Vol.2013-DPS-155 No.15
Vol.2013-MBL-66 No.15
2013/5/23

構造化P2Pネットワークにおけるコンテンツの人気度を考慮したショートカットリンクの生成方法とその評価

成茂 優季[1,a)]　安倍 広多[1]　石橋 勇人[1]　松浦 敏雄[1]

概要：構造化 P2P ネットワーク上に配置されたコンテンツが検索される頻度は一様ではなく，人気の高いコンテンツを持つノード（ホットなノード）に検索が集中する．このため，ホットなノードに対してショートカットリンクを生成することで，コンテンツの検索時間を短縮する手法が提案されている．本稿では，広い範囲の構造化 P2P ネットワークを対象としたショートカットリンク生成法を提案する．提案手法は P2P ネットワーク全体に関する大域的な情報を必要とせず，また，既存の手法よりもショートカット生成のコストが低い．提案手法の有効性はシミュレーションにより評価した．

キーワード：構造化 P2P ネットワーク，人気度，ショートカットリンク

A Method for Creating Shortcut Links by Considering Popularity of Contents in Structured P2P Networks

NARISHIGE Yuki[1,a)]　ABE Kota[1]　ISHIBASHI Hayato[1]　MATSUURA Toshio[1]

Abstract: Considering lookup queries on a structured P2P network, target contents are not uniformly distributed on the network. On the contrary, some specific nodes have very popular –*hot*– contents and receive a large number of queries. Several works propose methods to reduce average search time by creating shortcut links to those hot contents. This paper proposes a method for creating shortcut links that is applicable to variety of structured P2P networks. This method requires no global knowledge of the P2P network and is more efficient than existing works. Effectiveness of the method is experimentally confirmed by simulation.

Keywords: Structured P2P Networks, Popularity, Shortcut Links

1. はじめに

Chord[1] や Skip graph[2] などの既存の代表的な構造化 P2P ネットワークでは，ネットワーク上のどのコンテンツに対しても同等の検索時間を要する．一方，実際の P2P ネットワークではコンテンツの検索頻度（人気度）は一様ではなく，一部の少量のコンテンツに関する検索トラフィックが全体の大半を占めている [3]．このため，人気の高いコンテンツをより高速に検索可能にすることで検索時間を短縮する方法が提案されている．

この方法は 2 種類に大別される．1 つは人気の高いコンテンツの複製をネットワーク中に拡散し，検索経路上に出現する確率を上げる方法であり（複製拡散法と呼ぶ），具体例としては Beehive[4] などがある．もう 1 つは複製を拡散する代わりに人気の高いコンテンツを持つノード（以下，ホットなノードと呼ぶ）へのショートカットリンクをネットワーク中に生成する方法であり（ショートカットリンク生成法と呼ぶ），具体例としては Hot-Chord[5] や重み付き Skip graph[6] などがある．

複製拡散法は，ホットなノードの検索応答への負荷が複製により分散されるという利点がある一方，コンテンツの内容が動的に変化する場合や，送信元ノードが宛先ノードと直接なんらかの通信を行う必要がある場合（例えば構造

[1] 大阪市立大学大学院創造都市研究科
Graduate School for Creative Cities, Osaka City University
[a)] narishige@sousei.gscc.osaka-cu.ac.jp

化 P2P ネットワークを用いて ID/Locator 分離ネットワークを実現する場合など）には適用が困難である．

一方のショートカットリンク生成法は，ホットなノードの負荷は分散しないものの，コンテンツの内容が動的に変化する場合や，送信元ノードが宛先ノードと直接通信する場合にも容易に適用できる．しかし，既存のショートカットリンク生成法は Chord や Skip graph といった特定の構造化 P2P ネットワークを対象としていたり，また，P2P ネットワーク全体に関する大域的な情報（全ノードで最も多く検索されたノードの検索回数など）を必要とするといった問題がある．

本稿では広範囲の構造化 P2P ネットワークに適用可能で，また P2P ネットワークに関する大域的な情報が不要なショートカットリンク生成法を提案する．提案手法では少ないコストでショートカットリンクを生成し，ホットなノードの検索時間を短縮する．提案手法はシミュレーションによって評価した．

以下，2 章で関連研究について述べ，3 章で提案手法のアルゴリズムを述べる．4 章で提案手法の評価と考察を行い，最後に 5 章でまとめと今後の課題を述べる．

2. 関連研究

ショートカットリンク生成法の先行研究として，Hot-Chord と重み付き Skip graph が挙げられる．

2.1 Hot-Chord

Hot-Chord は Chord を対象としたショートカットリンク生成法である．この手法は，P2P ネットワーク全体におけるコンテンツのキーの総数や，1 日あたりの検索総数といった大域的な情報を必要とする．

Hot-Chord について述べられている文献 [5] ではこれらの情報を取得する方法が述べられておらず，実装が困難であったため，本稿では Hot-Chord は比較の対象としない．

2.2 重み付き Skip graph

重み付き Skip graph は Skip graph を対象としたショートカットリンク生成法である．まず Skip graph の概要を述べる．Skip graph では，各ノードにメンバシップベクタと呼ばれる基数 w の乱数が 1 つ割り当てられ，これを用いて接続関係が決定される．Skip graph には複数の level（階層）があり，level $i(i = 0, 1, 2, ...)$ においてメンバシップベクタの接頭部が i 桁一致するノード同士が，それぞれの保持するコンテンツのキーの昇順に接続される．これにより各 level i に，ノードを要素とする w^i 個の分散双方向連結リストが構築される．Skip graph では各ノードがこの連結リストを上から（= level が大きい方から）順に辿ることでコンテンツの検索を行う．

重み付き Skip graph は，各ノードが自身の持つコンテンツが検索される頻度に応じて，メンバシップベクタを複数個持つように Skip graph を拡張する．これにより，頻繁に検索されるコンテンツを持つノードほど多くの連結リストに所属するようになり，他のノードから検索されやすくなる．ノードが保持するメンバシップベクタの数をそのノードの重みと呼ぶ．

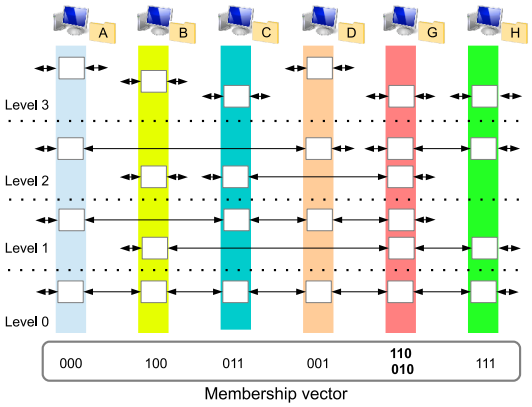

図 1 ノード G を多重化した重み付き Skip graph

各ノードに適切な重みを与える方法としては CutOff と Scaling という 2 つの手法が提案されているが，Scaling のほうが検索速度・メンテナンスコストの面で優れている．ただし Scaling を用いるためには，各ノードが，全ノード中最も多く検索されたノードの検索回数という大域的な情報を取得する必要がある．

図 1 に重み付き Skip graph $(w = 2)$ の構造の例を示す．矢印はノード間の接続関係を表す．ここではノード G の重みが 2，その他のノードの重みが 1 となっているため，ノード G は他と比べて多数のノードと接続されている．

3. 提案手法

提案手法は，すべてのコンテンツがキーによって識別され，コンテンツの検索がキーに基づくマルチホップルーティングにより行われるような構造化 P2P ネットワークに対し，ホットなノードへのショートカットリンクの生成を行うための拡張手法である．提案手法の拡張のベースとなる構造化 P2P ネットワークを「ベース P2P」と呼ぶ．

3.1 基本的なアイデア

一般的にマルチホップルーティングを行う構造化 P2P ネットワークでは，キーの検索経路がホップ数を経るごとに少数のノードに集約されていく．このため，ホットなノードまでの多くの検索経路に同じノードが現れる．そこで提案手法ではそのようなノードからホットなノードに直接ショートカットリンクを生成することで，人気の高いコンテンツの検索ホップ数を減少させる．

まずベース P2P 上のノード u（キー u を保持するノード）は，各ノード x 宛のクエリ Q_x の受信回数をカウント

Vol.2013-DPS-155 No.15
Vol.2013-MBL-66 No.15
2013/5/23

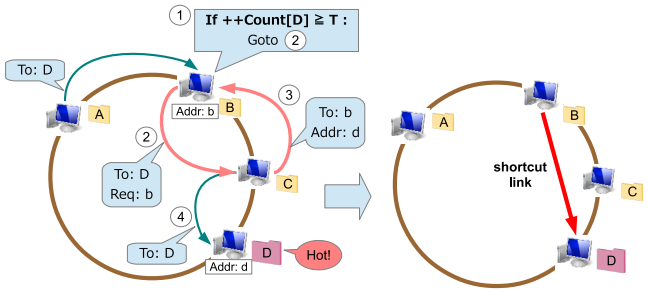

図 2 提案手法の流れ

する．この値が閾値 T 以上になった時点で x をホットなノードと見なし，u から x に対してショートカットリンクを生成する．そのためには，u は x のアドレスを知る必要があるが，これは u が Q_x を次ホップに転送する際，u のアドレスを Q_x に追記しておき，以降 Q_x が x のアドレスを知っているノードに転送されたときに u に x のアドレスを通知することで行う．

以上のように，よく使われる検索経路を閾値で判別すること，クエリに追記することでメッセージ数を節約すること，およびアドレスの返送処理の負荷を転送経路上のノードに分散することが提案手法のアイデアである．

具体例を図 2 を用いて説明する．ベース P2P において，あるクエリがノード A → B → C → D を経路とする場合を考える．A が D 宛のクエリを，隣接する B に転送する．クエリを受信した B は，D に対応するカウンタ値を加算する．ここでその値が T 以上になったとする．B は自身のアドレス b をクエリに追記し，C へ転送する．C は D のアドレス d を保持しているため，アドレス b を参照して B にアドレス d を返送する．これにより B は D へのショートカットリンクを生成する．C が D へクエリを転送する．

3.2 アルゴリズムの説明

提案手法において各ノードが保持するデータ，送信するメッセージ，および実行する動作を説明する．

3.2.1 ノードが保持するデータ

各ノードは過去にクエリを転送した回数を検索キーごとに記録しておくためのカウンタ C と，ホットなノードのキーとアドレスの組を保持しておくためのコレクション H（例：ハッシュ表）を持つ．

3.2.2 ノードが送信するメッセージ

提案手法においてノードが送信するメッセージは ⟨QUERY, destkey, sc_requestors[]⟩ と ⟨NOTIFY, destkey, shortcut⟩ の 2 種類がある．QUERY はコンテンツの検索に用いるメッセージであり，ベース P2P で規定された方法に従ってルーティングされる．destkey は検索するコンテンツのキー，sc_requestors[] はショートカットリンク生成元のアドレスを追記するためのリストである．NOTIFY は，sc_requestors[] に格納されたノードに対する応答メッセージである．

3.2.3 ノードの動作

ノード u がメッセージを受信した際の動作を擬似コードの形で Algorithm 1 に示す．リスト中では，ベース P2P における経路表に含まれるリンクを「base link」，ベース P2P のアルゴリズムに従ったときのクエリ転送先のノードを「next node」と表記している．ここではショートカットリンクを生成する方法だけに注目しており，クエリの宛先ノードがコンテンツを返送する処理は省いている．

Algorithm 1 ノード u の動作

```
1: upon receiving ⟨QUERY, destkey, sc_requestors⟩:
2: u.C[destkey] ← u.C[destkey] + 1
3: if destkey ∈ u.H or u has a base link to destkey then
4:     x ← u.H[destkey] or the base link to destkey
5:     send ⟨QUERY, destkey, sc_requestors⟩ to x directly
6:     for all y ∈ sc_requestors do
7:         send ⟨NOTIFY, destkey, x⟩ to y directly
8:     end for
9: else
10:    if u.C[destkey] ≥ T then
11:        sc_requestors ← sc_requestors ∪ u
12:    end if
13:    send ⟨QUERY, destkey, sc_requestors⟩ to the next node
14: end if
15: upon receiving ⟨NOTIFY, destkey, shortcut⟩:
16: u.H[destkey] ← shortcut
```

3.2.4 例外処理とショートカットリンクの消去

ショートカットリンク先のノードがアドレスを変更していた，あるいはネットワークから離脱していたなどの理由によりクエリに応答しない場合は，そのショートカットリンクを消去し，ベース P2P で規定された方法でクエリを転送する．ベース P2P ではキーをもとにルーティングが行われるため，リンク先ノードのアドレスが変更されていた場合にはこの方法によりクエリが宛先に到達する．

また，ショートカットリンクが蓄積するとメモリ消費量が問題となるため，Least Recently Used(LRU) などの方式を用いてショートカットリンクを消去する．

3.2.5 カウンタについて

ホットなノードは時間とともに変化するため，各ノードが持つカウンタは定期的にクリアする．またカウンタはキーごとに値を保持するため，キー空間が巨大な場合にカウンタのメモリ消費量が問題となる可能性がある．この問題の解決策としては，Counting Bloom Filter[7] などの確率的データ構造を用いて，カウンタ値の正確性と引き換えに空間効率を改善する方法が考えられる．

4. 評価と考察

ベース P2P として Skip graph を用いた提案手法（以下

単に提案手法と呼ぶ），重み付き Skip graph，通常の Skip graph のそれぞれについて，人気度に偏りがある環境におけるコンテンツの検索効率を評価した．

4.1 評価指標

評価は，シミュレーションにより各手法間で以下の 3 つの値を比較することで行った．

4.1.1 総検索ホップ数

検索時間の指標として，総検索ホップ数を用いた．

4.1.2 総メッセージ数

ネットワーク全体の負荷の指標として，総メッセージ数を用いた．Skip graph における総メッセージ数とは全クエリの転送回数の総和であり，提案手法においてはこれに NOTIFY メッセージの総数を加えたものである．

重み付き Skip graph における総メッセージ数は，全クエリの転送回数の総和と，各ノードが重みを増やす（＝新たなリンクを生成する）際に発行するメッセージの総数との和である．重み付き Skip graph において重みを増やすということは，新たな分散双方向連結リストに参加することに等しい．このときに必要となるメッセージの数は分散双方向連結リストを構成するプロトコルに依存するが，ここでは DDLL[8] を用いるものとして計算した．DDLL はノードの参加・離脱が並行して行われる場合でも，分散排他制御を用いずに隣接ノード間のリンクを一貫性を保ちつつ更新でき，また，分散排他制御を用いる他のプロトコルと比較して必要なメッセージ数が少ないという特徴がある．

4.1.3 メッセージの最大送信回数

負荷の偏りの指標として，各ノードにおけるメッセージ送信回数を求め，その最大値を用いた．メッセージの送信回数とは，Skip graph においては各ノードがクエリを転送した回数であり，提案手法においてはこれに NOTIFY メッセージの送信回数を加えたものである．重み付き Skip graph においてはクエリを転送した回数に，重みを増やす際に発行するメッセージの数を加えたものである．

4.2 実験方法

本節では実験の方法を説明する．ノード数を $N_a = 1{,}024$, 総クエリ発行回数を $Q_{max} = 4{,}096$ とする．クエリの発行ノードは 1 単位時間につき全ノードから一様乱数で 1 つ選択し，クエリの宛先ノードはクエリ発行のたびに全ノードから Zipf 分布 [9] に従う乱数で 1 つ選択する． Zipf 分布とは，k 番目に頻度の多い事象の起こる回数が $k^{-\alpha}$ （α は Zipf 係数と呼ばれる）に比例するような逆べき乗分布であり，以下の式で表される．

$$f(k;\alpha,N) = \frac{k^{-\alpha}}{\sum_{n=1}^{N} n^{-\alpha}}$$

N は全要素数であり，Zipf 係数 α が大きいほど分布が高順位に偏る．実験では，k 番目に人気のあるコンテンツに対して $f(k;\alpha,N_a)$ の確率で各ノードがクエリを発行することで，コンテンツの人気度の偏りをシミュレートする．また，α を変化させることにより，人気度の偏りの大きさによる各手法の性能変化を検証する．

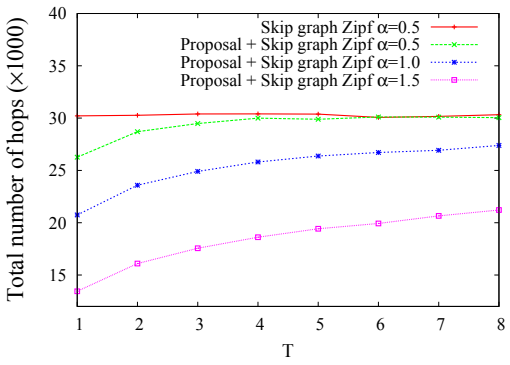

図 3 閾値 T と総検索ホップ数の関係

いずれの手法についても，ベース P2P，すなわち Skip graph の構築のみが完了し，ショートカットリンクが生成されていない状態で実験を開始する．このため，重み付き Skip graph については全ノードの重みが 1 の状態で開始し，適当な時点で重みを増やす．重みの設定方法については 4.4 節で後述する．

4.3 提案手法における適切な閾値

提案手法では，3.1 節で述べたように閾値 T を設定する必要がある．そこで，各手法を比較する前に適切な T の値を評価・考察する．

4.3.1 T と総検索ホップ数の関係

図 3 に T と総検索ホップ数の関係を示す．グラフでは，人気度の偏り（α の値）の大きさに関わらず T が 1 に近いほど総検索ホップ数が減少している．これは，T が小さいほどクエリの発行回数が少ないうちからショートカットリンクが生成されるためである．

4.3.2 T と総メッセージ数の関係

図 4 に T と総メッセージ数の関係を示す．グラフでは，人気度の偏りが特に大きい場合 ($\alpha = 1.5$) を除いて $T = 1$ で総メッセージ数が突出している．これは，実際には特にホットではないノードに対してもショートカットリンクを生成するための NOTIFY メッセージが発行されるためである．

4.3.3 T とメッセージの最大送信回数の関係

図 5 に，T とメッセージの最大送信回数との関係を示す．グラフでは人気度の偏りの大きさによらず $T = 1$ で最大送信回数が突出しているが，これは 4.3.2 節で述べたように NOTIFY メッセージが過剰に発行されるためである．

4.3.4 総括

総検索ホップ数は T が 1 に近いほど減少するが，$T = 1$ とすると総メッセージ数およびメッセージの最大送信回数

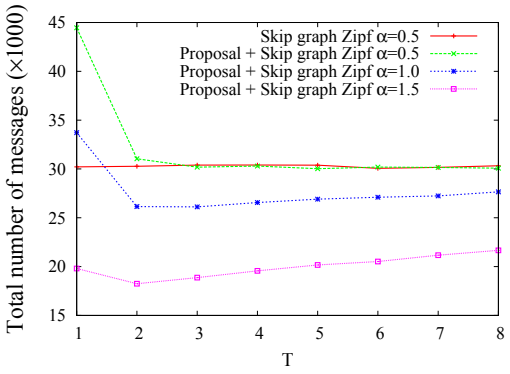

図 4 閾値 T と総メッセージ数の関係

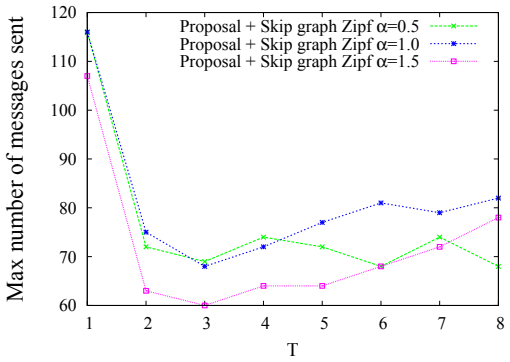

図 5 閾値 T とメッセージの最大送信回数の関係

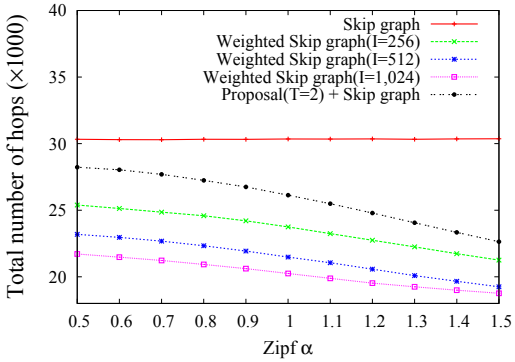

図 6 各手法における総検索ホップ数

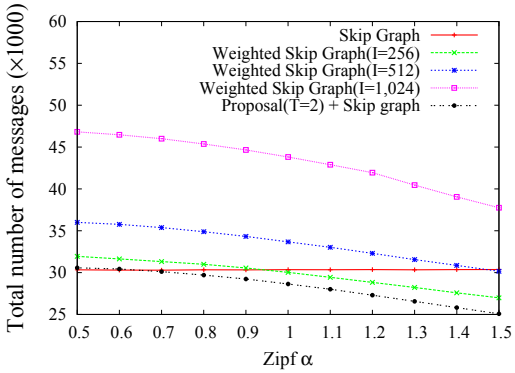

図 7 各手法における総メッセージ数

が大きく増加する．そこで本稿では，$T = 1$ に近く，かつ総メッセージ数およびメッセージ送信回数の抑制に対して良好な結果を示した $T = 2$ を提案手法における適切な閾値とし，次節以降の評価においてこれを用いる．

4.4 重み付き Skip graph の設定

各手法間の比較に際して，重み付き Skip graph に関する設定をいくつか決める必要がある．

重みの上限 MAX については文献 [6] における実験の設定と同じ $MAX = 256$ を採用した．

各ノードに重みを与える方法としては 2.2 節で述べた Scaling を採用した．Scaling を用いるためには，各ノード u は一定時間あたりの自身に対する検索回数 $s(u)$ と，全ノード中最も多く検索されたノードの検索回数 $s(1)$ を知っている必要がある．

本稿では，$s(u)$ を求めるための計測時間 I をパラメータとして導入した．各ノード u は I 単位時間の間に受信した自分宛てのクエリの数を $s(u)$ とする． $s(1)$ の求め方には様々な方法が考えられるが，ここでは詳細には立ち入らず，各ノードはコスト 0 で s(1) を求められるものとしてシミュレーションを行った． 各ノード u は，I 単位時間経過した時点で $s(u)$ と $s(1)$ を用いて重み付けを実行し，新たなリンクを生成する．次節以降では，I を $256, 512, 1024$ の 3 通りに変化させて実験を行った．

4.5 総検索ホップ数の比較

各手法における総検索ホップ数を図 6 に示す．提案手法，重み付き Skip graph のいずれも，人気度に偏りがある（α が大きい）ほど総検索ホップ数が減少している．これはショートカットリンクが有効に機能しているためである．重み付き Skip graph は提案手法と比べて人気度の偏りが小さいうちから高い効果を発揮しているが， これは提案手法では宛先をショーカットリンクが直接指している場合のみそのリンクがルーティングに使用されるのに対して， 重み付き Skip graph では追加されたリンクがクエリの宛先に関わらずルーティングに使用され，人気度の偏りに関係なく検索ホップ数の削減に貢献するためである．

4.6 総メッセージ数の比較

各手法における総メッセージ数を図 7 に示す．提案手法では，人気度の偏りがごく小さい場合を除いて通常の Skip graph よりも総メッセージ数が少ない．これは，ショートカットリンクが利用されたことによる QUERY メッセージの減少量が，NOTIFY メッセージの増加量を上回ったためである． また人気度の偏りに関わらず，重み付き Skip graph は提案手法よりも総メッセージ数が多い． この主な理由は，重み付き Skip graph は重みを増やす際に複数の分散双方向連結リストにノードを挿入するため，多数のメッセージが必要となるためである．

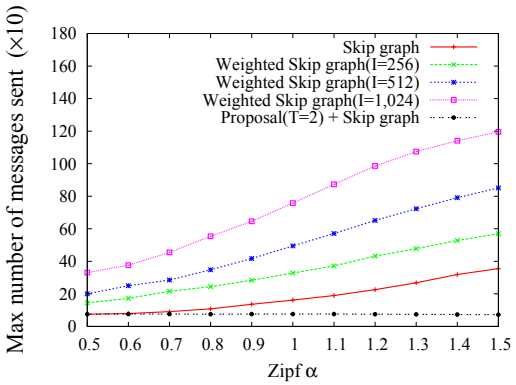

図 8 各手法におけるメッセージの最大送信回数

4.7 メッセージの最大送信回数の比較

各手法におけるメッセージの最大送信回数を図 8 に示す．Skip graph では人気度が偏るにつれてメッセージの最大送信回数が上昇している．これは Skip graph の性質上，ホットなノードとの距離が近いノードほどホットなノードへのクエリのルーティングを担う確率が高くなるためである．

提案手法では，人気度の偏りが大きくなるにつれて，通常の Skip graph よりもメッセージの最大送信回数が減少している．これは，ホットなノードとの距離が近いノードがショートカットリンクにより迂回されたためである．

重み付き Skip graph では，全体的にメッセージの最大送信回数が非常に大きい．これは次の理由による．

(1) 重み付き Skip graph では提案手法と異なり，クエリの宛先となったノード自身が新たなリンクを生成するためのメッセージを発行する．このため，クエリの宛先が偏るにつれて同じノードが何度もメッセージを発行し，メッセージの最大送信回数が増大する．

(2) 重み付き Skip graph において，あるノードが重みを増やす際には，各 level において分散双方向連結リストを辿り，メンバシップベクタが一致したノードとメッセージを交換してリンクを確立する．このとき，ネットワーク上で既に大きな重みを持つ＝多くのメンバシップベクタを持つノードは，「メンバシップベクタが一致したノード」に選ばれる確率が高くなる．したがって，大きな重みを持つホットなノードほど，別のノードが重みを増やす際にメッセージを交換する相手として選ばれやすくなり，結果として一部のホットなノードが更に多くのメッセージを送信することになる．

4.8 考察

提案手法，重み付き Skip graph ともに，コンテンツの人気度が偏るにつれて通常の Skip graph よりも総検索ホップ数を削減できる．提案手法と重み付き Skip graph の相違点として，以下のことが挙げられる．

- 単純な総検索ホップ数の削減量では，提案手法よりも重み付き Skip graph が上回る．
- 総メッセージ数は提案手法のほうが少ない．
- 重み付き Skip graph ではホットなノードにメッセージの送信負荷が集中するのに対して，提案手法ではホットなノード以外に負荷が分散する．

以上より，提案手法は単純な検索ホップ数の削減量では重み付き Skip graph に及ばないものの，総メッセージ数や個々のノードにかかる負荷を総合して考えると，重み付き Skip graph よりも優れていると考えられる．また提案手法は，重み付き Skip graph とは異なり，ネットワーク全体に関する大域的な情報が必要ないという利点も有する．

5. おわりに

本稿では，構造化 P2P ネットワークにおいて，ホットなノードに対してショートカットリンクを生成する方法を提案した．提案手法はマルチホップルーティングを行う構造化 P2P ネットワークに対して幅広く適用可能であり，またネットワーク全体に関する情報を必要としない．

提案手法を Skip graph に適用し，類似研究である重み付き Skip graph とシミュレーションにより比較した結果，提案手法は総メッセージ数や個々のノードにかかる負荷において優れていることを示した．今後の課題として，Skip graph 以外の構造化 P2P ネットワークに適用して評価を行うことが挙げられる．

謝辞 本研究は JSPS 科研費 24500089 の助成を受けている．

参考文献

[1] Stoica, I. et al.: Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications, *IEEE/ACM Transactions on Networking*, Vol. 11, No. 1, pp. 17–32 (2003).
[2] Aspnes, J. et al.: Skip graphs, *ACM Trans. on Algorithms*, Vol. 3, No. 4, pp. 1–25 (2007).
[3] Huang, Y. et al.: Challenges, Design and Analysis of a Large-scale P2P-VOD System, *Proc of ACM SIGCOMM*, pp. 375–388 (2008).
[4] Ramasubramanian, V. et al.: Beehive: O(1) Lookup Performance for Power-Law Query Distributions in Peer-to-Peer Overlays, *Proc. of Networked System Design and Implementation (NSDI)*, pp. 99–112 (2004).
[5] Yuting, M. and et al.: Hot-Chord: A Query Algorithm for Accelerating the Locating of Hot Resources, *2010 6th International Conference on WiCOM*, pp. 1–4 (2010).
[6] 原口高裕ほか：分散データ構造スキップグラフの探索頻度偏りを考慮した拡張について，情報処理学会研究報告., Vol. 2006-AL-105, No. 30, pp. 33–40 (2006).
[7] Fan, L. et al.: Summary cache: A scalable wide-area Web cache sharing protocol, *IEEE/ACM Transactions on Networking (TON)*, Vol. 8, pp. 281–293 (2000).
[8] 安倍広多ほか：構造化オーバーレイネットワークに適した分散双方向連結リスト DDLL，情報処理学会研究報告, Vol. 2010-DPS-144, No. 1, pp. 1–8 (2010).
[9] Zipf, G.: *Human behavior and the principle of least effort*, Cambridge, Mass., Addison-Wesley Press (1949).