Vol.2009-DPS-139 No.1
2009/6/18

範囲をキーとして保持可能とする Skip Graph 拡張の提案

石　芳正[†1]　寺西裕一[†1,†2]　吉田　幹[†3]
下條真司[†2]　西尾章治郎[†1]

範囲探索に対応した構造化オーバレイネットワークである Skip Graph では，キーが単一の値であることを前提としているため，範囲対範囲による探索が行えない．本研究では，Skip Graph を拡張することで，範囲をキーとして保持可能とし，範囲対範囲探索を実現可能とする Range-Key Skip Graph を提案する．また，提案手法を実装し，JGN2Plus により提供された PlanetLab 環境上において評価を行った．

A Proposal of Range-key Extension of Skip Graph

Yoshimasa Ishi,[†1] Yuuichi Teranishi,[†1,†2]
Mikio Yoshida,[†3] Shinji Shimojo[†2] and Shojiro Nishio[†1]

Skip Graph, a structured overlay network supports range retrievals, cannot support range-to-range retrievals since a key corresponds to only one value. To solve this issue, I propose a Range-Key Skip Graph to support range-to-range retrievals by extending Skip Graph and adding range-key in this paper. Finally, I implemented this proposal method and evaluates on a private Planet Lab of JGN2plus.

1. はじめに

近年，インターネットへの高速常時接続環境が整ったことに加え，ワイヤレスネットワークについても整備が進められており，有線での接続が困難な場所や，移動体においても高速な常時接続環境が実現されようとしている．ネットワークに接続される機器についても，PDA や携帯電話といった小型アプライアンスや情報家電も接続対象となってきている．こうした機器は，デバイス技術や無線伝送技術の高度化に伴い，ますます小型化，廉価化，多様化が進むことが予想される．さらに，ワイヤレスネットワーク環境の充実による設置作業の簡素化により，環境中にはセンサやカメラといった多種多様なデバイスが高密度に配置され，それらのデバイスから生じる情報も多様化，増加すると考えられる．

しかしながら，これらの膨大な情報を従来の集中型システムで扱うには設備投資が膨大となり，スケーラビリティを保つことが難しいという問題が生じる．このため，これらのデバイスを自律的に相互接続させ，広域に分散したリソースを効率的に探索可能とするオーバレイネットワークを用いることで，スケーラビリティの高い情報資源共有を目指すための研究開発が盛んに行われている．例えば，分散 Key-Value ストアとして汎用的に利用されている分散ハッシュテーブル（ DHT ）には Chord[1] や Kademlia[2] が挙げられる．また，完全一致探索に加えて範囲探索も実現した Skip Graph[3] や，2 次元範囲の探索を可能とした LL-Net[4] や Delauney Network[5][6] がある．

一方，例えば，センシングデータの共有システムでは，ある地理的な範囲内に設置されたセンサから取得されたデータをシンクと呼ばれるサーバーに集約・蓄積させる形態となる．この場合，シンクサーバは配下のセンサが設置されている地理的な範囲へのセンシングデータの探索要求に応答することとなる．映像カメラのように，観測対象が広がりを持つセンサである場合も同様に，そのセンサの観測対象範囲に対する探索要求に応答できる必要がある．

また，大規模なクラスタシステムにおいては，取り扱うデータ量やトラヒックの増加に追従するために性能強化を迫られるが，高性能な機材を用いることで対応するスケールアップ戦略による性能強化ではコストに見合わなくなってきており，安価なサーバを多数用意しデータや処理を分散させることで性能強化を目指すスケールアウト戦略が採られることが多くなっている．このような分散システムでは，大規模なデータをある範囲を分割し，個々のノードにそれらの範囲のデータを担当させるいわゆるパーティショニングが行われる．

これらのアプリケーションでは，担当範囲に重複が生じたり，冗長化のため敢えて重複させる場合があり得る．さらに，担当範囲が動的に変わる場合や，故障などの要因でノード自体がオーバレイネットワーク上から離脱する場合も考えられる．加えて，取り扱うデータやノード数も膨大となるため，あらかじめ全てのノードの情報を静的に把握することは難しい．

†1 大阪大学大学院 情報科学研究科
Graduate School of Information Science, Osaka University
†2 独立行政法人情報通信研究機構
National Institute of Information and Communication Technology
†3 株式会社ビービーアール
BBR Inc.

Vol.2009-DPS-139 No.1
2009/6/18

このような条件下において,「ある範囲の観測データを保有しているシンクサーバ」「ある範囲を撮影範囲としているカメラ」「大規模データの部分範囲を担当するサーバ」といった範囲を担当するノードを動的に探索可能とする仕組みが必要となるが，既存のオーバレイネットワークでは実現が困難である．例えば， DHT では Key に対する完全一致探索のみでの探索しかサポートしておらず，範囲探索が行えない．また，Skip Graph や LL-Net では，範囲をキーとして持つことはできない．Delauney Network では，ノードが担当範囲を持つことができるが，その範囲は隣接ノードとの位置関係により決まるため，任意の範囲を持つことはできないという問題がある．

そこで本研究では，Skip Graph を拡張することで，上限値と下限値で指定される範囲を持つキー，即ちレンジキーを扱える Range-Key Skip Graph を提案する．本研究では 1 次元範囲についてのみ記すが，地理的範囲のように 2 次元範囲を扱う場合においても Z-ordering 等の 2 次元 1 次元変換を行うことで対応可能である．また，本処理系を実装し，JGN2Plus により提供された PlanetLab 環境上において評価を行う．

2. Skip Graph

Skip Graph は Skip List[7] を分散環境に適用し， P2P システムよる運用を可能としたオーバレイネットワークである．図 1 に Skip Graph によるオーバレイネットワークの構造を示す．図中の，縦に並んだ同色の正方形の組がノード上に保持されるキーを表し，個々のノードには 1 個のキーを保持することとする．個々の正方形はルーティングテーブルのエントリを表し，中の数値がキーの値を表す．キーとする値は数値に限らず，全順序関係を定義可能であればよいが，ここでは簡便化のため整数値を用いている．ルーティングテーブルの各エントリはそれぞれ図中左側に示したレベルに属し，レベルごとに隣接ノードが保持しているキーの値と隣接ノードのアドレスを経路情報を持つ．この構造により，各レベルごとにキーを結ぶ双方向連結リストを構成することとなる．また，各キーは後述する参加・離脱アルゴリズムにより整列されている．各キーの下部に示した 2 進数値は membership vector を示す．ここでは，各キーはそれぞれ一つのノードに対応するため， membership vector が一つのノードに対応している．エントリが各レベルで属するリストは membership vector によって決まる．具体的には，レベル i において membership vector の接頭 i 桁までが等しいエントリ群により各レベルでのリストが構成される．この membership vector をランダムに割り当てることで，各リストに属するエントリ数がほぼ均一に保たれる．

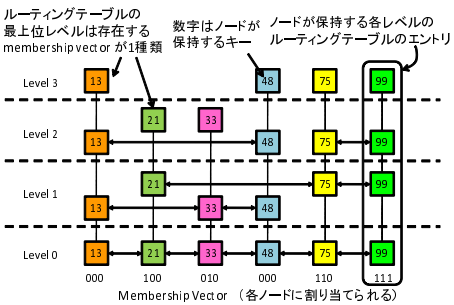

図 1 Skip Graph の構造

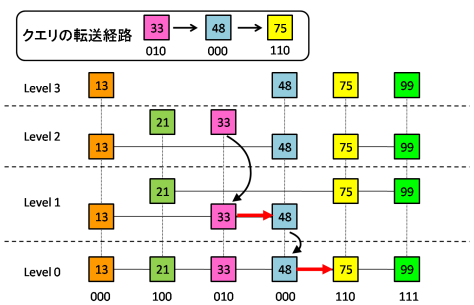

図 2 ノード 010 の キー 33 からキー 75 を探索

2.1 キー探索アルゴリズム

Skip Graph では，単一キーの探索は上位レベルから下位レベルへと行われていく．これは，探索クエリの転送において，上位レベルのリンクを用いた方が下位レベルのそれと比べてキー間の距離が大きく，より効率的に目標とするキーの近傍に探索クエリを転送できるためである．各ノードはクエリの届いたレベルにおいてクエリの値と自身のキーと比較し，左右どちらに転送するかを選択する．次に隣接ノードのキーと比較し，クエリの値を超えない場合はそのノードにクエリを転送する．隣接ノードのキーがクエリの値を超える場合は，1 つレベルを下げ，再び隣接ノードのキーと比較する操作を条件を満たすまで繰り返す．

例えば，図 1 において，キー 33 を持つノード 010 よりキー 75 を探索する場合を考える．（図 2）探索開始ノードであるノード 000 は，キー 75 が自身のキー 33 よりも大きいため，クエリの転送方向を右方向と決定する．そして，ノード上の最上位の Level2 から転送先を探索するが，Level2 ではリンクが無いため，探索レベルを一つ下位の Level1 におとし，そのレベルにおいて右側の隣接ノード 000 が持つキーを参照する．この場合，キー 75 より小さい キー 48 であるため，このノード 000 にクエリを転送する．クエリを受け取ったノード 000 は レベル 1 を開始レベルとして，先と同様の操作により探索を進める．結果として Level0 でクエリを受けたノード 110 において探索クエリのキー 75 と自身のキー 75 が一致するため，探索を開始したノードに応答を返し，クエリは図 2 の通り転送されることとなる．

範囲探索を行う場合は，単一キーの探索と同様の手法により，まずクエリの範囲に含まれるキーを持つノードに到達するまでクエリの転送を行う．目的とするノードに到達すると，自身のキーを境界としてクエリの範囲を左右 2 個に分割する．分割されたそれぞれの範囲

のクエリについて，元のクエリが届いたレベルより下位レベルのリンクを調べ，左右それぞれのクエリの範囲に含まれるキーを持つノードを発見し，それらのノードに分割されたクエリを転送する．この操作を繰り返すことにより，範囲探索が実現される．これらのキーの探索における平均ホップ数は，キー数を n として $\log(n)$ となる．

2.2 キーの追加・除去アルゴリズム

Skip Graph におけるキーの追加と除去は，ノードのオーバレイネットワークへの参加と離脱と同義となる．新規キーを Skip Graph オーバレイネットワークに追加するには，既にオーバレイネットワークに参加している任意ノードに対して参加要請を行う．参加要請を受けたノードは，新規ノードのキーを探索し， Level0 における新規ノードの隣接ノードを発見し，新規ノードを Level0 のリストに参加させる．次に，Level$i+1$ における隣接ノード候補を知るため，Level$i$ において自身の隣接ノードを介して， membership vector の接頭 $i+1$ 桁が同じノードを探索する．該当するノードが発見された場合，そのノードに対して Level$i+1$ でのリンクを構築する．さらに，Level$i+1$ においても同様の操作を行ない，さらに上位レベルでの隣接ノードを探索する．隣接ノード候補が存在しなくなるまでこの操作を繰り返す．ノード参加における平均メッセージ数のオーダーはキー数を n として $O(\log n)$ である．

ノードが離脱する際は，まず Level0 以外のレベルで離脱を隣接ノードに告知し，リンクを切断する．その後， Level0 での隣接ノードに離脱を告知し，ネットワークから離脱する．ノード離脱における平均メッセージ数のオーダーは $O(\log n)$ となる．

3. Skip Graph のレンジキー拡張

Skip Graph のレンジキー拡張は，Skip Graph の持つ基本アルゴリズムを基に，範囲をキーとして扱い，なおかつそのキーを探索により発見可能とすることに相当する．ここではその実現手法について述べる．

3.1 レンジキーの探索要件

レンジキー RK が持つ範囲の最小値を RKs，最大値を RKe とし，範囲探索クエリ Q が持つ範囲の最小値を Qs，最大値を Qe とした場合，以下の 3 条件のいずれかを満たした場合に探索クエリ Q によりレンジキー RK を発見可能としなければならない．

(1) $RKs \leq Qe \leq RKs$ （図 3（ a））
(2) $RKs \leq Qs \leq RKe$ （図 3（ b））

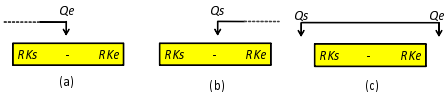

図 3 範囲探索クエリに合致するレンジキー

表 1 Skip Graph と Range-Key Skip Graph の対比

	Skip Graph	Range-Key Skip Graph
オーバレイが持つキー	数値や文字列などの特定の値（キー）	範囲を持った情報（レンジキー）
クエリ	単一値, 範囲	単一値, 範囲
探索対象	クエリにより指定された範囲に含まれるすべてのキー	クエリにより指定された範囲とレンジキーの持つ範囲に共通範囲を持つすべてのレンジキー

(3) $Qs \leq RKs$ かつ $RKe \leq Qe$ （図 3（ c））

単一値探索クエリの場合は，範囲探索クエリにおける $Qs = Qe$ の場合と同等となる．Skip Graph との差異を表 1 に記した．

3.2 レンジキーの表現

ここでは 3.1 節で述べた条件を満すレンジキーの実現方法について述べる．Skip Graph は，全順序関係を持つデータが分散するネットワークの中から，目的のデータを探し出すためのアルゴリズムである．この機構を活用するためには，レンジキーが持つ “範囲” を全順序関係を保つ形に変換し， Skip Graph の中で取り扱える形にすることが必要となる．そのためのアプローチとして次の 2 つが考えられる．

(1) 範囲を互いに素な部分範囲に分割し，分割した個々の範囲が全順序関係を満たすようにする．
(2) 範囲の持つ特定の部分をキーとして，全順序関係をなすよう順序定義をする．

1 はレンジキーの範囲を加工し，一般的な順序を定義できる形にする手法で，図 4 に示したようにレンジキーを登録する際に，既に Skip Graph 上に存在しているレンジキーの範囲と重複する部分については，双方が完全に重複する部分範囲と全く重複しない部分範囲に分割することで全順序関係を維持する．これにより， Skip Graph の探索法をそのまま流用

Vol.2009-DPS-139 No.1
2009/6/18

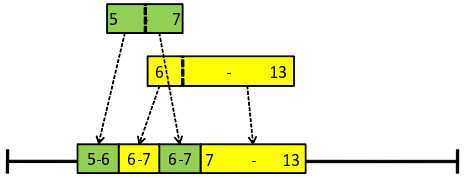

図 4　範囲分割によるレンジキー

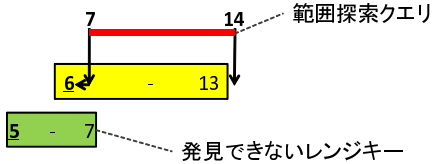

図 5　合致条件に合うレンジキーを取得しきれない事例

することができる．しかしながら，レンジキーを登録する際に既存のレンジキーの削除・分割・再登録を含めた分割処理が必要になり，また，レンジキーを削除する際にも分割されていたキーの統合処理が必要となるため，実装が複雑になる．また，分割により Skip Graph 上で管理するキー数が増大することによるオーバーヘッドも予想される．

2 は，レンジキーの範囲自体は加工せず，範囲内の特定の値を順序関係を確立するためにを用いるアプローチである．範囲内の特定の値をキーとして用いることで Skip Graph におけるキー登録操作を流用できるが，探索時に正確にレンジキーを発見できない場合が生じる．レンジキー RK の範囲内の特定の値を RKv とすると，$Qs \leq RKs \leq Qe \leq RKv$ や $RKv \leq Qs \leq RKe \leq Qe$ という場合には合致条件を満たすもののレンジキー RK が発見できないこととなる．そのため，ここでは他の仕組みを加えることで，範囲に対する探索を Skip Graph 上で実現することを考える．

まず，レンジキーの範囲内の特定の値 RKv であるが，これは $RKv = RKs$ か $RKv = RKe$ とすることが望ましい．例えば，レンジキー RK と範囲探索クエリ Q が $RKs \leq Qs \leq RKe$ の関係にある場合，$RKv = RKs$ とすることで Skip Graph における範囲探索操作に加え，Qs を越えない最大のキーを探索をすることで RK を発見できる．このある値を超えない最大のキーを探索する探索操作は Skip Graph の探索操作を部分的に修正するだけで容易に実現できる．ここで，RKv を $RKs < RKv < RKe$ を満たす任意の値

表 2　レンジキーが一様分布する場合における包含キー情報所持方式の対比

	加味されるホップ数	メッセージ数	エントリ数
1	1	$O(N)$	$O(N)$
2	$O(\log N)$	$O(\log N)$	$O(\log N)$

とした場合，Qs，Qe に対して RKv がどちらにあるかを確定できないため，Qs を越えない最大のキーと Qe を越える最小のキーをそれぞれ探索する必要が生じる．RKv をレンジキーの範囲の境界値とすることで，この探索方向を片方向に固定でき探索時のメッセージ数を削減できる．本研究では，RKv にはレンジキーが持つ範囲の最小値を用いる．

次に 2 個のレンジキー R_1 と R_2 の範囲が重複している場合を考える（図 5）．R_1 の最小値と最大値をそれぞれ R_1s と R_1e，R_2 の最小値と最大値をそれぞれ R_2s と R_2e として，$R_1s \leq R_2s \leq Qs \leq R_1s \leq R_2s$ となる場合において，R_1，R_2 ともに探索条件に合致しているが，R_1s は Qs を越えない最大のキーではないためレンジキー R_1 が発見できない．この問題は 3 個以上のレンジキーが重複する場合においても生じる．

この問題に対しては，レンジキー R_2 より，自身の最小値 R_2s を含む範囲を持つ他のレンジキー（これを包含キーと呼ぶ）に探索クエリを転送することで対応する．これには，個々のレンジキーにそれぞれ包含キーの範囲とその包含キーに対するリンク情報を持たせる必要がある．この包含キーの情報の持ち方として，次の 2 手法が考えられる．

(1)　自身の包含キーの情報を全て持つ手法．
(2)　最低限の包含キーの情報のみを持つ手法．

1 では，探索クエリを受け取ったレンジキーが自身の包含キー全てに対して直接探索クエリを転送する手法，2 では，ある包含キー CK_x が他の包含キー CK_y の情報を保持している場合，CK_x にのみ探索クエリを転送し，包含キー CK_x が改めて包含キー CK_y に探索クエリを転送する手法となる．これらの方式の優劣はレンジキーの分布状態により異なる．

一定長 r のレンジキーがキー空間 [0，R] に一様に分布する場合，それぞれの手法に要するコストは表 2 となる．N はレンジキー数を表す．ここで，エントリ数 $O(N)$ は $r/R \times N$ となり，緯度経度を Z-ordering により 1 次元化したキー空間を想定すると問題なく扱える値となるが，気温のようにキー空間が狭い場合を想定すると，エントリ数は N に近い値となり，アルゴリズムとして問題が生じる．[⋆1] この場合においても，方式の優劣は対象とするアプリケーションに依存することとなる．

⋆1 どちらの例もレンジキーは一様に分布しない．正確に計算するためにはレンジキーの分布を考慮する必要がある．

Vol.2009-DPS-139 No.1
2009/6/18

表 3　レンジキーの重複部が少ない場合における包含キー情報所持方式の対比

	加味されるホップ数	メッセージ数	エントリ数
1	1	$O(1)$	$O(1)$
2	≒ 1	$O(1)$	$O(1)$

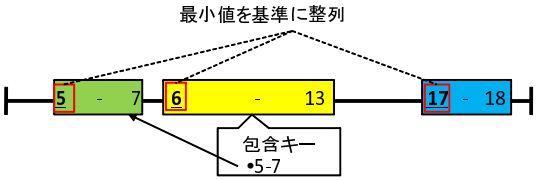

図 6　Range-Key Skip Graph におけるキー配置

また，センサデータの共有システムのように，各ノードが地理的な位置を持ち，ノードの分布状態に従って各ノードの担当範囲を調整した場合，即ち，ノード密度の高い部分ではレンジキーの範囲を小さく，ノード密度の低い部分ではレンジキーの範囲を大きくした場合については表 3 となる．本研究ではセンサデータの共有システムへの適用を考慮し，1 の手法を採用した．

以上に基づいてレンジキーをキー空間上に配置した状態を図 6 に示す．矩形はレンジキーを示しており，図では，[5，7]，[6，13]，[17，16] の範囲を持つレンジキーが登録されている状態を示している．それぞれの最小値 5，6，17 の順にレンジキーが整列されており，Skip Graph 上にはこれらの最小値が登録される．これらの値は図中では太字で示している．また，レンジキー [6，13] は レンジキー [5，7] の最小値 5 を含むため，包含キー情報として [5，7] とそのキーを持つノードへのリンク情報を保持している．

3.3　レンジキーの探索アルゴリズム

レンジキーの探索アルゴリズムを図 7 に示す．まず，クエリで指定された範囲 [Qs，Qe] に対して，範囲探索クエリを発行する．これにより範囲 [Qs，Qe] にレンジキーの範囲の最小値が含まれるレンジキーが取得できる．Qe より大きい範囲に最小値を持つレンジキーは，合致条件から外れる．逆に Qs より小さい範囲に最小値を持つレンジキーの中には，合致条件に合う可能性があるため，Qs を越えない最大の最小値を持つレンジキーを探し，さらに発見されたレンジキーが持つ包含キー情報を介して合致条件に合うレンジキーを探索する．

図では，$Qs = 7$，$Qe = 14$ なので，まず範囲 [6，14] に対する範囲探索クエリを発行す

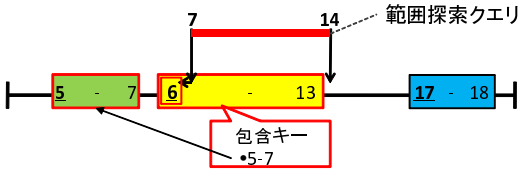

図 7　Range-Key Skip Graph におけるレンジキーの探索

るが，この範囲内に最小値を持つレンジキーは存在しない．続いて，$Qs = 7$ を越えない最大の最小値を持つレンジキーを探索すると，レンジキー [6，13] が発見される．このレンジキーが持つ包含キー情報 [5，7] と探索クエリの範囲を比較すると，探索範囲内に含まれていることが分かるため，探索クエリをレンジキー [5，7] に転送する．以上の手順により，範囲探索クエリ [6，14] に合致するレンジキー [6，13] とレンジキー [5，7] が発見される．

3.4　レンジキーの追加・除去アルゴリズム

レンジキーの追加アルゴリズムを図 8 に示す．まず，追加しようとしているレンジキー RK の範囲の最小値 RKs を包含するレンジキーを探索する．これは，先に示した探索手法による実現できる．この探索により得られた包含キー情報を，レンジキー RK に保持させる．次に RKs をキーとして，Skip Graph の手順により登録する．

図 8 の例では，追加しようとしているレンジキー [7，11] の最小値 7 で探索することで，この値を包含している登録済みのレンジキー [6，13] とレンジキー [5，7] が見つかる．これらの包含キー情報をレンジキー [7，11] に付与した後，7 を Skip Graph のキーとして追加する．これにより，図下側のキー配置となり追加操作が完了する．

次に，除去アルゴリズムを図 9 に示す．まず，除去しようとしているレンジキー RK が持つ範囲で探索を行うことで，レンジキー RK を包含キーとして保持しているレンジキーが発見できる．そして，発見されたレンジキーの包含キー情報から，レンジキー RK の情報を削除した後，レンジキー RK の範囲の最小値 RKs を Skip Graph の手順により除去する．

図 9 の例では，除去しようとしているレンジキー [6，13] の範囲で探索することで，レンジキー [7，11] が発見される．このレンジキー [7，11] は，レンジキー [6，13] を包含キーとして保持しているため，レンジキー [7，11] に対してレンジキー [6，13] の情報の除去を依頼する．その後，Skip Graph の手順によりレンジキー [6，13] の範囲の最小値 6 を除去することで，図下側のキー配置となり除去操作が完了する．

Vol.2009-DPS-139 No.1
2009/6/18

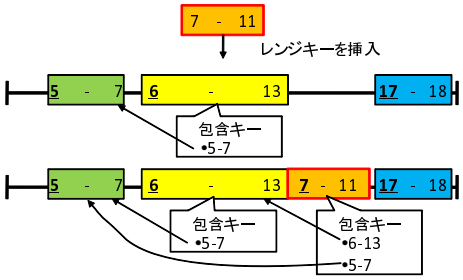

図 8　Range-Key Skip Graph におけるレンジキーの追加

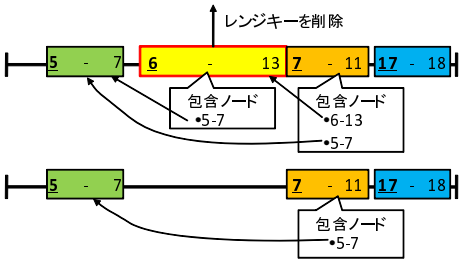

図 9　Range-Key Skip Graph におけるレンジキーの除去

4. 評　　価

Range-Key Skip Graph の動作検証と性能確認のため評価を行った．範囲ごとに分割可能な膨大なデータを扱う場合を想定し，Multi-Key Skip Graph[8] により個々のデータをそれぞれキーとして扱う場合と，Range-Key Skip Graph により担当範囲内のデータをレンジキーとしてまとめて扱う場合の差異を評価する．Multi-Key Skip Graph は，Skip Graph を拡張し，単一のノード上に複数のキーを保持可能とした上で，効率のよい探索を実現する手法である．評価には JGN2plus が運営する PlanetLab オーバレイテストベッド上の 10 台のサーバを利用し，範囲探索時の最大ホップ数，メッセージ数，探索応答時間の実測データを取得する．

Range-Key Skip Graph では，レンジキーの範囲重複の有無によりクエリ転送手順が異なるため，キーが重複しない状態（図 10）とキーが重複する状態（図 11）の 2 状態において実測を行う．図 10 の場合では，ノード 0 が 0 ～ 1000，10000 ～ 11000…，ノード 1 が 1000～2000，11000～12000… という形で，全 10 ノードにわたり範囲が重複しないようにレンジキーを配置した．各ノードはそれぞれ 10 個ずつレンジキーを持つ．図 11 の場合では，ノード 0 が 0 ～ 1000，1500 ～ 2500…，ノード 1 が 150～1150，1650～2650… という形で，隣接するノードが持つレンジキーと 85 % を重複させて配置した．この配置においても各ノードはそれぞれ 10 個ずつレンジキーを持つ．比較に用いる Multi-Key Skip Graph では，これらのレンジキーの範囲に対して，通常の単一値によるキーを 1 間隔で登録した．すなわち，各ノードが 1 万ずつデータを保持している状況であり，合計 10 万のデータが保持されている状況である．

探索対象とする範囲は，範囲の始点をランダムに選択し，範囲幅は 300 を上限としたランダム値とし，キーが存在する範囲内に納まるように制限した．この範囲を探索範囲としたクエリを，ノードをランダムに選択しながら 70 回発行し，各探索処理に要した最大ホップ数，メッセージ数，最小応答時間，最大応答時間の平均値を評価値としている．最小応答時間とは，探索結果の一部が最初に探索元ノードに届くまでの時間，最大応答時間とは，探索結果の最終部が探索元ノードに届くまでの時間を意味している．図 12 に最大ホップ数，図 13 にメッセージ数，図 14 に最小応答時間，図 15 に最大応答時間を示す．

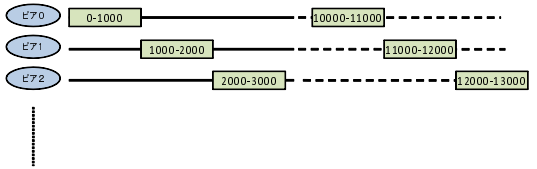

図 10　評価データイメージ（重複幅 0）

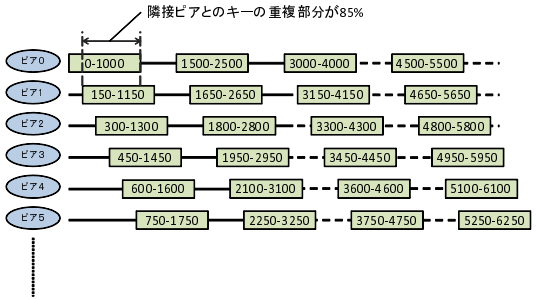

図 11　評価データイメージ（重複幅 850）

最大ホップ数は，範囲重複がない場合において，Range-Key Skip Graph，Multi-Key Skip Graph ともにほぼ同じホップ数となっている．一方，範囲重複がある場合，Multi-Key Skip Graph での 6.8 ホップに対し，Range-Key Skip Graph では 4 ホップと少ないホップ数で探索を終えている．メッセージ数では，重複の有無に関係なく Range-Key Skip Graph が Multi-Key Skip Graph よりも少ないメッセージ数で探索を終えている．最小応答時間は，範囲重複が無い場合において，Multi-Key Skip Graph が約 4.2 ms で応答を返しているのに対し，Range-Key Skip Graph では約 1.1 ms で応答を返し始めている．また，範囲重複がある場合では差は縮まるものの Range-Key Skip Graph が早く応答を返し始めている．最大応答時間は，範囲重

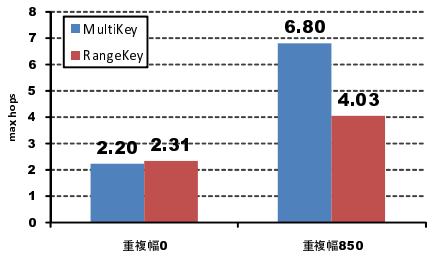

図 12　最大ホップ数

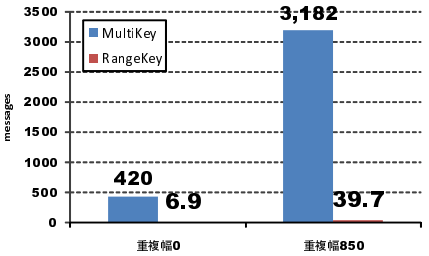

図 13　メッセージ数

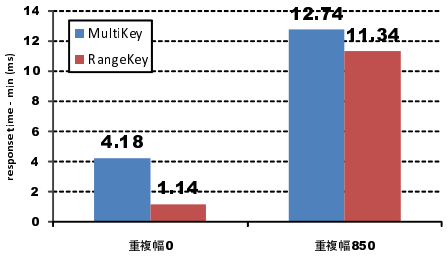

図 14　最小応答時間

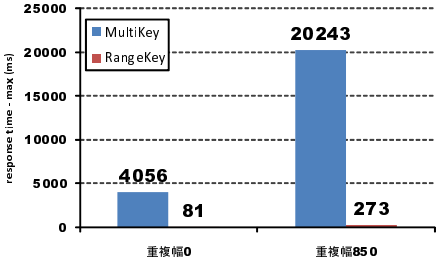

図 15　最大応答時間

複の有無に関係なく Range-Key Skip Graph が Multi-Key Skip Graph よりも短時間で応答を終えている．

Multi-Key Skip Graph においては，個々のデータがそれぞれのキーを登録するため，探索範囲においてノード内での探索操作が複雑になり，これが最小応答時間の差として表れたと考えられる．また，範囲内に複数のキーが発見され，その結果がキーごとに探索要求元のノードに送り返されることで，多数のメッセージが発生し，それらの転送にもより多くの時間がかかることが最大応答時間に現れている．

5. ま と め

本研究では，構造化オーバレイネットワークである Skip Graph を拡張することで，上限値と下限値で指定された範囲をキーとして持つ Range-Key Skip Graph を提案した．この拡張により，従来の値対値，範囲対値の探索に加え範囲対範囲による探索も可能とした．また，JGN2plus PlanetLab テストベッドにおいて Multi-Key Skip Graph との比較評価を行った．

今後の課題として，ノード数やキー数が増加した場合における動作検証や性能評価があげられる．また，包含キーを介したレンジキーの探索手順において，現時点ではあるレンジキーの包含キー情報はそのキーが全て所持し，個々のレンジキーに探索クエリを転送しているが，範囲の分布状態が異なる場合も考慮して，他のレンジキーが持つ包含キーを利用してツリー構造等で探索クエリを転送するといった対応を検討する予定である．

謝辞　本研究の一部は，平成 20 年度総務省委託研究「ユビキタスサービスプラットフォーム技術の研究開発」による成果である．また，評価に際し独立行政法人情報通信研究機構が運用する JGN2Plus の PlanetLab オーバレイテストベッドを利用した（JGN2P-A20089：「PIAX に基づく広域オーバレイネットワークを用いたユビキタスサービスプラットフォームの検証」）．ここに記して謝意を表す．

参　考　文　献

1) Morris, R., Karger, D., Kaashoek, F. and Balakrishnan, H.: Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications, *ACM SIGCOMM 2001*, San Diego, CA (2001).
2) Maymounkov, P. and Mazieres, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric, *the 1st International Workshop on Peer-to-Peer System(IPTPS'02)*, p.263 (2002).
3) Aspnes, J. and Shah, G.: Skip Graphs, *ACM Transactions on Algorithms*, Vol.3, No.4, p.37 (2007).
4) 金子雄，　春本要，福村真哉，下條真司，西尾章治郎：ユビキタス環境における端末の位置情報に基づく P2P ネットワーク，情報処理学会論文誌. データベース，Vol.46, No.18, pp.1–15 (2005).
5) Araujo, F. and Rodrigues, L.: GeoPeer: A Location-Aware Peer-to-Peer System, Proc. The 3rd IEEE International Symposium on Network Computing and Applications (NCA ' 04), pp.39–46 (2004).
6) 大西真晶，源元佑太，江口隆之，加藤宏章，　西出亮，上島紳一：ノード位置を用いた P2P モデルのためのドロネー図の自律分散生成アルゴリズム，情報処理学会論文誌：データベース，　Vol.47, pp.51–64 (2006).
7) Pugh, W.: Skip Lists: A Probabilistic Alternative to Balanced Trees, *Workshop on Algorithms and Data Structures*, pp.437–449 (1989).
8) 小西佑治，　吉田幹，　竹内亨，寺西裕一，　春本要，下條真司：単一ノードに複数キーを保持可能とする SkipGraph 拡張，情報処理学会論文誌，　Vol.49, No.9, pp.3223–3233 (2008).