関 連 研 究

スキップリストを用いた別の手法として SkipNet[4]，スキップグラフの改良として Skip B-Tree[1] や Rainbow Skip Graph[3] 等が存在する．本論文では，通常のスキップグラフに対して提案手法を適用しその比較を行っているが，これらの他の手法に対しても適用することで近接性を考慮し，データ検索に必要な通信時間を削減することができると考えている．

3. 提案スキップグラフ

本論文では，グラフのバランスと近接性とを考慮したスキップグラフの構築手法を提案する．提案するスキップグラフは，各ピアが自立分散的なグラフの再構築処理を繰り返し行うことで構築される．本章では 3.1 節でバランス調整のみを行ったバランススキップグラフを説明し，3.2 節でバランス調整と近接性の考慮の両方を行った提案スキップグラフの説明を行う．

なおバランススキップグラフ，提案スキップグラフ共に，各ピアがグラフへの参加時およびその後定期的に，グラフの再構築処理を独立して行う．

3.1 バランススキップグラフ

スキップグラフでは，グラフを構築するために用いるメンバーシップベクトルの値をランダムに決定している．これにより，ピア数が増加するに従いグラフ全体でのリストのバランスは均一化していく事が期待されるが，逆に局所的にはバランスが極めて悪いリストが生成される可能性もある．本論文では，グラフのバランスを表すための値として，$N_{p,i}$ という値を用いる．

$N_{p,i}$：ピア p が属するレベル i とレベル $i+1$ のリストに共通の部分リストで，p を含む最大のものの大きさ．最小では 1 となる．また，この部分リストを $L_{p,i}$ と表す．

また，$N_{p,i}$ の値が大きい部分をバランスの悪い部分，$N_{p,i}$ の値が小さい部分をバランスの良い部分とし，グラフ全体でこの $N_{p,i}$ の値の平均値が小さいグラフほどバランスの良いスキップグラフであると考える．最もバランスの良いスキップグラフでは，グラフ中の全てのピア，全てのレベルで $N_{p,i}=1$ となる．バランスの悪いグラフでは，各ピア毎のレベルの最大値がまちまちになったり，検索に必要な通信時間・ホップ数が増加してしまうという問題がある．そのため，スキップグラフはバランスが良いものであることが求められる．

図 2，3 は，リストのバランスが良いグラフと悪いグラフの例である．図 2 のグラフでは，レベル i に含まれるピアが，レベル $i+1$ のリストにバラバラに参加しているのに対し，図 3 のグラフではほぼ全てのピアが片方のリストに含まれてしまっている．このため，$N_{p,i}$ の値の最大値は図 2 の範囲では 3，図 3 の範囲では 6 となっており，図 3 はバランスの悪いグラフであるといえる．

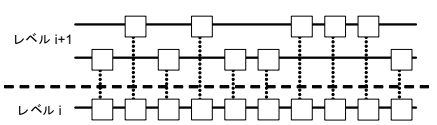

図 2 バランスの良いスキップグラフ例

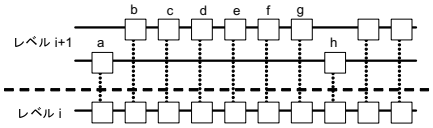

図 3 バランスの悪いスキップグラフ例

ここでは，$N_{p,i}$ が大きい部分を解消するためのグラフの再構築の手法を提案する．あらかじめある定数 K を設定し，各ピア p が各レベル i ごとに $N_{p,i}$ の値を定期的に計測する．その結果，$N_{p,i}$ の値が K を超えている場合は，グラフの部分的な再構築を行い $N_{p,i} \leq K$ を満たすようにする．例えば $K=3$ と設定した場合を考える．このとき図 3 でピア d が逆側のリストに移動すると，図に含まれる範囲では $N_{p,i} \leq K$ を満たすことがわかる．この場合であれば，ピア e が移動してもかまわないし，ピア c とピア f が移動しても条件を満たす．

ピアの移動は，以下の手順によって行う．ここでは，ピア p が $N_{p,i}$ を計測しピアの移動を行うかどうかを判定する．

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

(1) レベル i のリストの左右のピアにメッセージを送信する．このメッセージはレベル $i+1$ でピア p とは異なるリストに属するピアが受信するまで転送され，その返信情報から $N_{p,i}$ の値と，レベル $i+1$ で逆のリストに参加した場合の $N_{p,i}$ の値を調べる．レベル $i+1$ のリストに含まれるピアの数が 1 であるなら，最大レベルに達しているため処理を終了する．

(2) $N_{p,i} > K$ であり，かつレベル $i+1$ で逆のリストに参加した際に $N_{p,i} \leq K$ となるならば，レベル $i+1$ で逆のリストに移動すると決定する．移動後のリストでも $N_{p,i} > K$ となるようであればレベル $i+1$ でピアの移動は行わず，一つ上のレベルで同様の処理を行う．

(3) 移動すると決定した場合，レベル $i+1$ 以上のレベルでピアの離脱処理を行い，レベル $i+1$ の逆のリストで隣接するピアに参加要求を送信する．この際，レベル $i+2$ 以上のレベルでどちらのリストに参加するかを，同様の処理を行うことで順次決定していく．

以上の処理を，全てのピアでレベル 0 において繰り返し行うことによって，最終的にグラフ中の全てのレベルにおいて $N_{p,i}$ の値を K 以下に抑えることができる．

なお，ピアがグラフに参加する場合，およびあるレベルでピアの移動を行った後の上位レベルで参加処理を行う場合，上位レベルにピアが参加していないため $N_{p,i}$ の値が定義できない．そのため，仮想的にどちらかのリストに参加しているものとして参加処理を行う．参加処理を行うための暫定的な決定であるため，実際にはリンクの設置などは行わない．

3.2 提案スキップグラフ

続いて，グラフのバランスと近接性の両方を考慮したスキップグラフを構築するための手法を提案する．スキップグラフでは，あるレベル i でひとつのリストに含まれているピアがレベル $i+1$ で 2 つのリストに分割される．この時に，どちらのリストに属しているほうが通信時間を下げられるかをチェックし，必要があればリストの移動を行う．また通常のスキップグラフと異なり，ピアが含まれるリストの構築に乱数を用いないため，リストに偏りが出る可能性がある．そのため，前述したスキップグラフのバランスを調整する手法をあわせて適用する．

既にグラフに参加している各ピアは定期的に以下の処理を行い，グラフの構造を変化させていく．なお，処理は通信時間を下げる移動判定と，$N_{p,i} \leq K$ を満たすための移動判定の 2 つを行い，その結果グラフの移動を行うか否かを判断する．以下の処理はピア p がレベル $i+1$ でリストの移動を行うかどうかを判定する．

再構築判定 1：近接性の考慮

(1) レベル i でリストの左右にメッセージを送信し，レベル $i+1$ で隣接しているピアとの通信時間，およびレベル $i+1$ で逆のリストに参加した場合の通信時間をそれぞれ計測する．また，再構築処理 2 の準備としてそれぞれのリストに参加した場合の $N_{p,i}$ の値も計測する．メッセージがリストの端に到達した場合は，そちら側の通信時間は 0 とする．

(2) 現在参加しているリストから逆のリストへ移動したほうが通信時間が削減できる場合，リストを移動すると決定する．

再構築判定 1 の後，移動するしないに関わらず結果として $N_{p,i} > K$ となる場合のみ，再構築判定 2 を行う．

再構築判定 2：グラフのバランス調整

(1) 部分リスト $L_{p,i}$ に含まれる各ピアに対し，再構築処理 1 によってレベル $i+1$ でリストを移動した場合の通信時間の変化量をそれぞれ計測するよう要求する．

(2) 通信時間の増加量が最も小さいピアを選択する．ただし，移動した結果のリストで $N_{p,i} > K$ となるピアは選択しない．

(3) 選択されたピアにリストの移動要求を出す．この場合に限り，通信時間が増加してしまうようなピアの移動が発生する可能性がある．このときピア p 自身が選択された場合は，再構築判定 1 での移動の決定が取り消されることとなる．

再構築処理：ピアの移動

再構築処理 1 および 2 の結果移動すべきと判断したピアは，レベル $i+1$ 以上のリストで離脱処理を行い，レベル $i+1$ から順次上位レベルに向かって参加処理を行う．ピアがリストに参加する際も，上記の再構築処理を行い，参加するべきリストを決定する．

以上の処理を，全てのピアでレベル 0 から開始して繰り返し行う．また，ピアが新たにグラフに参加した場合も同様の処理を行い，通信時間と $N_{p,i}$ の値を考慮する．この処理は，一度行った後はグラフの構造が自身の付近で変化しない限り行う必要がない．そのため，再構築判定 1 のはじめの処理の段階でグラフの構造の変化を見つけられなかった場合は以降の処理を行わない．

これらの処理を繰り返して行った結果，スキップグラフの全体で通信時間を下げ，また $N_{p,i}$ の値も K 以下に抑えることができる．グラフの再構築を繰り返し，かつ定期的に行うため，ピアの参加や離脱によって一時的に $N_{p,i} > K$ となる部分が発生しても，いずれはグ

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

ラフの再構築によって $N_{p,i} \leq K$ を満たす構造に作り変えられる．

なお以降では，各ピアが再構築処理を充分な回数行いグラフの任意の場所で $N_{p,i} \leq K$ となったものを提案スキップグラフと呼ぶ．

4. 各処理のアルゴリズム

この節では，提案スキップグラフを用いて行う各種の処理のアルゴリズムを説明する．なお，3.2 節で説明したグラフの再構築処理は充分な回数行われているものとし，グラフのすべての部分で $N_{p,i} \leq K$ が成り立っているものとする．また，K の値はピアの総数と比較して充分小さく，かつあらかじめ決定された定数として扱うものとする．また以降では，ピアの総数を表す値として n_0 を用いる．

各処理にかかる時間はグラフの高さに依存している．そのため 4.1 節においてグラフの高さが最大でも $O(\log n_0)$ で抑えられることを示し，4.2 節以降ではグラフの高さを用いて各処理のアルゴリズムおよび必要な処理時間に関する説明を行う．

4.1 グラフの高さ

スキップグラフでは，キーの検索やピアの参加などの処理に必要な時間はグラフの高さに依存する．そのため，ここではグラフの高さがピア数を n_0 としたときに $O(\log n_0)$ で抑えられることを示す．

[**補題 1**] あるレベルのあるリストに参加しているピア全てが，一つ上のレベルでも同じリストに参加しているということはない．

[**証明**] $N_{p,i} \leq K$ より，レベル i のリストに含まれるピア数が K を超える場合は自明．

そうでない場合，ある瞬間にレベル i に含まれるピアの全てがレベル $i+1$ で同じリストに参加しているとする．いずれかのピアが再構築処理を行うとき，再構築判定 1 において逆のリストに参加した場合の通信時間が 0 と判定され，かつ再構築判定で移動後のリストで $N_{p,i} > K$ となることはありえない．そのため再構築処理を行ったピアは必ずレベル $i+1$ で逆のリストに移動する．また，レベル i でリストにピアが 1 つしか含まれない場合は，そのピアにはレベル $i+1$ は存在しない．

以上より，レベル $i+1$ におけるそれぞれのリストはレベル i におけるリストより，1 つ以上少ないピアしか含まない．

[**定理 1**] ピア数 n_0 の提案スキップグラフの高さは最大でも $O(\log n_0)$ となる．

[**証明**] 任意のピア p を選択する．ピア p がレベル i で属するリストに含まれるピア数を n_i とする．

(1)$n_i > K+1$ のとき n_{i+1} が最も大きくなる場合は，レベル i に含まれるピアが，リストの左端から K 個がレベル $i+1$ のリストに含まれ，1 個が逆のリストに含まれる，という状況を繰り返している場合である．この状況では $n_{i+1} = (\frac{K}{K+1}) * (n_i + 1)$ であるため，常に $n_{i+1} \leq (\frac{K}{K+1}) * (n_i + 1)$ が満たされる．よってレベル 0 におけるピア数を n_0 とすると $n_r \leq n_0 * (\frac{K}{K+1})^r + K$．

$n_r \leq K+1$ となる r のうち最も小さいものは $n_0 * (\frac{K}{K+1})^s + K = K+1$ となる s 以下であり，$s = \log_{\frac{K+1}{K}} n_0$ であるため $r \leq \log_{\frac{K+1}{K}} n_0$．

また，(2)$n_i \leq K+1$ のときは補題 1 より $n_{i+1} \leq n_i - 1$ であるため $n_{i+r'} = 1$ となる r' は $r' \leq K$．

ピア p の最大レベルは，(1) より $n_r \leq K+1$ となる r と (2) より $n_i \leq K+1$ の状態から $n_{i+r'} = 1$ となる r' の合計値であらわされ，$r + r' \leq \log_{\frac{K+1}{K}} n_0 + K$．

K は定数であるため，ピア p の最大レベルは $O(\log n_0)$ となる．

任意のピア p で成立するため，提案スキップグラフの高さ全てのピアの最大レベルのうちの最大値と等しく $O(\log n_0)$ となる．

4.2 キーの検索

キーの検索は最大レベルから行う．レベルを順次下がりつつ，目的のキーに近づいていく．検索を行うピアは，検索開始ピアである自身のアドレスと検索対象のキーを持った検索クエリを作成し，自身に対して渡すものとする．

(1) 検索クエリを受け取ったピアは，自身が持つキーと検索対象のキーを比較する．一致した場合は検索結果として自身のアドレスを検索開始ピアに送信する．一致しなかった場合，目的のキーとリストの左右のキーを比較する．自身が持つキーより検索対象のキーに近いキーをもつピアがリストに存在すれば，そのピアに検索クエリを転送する．なければ，一つ下のレベルに下がる．

(2) レベル 0 において検索クエリの転送が発生しなかった場合，目的のキーが存在しないことになるため，検索失敗のメッセージを検索を開始したピアに送信する．

[**補題 2**] ある検索クエリの転送は，同レベルでは最大で $K-1$ 回しか行われない．

[**証明**] グラフの任意の場所で $N_{p,i} \leq K$ を満たしているため，レベル i であるピア p においてリストの左右いずれも $K+1$ ホップ以内に，レベル $i+1$ でピア p と同じリストに含まれるようなピア p' が存在する．p と p' とはレベル $i+1$ で隣接しているため，検索クエリをピア p' に転送する必要がある場合はレベル $i+1$ において転送が行われている．よって，レベル i において $K+1$ 回以上のメッセージ転送は行われない．

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

[**定理 2**] 任意のキーの検索ホップ数は $O(\log n_0)$.

[**証明**] 定理 1 および補題 2 からメッセージ転送回数の最大値は $(\log_{\frac{K+1}{K}} n_0 + K - 1) * K$ より $O(\log n_0)$.

4.3 ピアの参加

新たにピア p がスキップグラフに参加するための処理である．ピアは一つのキーを持つ．

(1) 自身が持つキーを既にグラフに参加しているピアに検索してもらい，自身に最も近いキーを持つピアを得る．

(2) 検索して得られたピアに対し参加要求メッセージを送信する．自身が挿入されるべき位置の両側のピアと通信し，レベル 0 のリストを再構築する．

(3) レベル 1 から順に，どちらかのリストを選択して参加する．参加するリストは 3.2 節に述べたように，近接性および $N_{p,i}$ の値を考慮して選択する．レベルを順次上げていき，リストに自分自身しか含まれなくなるまで行う．

[**定理 3**] ピアの参加に必要なメッセージ数は $O(\log n_0)$.

[**証明**] ピアの参加はキーの検索と各レベルごとのリストの再構築からなり，キーの検索は定理 2 より $O(\log n_0)$ かかる．あるレベルでリストを再構築する際は左右にたかだか K ホップ先までのメッセージ送信しか必要とならないため，定数時間かかる．全てのレベルでリストの再構築が必要となるため，定理 1 より全レベルでのリストの再構築に必要なメッセージ数は $O(\log n_0)$．よって，ピアの参加が完了するまでに必要なメッセージ数は $O(\log n_0)$.

4.4 ピアの離脱

グラフに参加しているピアが離脱する際の処理である．この離脱処理は任意のタイミングで行うことができるが，故障やネットワークの接続の切断などによる不慮の離脱には対応していない．正規の方法によるピアの離脱，およびグラフの再構築の際に行われる部分的な離脱処理に用いられる．なお，ピアの離脱が起こった場合はその付近で $N_{p,i} \leq K$ が満たされなくなった可能性があるため，そのピアと隣接していたピアはグラフの再構築処理を行う．

(1) リストの左右のピアに離脱メッセージを送る．左右のピア同士が新たにリンクを持てるよう，お互いのアドレスも同時に送信する．

(2) 離脱メッセージを受け取った左右のピアは，新たにリンクを持つピアと通信し，リンクを持つ．リンクを持つことに成功した後，離脱するピアにメッセージを送る．

(3) 左右のピアがリンクを設置したことを受け取った離脱ピアは，そのレベルでの離脱処理を終了する．

(4) この処理を，最大レベルからレベル 0 まで順に行う．

[**定理 4**] ピアの正規の方法による離脱処理に必要な時間は $O(\log n_0)$.

[**証明**] あるレベルで離脱処理を行う際，そのレベルでのリストの左右のピアとメッセージの送受信を 1 回ずつ行うため定数時間かかる．レベルの数が $O(\log n_0)$ であるため，結果として離脱処理にかかる時間も $O(\log n_0)$ となる．

4.5 グラフの再構築

グラフの再構築は 3.2 節の処理を繰り返して行う．あるピアがレベル 0 で再構築処理を開始し，最大レベルまで処理が終了するまでを再構築処理 1 回とみなす．

[**定理 5**] 再構築処理 1 回にかかる時間は $O(\log n_0)$.

[**証明**] 再構築処理を行うピア p は，再構築判定 1 および 2 それぞれにおいて，ピアの移動を行うかどうかを判定するためにレベル i のリストの左右にメッセージを送信する．メッセージはレベル $i+1$ でピア p と同じリストに含まれるピアに転送され，リスト i+1 でピア p と逆のリストに含まれるピアが受け取った時点でピア p に返される．そのため，メッセージの転送が最大回数行われた場合でも $K+1$ 回となる．メッセージの送信も判定 1,2 それぞれで定数回数しか行われないため，レベル i での再構築判定は定数時間で終了する．再構築処理自体はレベル 0 から最大レベルに向かって順次行われるため，定理 1 より $O(\log n_0)$ 回行われる．よって，再構築処理 1 回が完了するまでに必要な時間は $O(\log n_0)$.

5. シミュレーション評価

提案手法によって構築されたスキップグラフを，既存のスキップグラフと比較することで提案手法の有効性を評価する．提案手法では，グラフのバランスを保つ手法と通信時間を削減するための手法を組み合わせて用いている．そのため，通常のスキップグラフと，通常のスキップグラフにバランスを保つ手法をのみを適用したグラフ，両方を適用したグラフの 3 つを用いて比較を行う．以降では，各グラフをそれぞれ通常スキップグラフ，バランススキップグラフ，提案スキップグラフと呼称する．

また，シミュレーションでは物理ネットワークとして Transit-Stub モデル[10] を用い，Transit ドメイン 100 個に Stub ドメインがそれぞれ 100 個ずつ接続した形状を想定する．また物理リンクの遅延時間として Transit ドメイン間の遅延時間を 10ms，それ以外の物理リンクの遅延時間を 1ms として想定している．物理ネットワークは乱数に基づいて生成され，シミュレーションを行うごとに異なる形状を取る．

全てのシミュレーション結果は物理ネットワークを構築する段階から 10 回行い，その平均の値を載せている．

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

5.1 提案手法のパラメータ

提案手法のパラメータ K についての評価を示す．本研究では，グラフのバランスを $N_{p,i}$ を用いて表現している．この $N_{p,i}$ の値が大きい場所はグラフのバランスが悪いと判断できるため，この $N_{p,i}$ の値の許容できる最大値 K をパラメータとして用い，グラフのバランスの悪化を防いでいる．シミュレーションでは，K の値を変えたときにグラフの検索時間と検索ホップ数がどのように変化するかを比較する．いずれの結果もピア数 $n_0 = 8000$ として実験している．

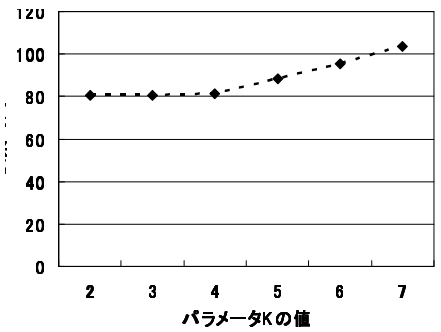

図 4 パラメータ K の値と検索時間の関係

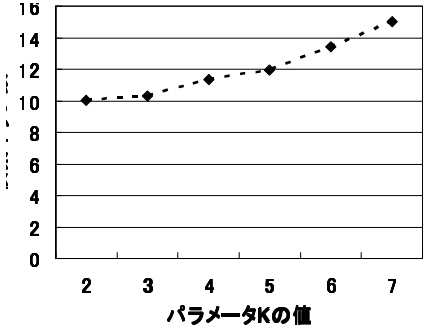

図 5 パラメータ K の値と検索ホップ数の関係

図 4 および図 5 は，それぞれ K の値を変化させたときの検索時間および検索ホップ数のグラフである．K の値が大きければ大きいほど，グラフにバランスの悪い部分が多くなっていくため，検索ホップ数は増加する．また，隣接するピアとの通信時間の平均値は，グラフのバランスの制限が緩和されるため K の値が大きければ大きいほど小さくなるが，検索時間は検索ホップ数の増加の影響で K が大きくなると増加していく．

また，提案手法ではグラフの再構築を行うことで検索時間・検索ホップ数を削減しているが，$N_{p,i} \leq K$ が満たされるまでにどの程度グラフの再構築処理が必要かを，K の値を変化させ実験により調べた．図 6 の横軸は再構築回数の各ピアの平均値であり，縦軸は任意のピア p，レベル i において，$N_{p,i} > K$ である部分リスト $L_{p,i}$ の個数である．異なるピア p, q で部分リスト $L_{p,i}$ と $L_{q,i}$ が同一のリストである場合，この部分リストは一度しかカウントされない．

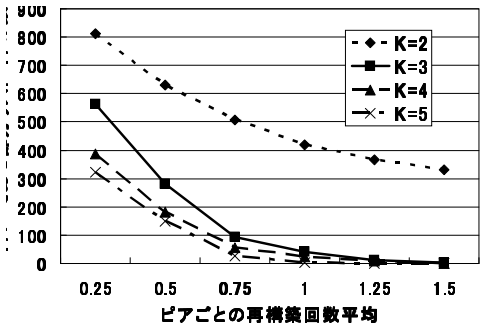

図 6 パラメータ K の値と再構築回数の関係

図 6 より，K の値が 3 以上の場合，$N_{p,i} \leq K$ を満たすまでに必要な再構築の回数はあまり変化が無いことが分かる．ただし $K = 2$ の場合，あるレベル i でどちらのリストに属していても $N_{p,i} \leq K$ が満たされなくなるケースが多いため，再構築処理を繰り返してもグラフが収束し難い．この図は $n_0 = 8000$ の場合の結果であるが，全てのピアで少数回再構築処理を行うことで，通常スキップグラフから提案スキップグラフに組み替えることができることがわかった．再構築処理 1 回にかかる時間は $O(\log n_0)$ であるため，ピア数に比べ必要な再構築時間は十分小さい．そのため，極めて頻繁にピアの参加・離脱が行われない限り，$K > 2$ であればグラフの任意の場所で $N_{p,i} \leq K$ を満たすことができると判断できる．

以上の結果から，以下のシミュレーションでは $K = 3$ をパラメータとして用いる．

5.2 検索ホップ数・検索時間の比較

あるピアからあるキーを検索する時の検索ホップ数および検索時間を比較する．全てのピアから，自身以外のピアが持つキー全てを検索した結果の平均値を示す．バランススキップグラフおよび提案スキップグラフにおける K は 3 としてある．

図 7 より，検索ホップ数に関してはバランススキップグラフが提案スキップグラフよりも若干低い値を示している．検索ホップ数はスキップグラフのバランスが良いほど少なくなることから，提案スキップグラフに比べバランススキップグラフの方がよりグラフのバランスを保つための手法が適切に働いていることになる．提案手法では，ピア間の通信時間を小さくする手法をとっているため，通信時間が小さいピアが隣接した場合，その間に別のピアが

© 2009 Information Processing Society of Japan

Vol.2009-DPS-141 No.5
Vol.2009-GN-73 No.5
Vol.2009-EIP-46 No.5
2009/11/26

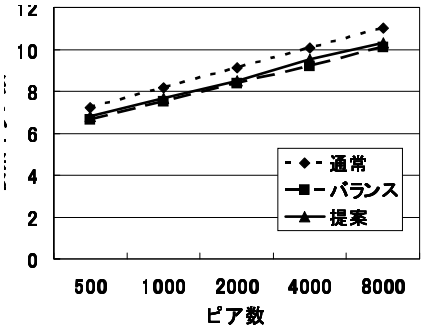

図 7 ピア数と検索ホップ数の関係

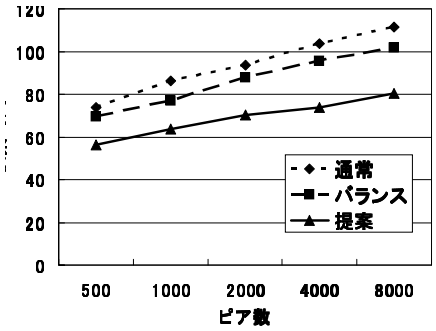

図 8 ピア数と検索時間の関係

入る可能性が小さくなる．その結果グラフ内にバランスの悪い部位が発生しやすくなることが考えられる．しかし，検索ホップ数の差は $n_0 = 8000$ であっても 0.2 ホップ程度であるため，パフォーマンスに大きな影響を及ぼすことは無いと考えられる．

図 8 から検索時間に関しては，提案スキップグラフは通常スキップグラフ，バランススキップグラフよりも小さい値を示していることがわかる．スキップグラフは，ピアの検索を主な目的として用いられる．そのため，検索時間を削減することはシステム全体の動作速度の向上につながる．提案スキップグラフは，通常スキップグラフと比較して 3 割，バランススキップグラフと比較しても 2 割程度検索時間を削減することが出来ているため，提案手法が検索処理に対して有効なスキップグラフを構築することが出来ているといえる．

6. ま と め

スキップグラフを改良し，任意のピアの検索時間を小さく出来るようなスキップグラフの構築手法を提案した．下位のレベルから上位のレベルのリストを構築する際，ピア同士の通信時間を考慮に入れることで，ピアの検索に必要な検索時間を削減した．また，リストの構築の際に乱数を用いないことによるグラフのバランスの悪化の可能性を踏まえ，グラフのバランスを保つための手法を同時に適用した．シミュレーション結果より，提案手法によって構築されたスキップグラフでは，通常のスキップグラフに比べ検索時間が 3 割程度削減できていることがわかった．

提案手法の更なる改善として，正規な方法によらないピアの離脱が発生した際のグラフの再構築の手法を導入し，再び $N_{p,i} \leq K$ が満たされるまでどの程度の処理が必要となるのかを評価したい．またスキップグラフを改良した手法に対しても提案手法を適用し，検索時間の削減を行えるかどうかを確認したい．

参 考 文 献

1) I. Abraham, J. Aspnes, and J. Yuan, “Skip B-trees”, In 9th International Conference on Principles of Distributed Systems (pre-proceedings, pp.284–295, 2005.
2) J. Aspnes and G. Shah, “Skip graphs”, ACM Transactions on Algorithms, vol.3, no.4, pp.37, November 2007.
3) M.T. Goodrich, M.J. Nelson, and J.Z. Sun, “The Rainbow Skip Graph: A Fault-Tolerant Constant-Degree Distributed Data Structure”, Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA ’06), pp.384-393, 2006.
4) N.J.A. Harvey, J. Dunagan, M.B. Jones, S. Saroiu, M. Theimer, and A. Wolman, “SkipNet: A Scalable Overlay Network with Practical proximity Properties”, Proc. 4th conference on USENIX Symposium on Internet Technologies and Systems, vol.4, pp.9–9, 2003.
5) F. Makikawa, T. Matsuo, T. Tsuchiya, and T. Kikuno, “Constructing overlay networks with low link costs and short paths”, Proc. 6th IEEE International Symposium on Network Computing and Applications (IEEE NCA07), pp.299–304, July 2007.
6) W. Pugh, “Skip lists: a probabilistic alternative to balanced trees”, Communications of the ACM, vol.33, pp.668–676, 1990.
7) S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, “A scalable content-addressable network”, Proc. 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM ’01), pp.161–172, 2001.
8) I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for Internet applications”, Proc. 2001 ACM SIGCOMM Conference, pp.149–160, 2001.
9) I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong, “Freenet: A Distributed Anonymous Information Storage and Retrieval System”, Proc. Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability, pp.46–66, 2000.
10) E.W. Zegura, K.L. Calvert, and S. Bhattacharjee, “How to model an Internetwork”, Proc. IEEE Infocom, vol.2, pp.594–602, 1996.

© 2009 Information Processing Society of Japan