G-42

複数の異なる時間間隔に基づく 構造化オーバレイネットワークでのノード仮想化の検討

Node Virtualization for Structured Overlay Networks Based on Multiple Different Time Intervals

久保 達也† 川上 朋也†
Tatsuya Kubo Tomoya Kawakami

1. はじめに

IoT（Internet of Things, モノのインターネット）[1]においては，多種多様でかつ多くのデバイスがネットワークに接続され，それらの IoT デバイスからは膨大な量のデータが送信されることとなる[2]．この送信されるデータの種類としては，例えば温度や電力消費量のようなセンサモジュールを利用して測定した測定データが考えられる．測定データは測定された地点や範囲という空間の情報や，測定日時といった時間情報に深く結びついたデータであるという特徴があり，データを利用するときにはそれらの情報に基づいてデータを選択する．そのため，膨大かつ流動的な量のノード（デバイス）やデータ，そして利用者を扱うための高いスケーラビリティを保ちつつ，要求されたデータを高速で検索可能なネットワークの仕組みの実現が必要となる．

この高いスケーラビリティを持つネットワークの実現手法として，分散ハッシュテーブル（Distributed Hash Table, DHT）や，地理的な位置情報に基づいてデータを分配するオーバレイネットワークなどの構築手法が提案されている[3-14]．これらのオーバレイネットワーク構築手法は各ノードの持つ一次元の ID，または多次元の情報に基づいてノード間に理論ネットワークを構築する．ノードの探索はキー（key）と呼ぶ値を単一または範囲で指定することで行う．各ノードは自身が持つ論理ネットワークの構造を示す経路表を参照することで，該当するノードへ到達できるようにクエリを転送していく．既存のオーバレイネットワーク構築手法では一次元の時間情報や二次元の位置情報に基づいてノード間に論理ネットワークを構築し，そのネットワーク上でさらに各ノードに担当範囲を割り当てていくことでセンサデータの分散管理や検索を実現できる．その際，利用者やシステムは「過去の 8 月の気温を知りたい」「過去の日曜日の電力消費量を抜き出す」といったある時間的な間隔を指定してセンサデータを要求することが考えられる．図 1 のように特定の時間間隔を持つデータ要求のことを本研究では「間隔クエリ（interval query）」と呼ぶ．間隔クエリで指定されるデータは年，月，週，日，時間などの単位となることが考えられるが，既存のオーバレイネットワーク構築手法ではこれら複数の時間の間隔による検索を効率的に行うことはできない．

筆者は現在，年，月，週，日，時間などの間隔クエリを効率的に処理可能なオーバレイネットワーク構築手法を提案している[15-17]．この提案手法では既存の Chord[3]と同様に環状のオーバレイネットワークを構築し，そこに一次元の時間情報に基づいてキー空間上にノードを配置する．さらに，時間の間隔に基づくクエリを効率的に処理するため，各ノードはキー空間上で年，月，週，日といった間隔の先にあるデータを保持するノードへショートカットの仮想リンクを構築する．それら複数の間隔で構築した仮想リンクを用いることで，提案手法では要求されるデータを担当するノードまでの転送ホップ数を削減できる．本研究では，オーバレイネットワーク上において負荷分散のために一つの物理的なマシンが複数のノードを持つノード仮想化についての検討を行う．

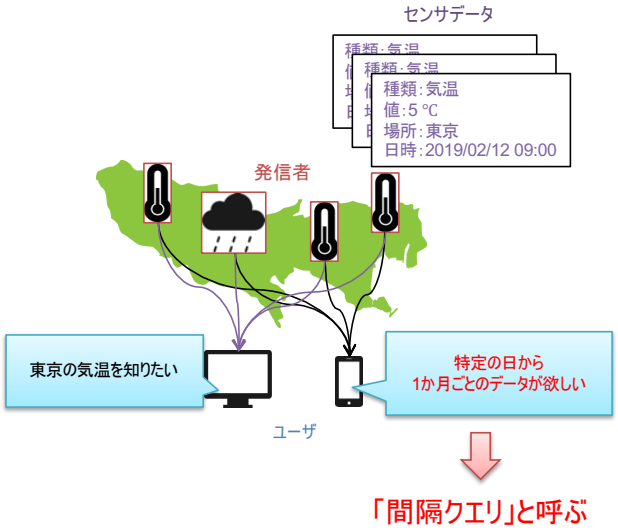

図 1：特定の時間間隔を持つデータ要求

以下，本研究に関連する既存手法である Chord について 2 章で述べる．本研究の提案手法は 3 章で説明し，その評価について 4 章で述べる．5 章では関連研究について，まとめと今後の課題を第 6 章で述べる．

2. Chord

Chord は代表的な環状のオーバレイネットワークであり，各ノードの ID 等に基づいてキー空間上にノードを一次元に配置する．キーの長さ（ビット数）はネットワークに加わるノードの総数を想定してあらかじめ定め，その両端を結んで環状のネットワークとする．Chord ではあるノードからキー空間上で時計回りの次のノードを successor，反時計回りで次のノードを predecessor と呼び，各ノードは自身の successor と predecessor へのリンクをそれぞれ保持する．各ノードは successor のキーから自身のキーまでの範囲のキーが担当区間となり，その範囲に相当するデータを保持する．そして，各ノードはネットワーク上を時計回りの一方通行でクエリやデータを転送していきノードの探索や情報取得を行う．それに加えて Chord ではノード探索時のホップ数を削減して効率化するため，前述のリンクに加え

†福井大学，University of Fukui

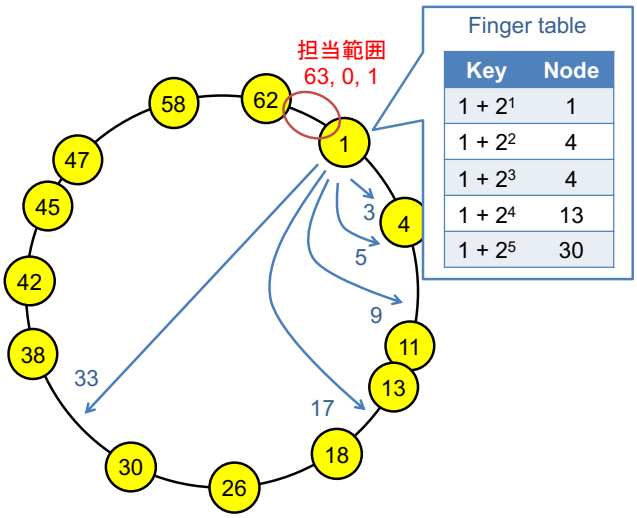

図 2：finger table の例

finger table と呼ばれるショートカットのリンクも各ノードが保持する．これは，キー空間全体の長さが m ビットかつ各ノードの配置された ID が重複しない場合，2^1, 2^2, 2^3, …, 2^{m-1} ビット先のキーを担当するノードへのリンクを保持するものである．finger table の例を図 2 に示す．図 2 ではキー空間の長さは 2^6 ビット（キーの範囲は 0～63）で，黄色の円はノード，そのなかの数字がノードのキーを表している．右上の表はキー1 のノードにおける finger table を，青色の矢印と数字は figner table のリンクを可視化したものである．また，キー1 のノードの predecessor はキー62 なので，キー1 のノードの担当範囲はキーが 63, 0, 1 のデータとなる．Chord では，finger table によって各ノードのルーティングテーブルの大きさを最大で m+1 程度に抑えつつ遠方のノードを参照する際でも常に経路を半分程度に短縮できるようになる．よって，N をノード数とすると O(logN)のホップ数でノードにアクセスできるようになっている．

Chord はノードの担当範囲を工夫したり検索機能によってデータの分散管理にも利用することができる．また，文献[4]では空間充填曲線である Z-ordering を用いて二次元の位置情報を一次元のキーに変換することによって二次元の範囲検索（range query）にも対応している．一方で，本研究の想定している間隔クエリは指定の間隔を指定した複数データへの検索となり，さらにその指定される間隔も年，月，週，日，時間など複数が考えられる．しかし，既存の手法では単一のキー間隔のクエリや独立した多次元のクエリを効率的に処理するオーバレイネットワークを構築できても，一次元でかつ複数のキー間隔のクエリを効率よく処理することはできない．

3. 提案手法

筆者は現在，複数の時間間隔を持つ間隔クエリを効率的に処理可能なオーバレイ構築手法を提案している[15-17]．本章では，これまで提案した手法について説明した後で改良点としてノードの仮想化について述べる．

3.1 アイデア

提案手法では既存の Chord[3]と同様の環状のオーバレイネットワークを構成し，一次元の時間情報に基づいてキー空間上にノードを配置する．ここでの時間情報は協定世界時や UNIX 時間のようなある基準時刻と年，月，日，時，分，秒などの単位を示すキーのビット値，さらにキー空間のビット長で表されるものとする．このとき，既存の Chord では finger table の指し示すリンク先は 2 のべき乗ビット先（2, 4 ,8, 16…）となる．一方で，提案手法ではキーが示す最小の時間単位を基に，1 週間先，1 か月先，1 年先といった間隔クエリを想定した間隔で担当ノードへのショートカットリンクを構築する．さらに，例えば年基準の場合では 1 か月先，3 か月先，6 か月先，12 か月先（一年先）のように同じレベルの時間単位の中でもホップ数を削減できる宛先ノードへ仮想リンクを構築する．本研究では，年，月，週，日，時刻を間隔として考慮し，表 1 に示すショートカットリンクを構築するものとする．キーの値を時刻（1 時間単位）としており，キー空間の長さは m ビットの年として表している．なお，分や秒といった基準でも同様にキー空間は定義できる．

表 1：構築ショートカットの種類と数

種類	対象キー（間隔）	数
年	2^0 年, 2^1 年, 2^2 年, ..., 2^{m-1} 年	m
月	6 か月, 3 か月, 1 か月	3
日	15 日, 7 日（曜日）, 3 日, 1 日	4
時刻	12 時間, 6 時間, 3 時間, 1 時間	4
隣接	successor, predecessor	2

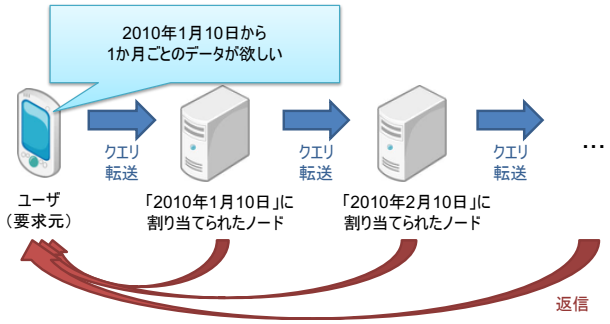

図 3：想定する間隔クエリの転送

図 3 は想定される間隔クエリの処理の流れを示すものである．この図はネットワークの一部を切り出したもので環状のトポロジは省略している．クエリは関連するノードへリンクを経由して再帰的に転送される．そして，クエリを受信したノードは該当するデータを要求元へ転送する．

3.2 ノードの仮想化

3.1 節で述べた基本的なアイデアにより，実際にユーザが要求しうる特定の時間間隔のクエリの処理に必要なホップ数を削減し，同時にルーティングテーブルのサイズも削減できることが確認されている[17]．

一方で，既存の Chord に比べて特定のノードに負荷が集中しやすいという課題が存在していた．Chord を含めた既存のオーバーレイネットワーク構築手法は分散ハッシュテーブル（Distributed Hash Table, DHT）の仕組みを利用している．これは，ネットワークにデータを格納する際にデータを特定するためのキーとなる値（本研究であれば時間情報）をそのまま利用するのではなく，SHA-1 等のハッシュ

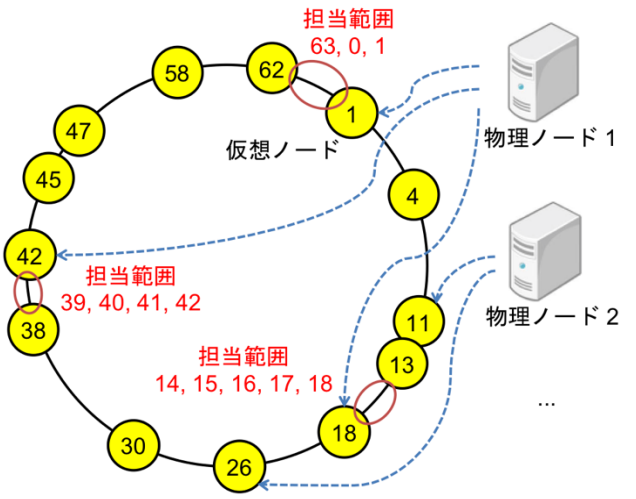

図 4：ノード仮想化の例

関数に通してからその値を ID として対応するノードへ格納する．そして，データを取得する際にも検索対象のキーをハッシュ関数に通して，得られた値の担当ノードへクエリを送信する．この方法を利用することにより，入力されるデータのキーの値の傾向に関係なくネットワークのノード全体へ均等に負荷を分散させることが可能となる[3]．一方で本研究では間隔クエリによる範囲検索を可能にするためキーは時間情報をそのまま利用しネットワークに格納している．そのため，必ずしもノードによって持つデータ数が分散するとは限らず，特定の時間帯のデータ数が多いような状態になるとその担当ノードに負荷が集中する状態が発生する可能性がある．この状態では本来オーバレイネットワークが持つ利点である負荷分散を発揮することはできない．

この問題を解決するため仮想ノード（virtual nodes）と呼ぶ仕組みを導入する．これは，一台の物理的なノードが複数の仮想的なノードを持ってそれらをオーバレイネットワーク上に配置するものである．一つの物理ノードが担当する範囲が仮想ノードによって複数の範囲に広がるため，一つのデータがネットワークの負荷に与える影響は分散されて小さなものになる．仮想ノードの総数が十分なものであり，かつ全ての物理ノードに均等に仮想ノードが割り当てられていれば確率的に物理ノードの負荷は均等になる．さらに，物理ノードの負荷状況に応じて動的に仮想ノードの割り当てを変更すれば，より負荷が集中しない適応性の高い割り当てが実現できる．本研究では，文献[17]に示された複数の時間間隔を扱うオーバレイネットワークを拡張する形で物理ノードの負荷を均一化することに主眼をおいてノード仮想化を導入した．

図 4 はノード仮想化の例を示したものである．黄色の円が仮想ノード，その中の数字がノードのキーを表しており，仮想ノードを通常のノードと見なして通常通りの Chord のネットワークが構築されている．水色の矢印が物理ノードと仮想ノードの対応を表していて，例えば右の列に示した物理ノード 1 はキーが 1, 8, 42 の 3 つの仮想ノードが割り当てられており，63, 0, 1 と 14, …, 18 と 39, …, 42 の 3 つの範囲のデータを担当することになる．同様に物理ノード 2 もキーが 11, 26 の 2 つの仮想ノードを持ち 2 つの範囲を担当している．

表 2：シミュレーションにおけるパラメータ

パラメータ	値
キー空間の最小単位	1 時間
キー空間の長さ	16 ビット（2^{16}=65536 時間）
物理ノード数	250, 500, 750, 1000
1 物理ノードあたりの仮想ノード数	2, 4, 8, 16
各ノードのキー	ランダムに割り当て
比較対象	仮想ノードを用いない状態
データ数	10000
データのパターン	一様分布・正規分布・Zipf 分布
クエリのパターン	52 キー（168 時間間隔）
試行回数	20
評価指標	fairness index(FI) 平均メッセージ数

4. 評価

本研究では，提案手法をシミュレーションして評価した．各手法は Java により実装した．シミュレーションにおけるパラメータを表 2 に示す．

4.1 シミュレーション環境

本シミュレーションはキーの最小間隔を 1 時間とし，キー空間全体の長さを 16 ビット，つまり 65536 時間とした．物理ノード数は 250, 500, 750, 1000 の 4 つの値を取り，さらに各物理ノードは 2, 4, 8, 16 個のいずれかの仮想ノードを割り当てられる．このとき，全ての物理ノードが同じ数の仮想ノードを持つ．仮想ノードにはランダムなキーを割り当て，お互いに文献[17]で示された方法でショートカットリンクを構築する．また，比較手法として仮想ノードを用いない，すなわち全ての物理ノードが 1 つの仮想ノードしか持たないネットワークを構築した．ネットワークに入力するデータの数は 10000 個とした．データのキー（時刻）はキー空間全体に一様にランダム分布しているパターンと，平均が 32768（キー空間の中央）で標準偏差が 6553.6（キー空間の大きさの 1/10）の正規分布に従ったパターン，キー空間全体で Zipf 分布[19]に従ったパターンの 3 種類を用いた．クエリは 52 個のキーを含み，キー空間上のランダムな位置から 168 時間（一週間）間隔で指定する．長さの合計は 8736 時間＝52 週＝1 年となり，「1 週間おきに 1 年間のデータを取得する」という間隔クエリとなる．シミュレーションは各条件で 20 回行い，仮想ノードへの負荷を測定するためクエリの処理に必要なメッセージ数を利用する．また，それとは別に物理ノードの負荷分散を評価するための公平性の指標として Fairness Index[18]を用い，以下の式で算出する．

$$FI = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \qquad (1)$$

ただし，x_i は i 番目の物理ノードに割り当てられたデータの数である．FI 値は 0 から 1 までの値を取り，FI=1 のときに $x_1 = x_2 = \ldots = x_i$ となり，全ての物理ノードに均等に仮想ノードが割り当てられている．よって，FI 値が 1 に近いほど公平であり均等に物理ノードに負荷が分散されていると言える．

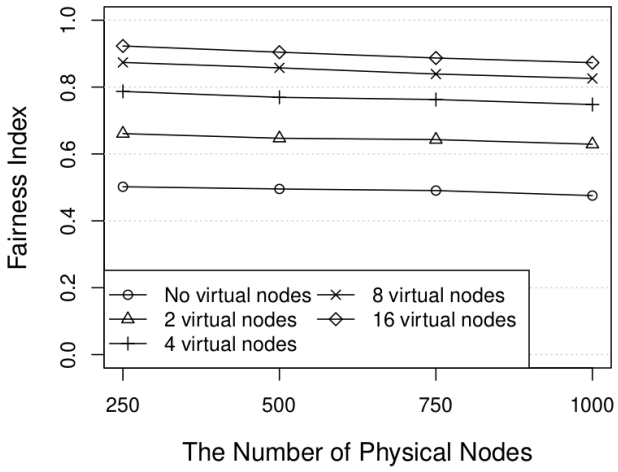

図 5：ランダム分布における fairness index

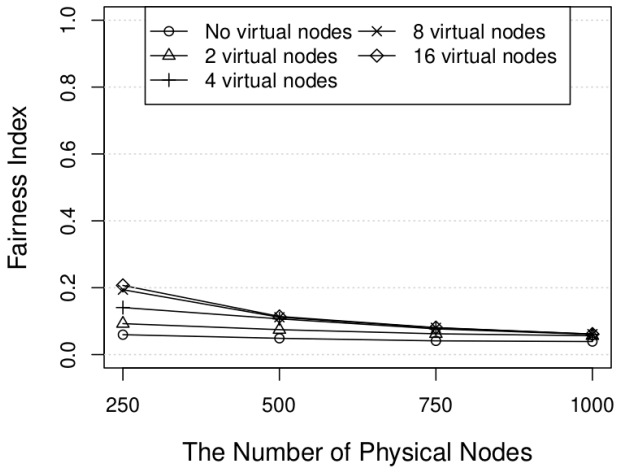

図 7：Zipf 分布における fairness index

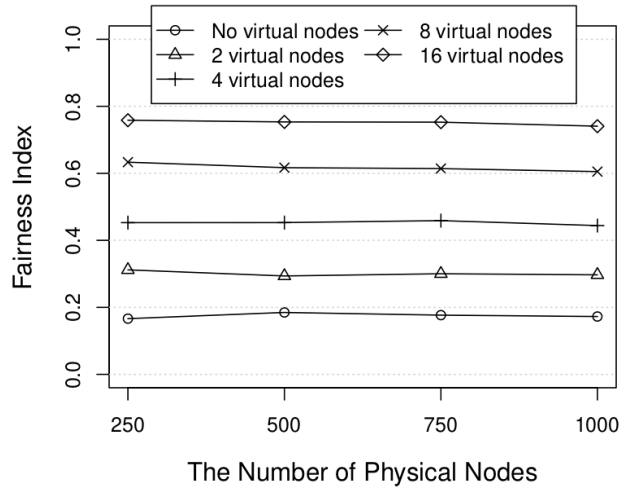

図 6：正規分布における fairness index

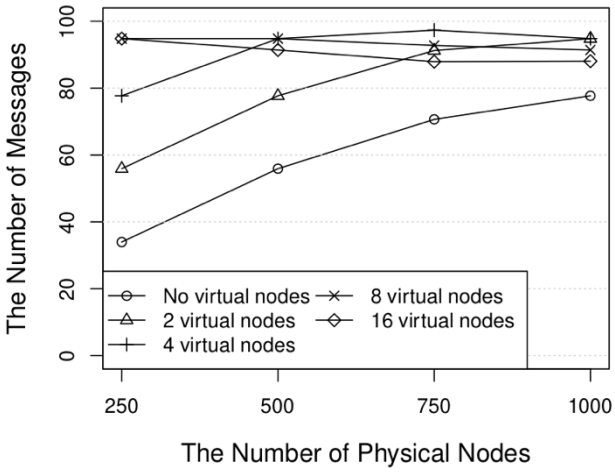

図 8：間隔クエリ 1 つ毎の仮想ノード間メッセージ数

4.2 負荷分散に関する結果

シミュレーションによる FI の値を図 5 から図 7 に示す．図 5 から図 7 はそれぞれ，キーをランダム分布，正規分布，Zipf 分布としたときの平均値である．各グラフの横軸は物理ノード数を示している．

図 5 から図 7 より，いずれの結果でも仮想ノードが多いほど FI が 1 に近くなっており，仮想ノードを設置しない場合と比べて FI が高くなっている．特に，図 6 ではすべてのノード数において同じ仮想ノード数でも FI が図 5 よりも低くなっているが，仮想ノードの増加による増加量が図 5 よりも大きくなっている．これらの結果から，仮想ノードを増やすことで物理ノードに対するデータ割り当ての不均衡を軽減できているといえる．また，図 5 では物理ノード数が増加すると同じ仮想ノード数でもわずかに FI が減少している．図 6 では物理ノード数の増加に伴い FI 値が減少していく現象は図 5 ほどはっきりとは起こっていない．一方，キーの偏りが大きい図 7 では FI がほかの結果よりも大きく低下し，物理ノード数が多い環境では，仮想ノードが多くとも FI が低い．これは特定のキーを担当している物理ノードに負荷が集中し，本シミュレーションでの仮想ノードの最大数 16 でも負荷分散には不十分であったためと思われる．

4.3 処理負荷に関する結果

間隔クエリ 1 つの処理に対して仮想ノード間でいくつのメッセージがやり取りされたかの平均を図 8 に示す．物理ノード数の増加や同じ物理ノード数でも仮想ノード数の増加によって平均メッセージ数は増大している．これは，どちらの場合にもネットワーク上に存在するノードの数が多くなってネットワークの規模が大きくなっていくためである．一方で，これらのメッセージ数は物理ノード単位，もしくは仮想ノード単位でのルーティングを最適化することで削減可能なものである．

5. 関連研究

Chord などの DHT 以外にも，多くのオーバレイネットワーク構築手法が研究されている[3-14]．スキップグラフとその関連手法はスキップリストを P2P モデルへ適用し，ノード（ピア）間に階層的なリンクを構築する[5-9]．スキップグラフはキーの始点と終点を指定する範囲クエリ（range query）に対応し，クエリはその範囲内か終点を超えないキーのノードへ再帰的に転送される．キーに基づくノード探索のホップ数は $O(\log N)$で表され，N はノード数である．また，各ピアのルーティングテーブルの長さは平均で $\log N$

である．一次元の範囲クエリのみでなく，多次元のクエリを扱うオーバレイネットワーク構築手法も研究されている[10-14]．それらの手法は 2D/3D の地図上の位置や地域に関係するデータの分散管理にも用いられ，「地理的オーバレイネットワーク（geographical overlay network）」とも呼ばれる．地理的オーバレイネットワークの多くは R 木や 4 分木，ドロネー図（ボロノイ図）などの幾何学の要素に基づいている．しかし，これらの既存手法は「間隔クエリ」を想定しておらず，本論文の提案手法と比べて効率的にクエリを転送できない．

オーバレイネットワークにおいて，特定の物理ノードへの負荷の偏りを解消するために仮想ノードを用いる既存手法として，文献[20, 21]が提案されている．文献[20, 21]では，短時間に特定のサービスへのリクエストが集中するフラッシュクラウド（flash crowd）[22]などを想定し，負荷が集中したノードが空きノードに助けを求める．助けを求められたノードは余裕があれば，負荷の集中したノードと同じキーを持つ仮想ノードを起動し，その仮想ノードをオーバレイネットワークへ参加させる．これにより，負荷が集中しているノードだけでなく，その近隣のキーへのリクエストによる負荷も分散させることができる．3 章の提案手法の例と 4 章の評価では仮想ノードのキーはランダムに割り当てているが，同様のアプローチを本論文の提案手法においても適用できる．

6. まとめ

本研究では，特定の複数の時間間隔に基づくクエリ（間隔クエリ）を効率的に扱うことが可能なオーバレイネットワークで負荷分散性能を向上させるための手法の検討を行った．Chord と同様の環状ネットワーク上で特定のノードへの負荷の偏りを避けるため，一台の物理ノードが複数の仮想ノードを担当し，その仮想ノードによってネットワークを構築するノード仮想化を行った．その結果，仮想化によりすべての物理ノードが持つデータの数が均一化されて公平性が向上することがシミュレーションによって確かめられた．一方で，仮想ノード間のメッセージ数は削減できうることも示された．

今後の課題として，クエリを処理する際のネットワーク上を流れるメッセージ数を削減するために物理ノード単位，あるいは仮想ノード単位でルーティングアルゴリズムを最適化する仕組みの導入を検討している．

謝辞

本研究の一部は G-7 奨学財団研究開発助成事業による成果である．

参考文献

[1] S. Hodges, S. Taylor, N. Villar, J. Scott, D. Bial, and P. T. Fischer: Prototyping Connected Devices for the Internet of Things, IEEE Computer, Vol. 46, No. 2, pp. 26-34 (2013).
[2] 第 6 章 2030 年の未来像 - ICT が創る未来のまち・ひと・しごと, 平成 27 年版 情報通信白書 第 2 部, 総務省, pp. 326-334, 2015. http://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h27/ (accessed July 17, 2020).
[3] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan: Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications, IEEE/ACM Transactions on Networking, Vol. 11, No. 1, pp. 17-32 (2003).
[4] S. Matsuura, K. Fujikawa, H. Sunahara: Mill: A Geographical Location Oriented Overlay Network Managing Data of Ubiquitous Sensors, IEICE Transactions on Communications, Vol. E90-B, No. 10, pp. 2720-2728 (2007).
[5] J. Aspnes and G. Shah: Skip Graphs, ACM Transactions on Algorithms (TALG), Vol. 3, No. 4 (37), pp. 1-25 (2007).
[6] A. González-Beltrán, P. Milligan, and P. Sage: Range Queries over Skip Tree Graphs, Computer Communications, Vol. 31, No. 2, pp. 358-374 (2008).
[7] S. Alaei, M. Ghodsi, and M. Toossi: SkipTree: A New Scalable Distributed Data Structure on Multidimensional Data Supporting Range-Queries, Computer Communications, Vol. 33, No. 1, pp. 73-82 (2010).
[8] S. Takeuchi, J. Shinomiya, T. Shiraki, Y. Ishi, Y. Teranishi, M. Yoshida, and S. Shimojo: A Large Scale Key-Value Store Based on Range-Key Skip Graph and Its Applications, Proc. of the 15th International Conference on Database Systems for Advanced Applications (DASFAA 2010), Vol. Part II, pp. 432- 435 (2010).
[9] R. Banno, T. Fujino, S. Takeuchi, and M. Takemoto: SFB: A Scalable Method for Handling Range Queries on Skip Graphs, IEICE Communications Express, Vol. 4, No. 1, pp. 14-19 (2015).
[10] A. Mondal, Y. Lifu, and M. Kitsuregawa: P2PR-Tree: An R-Tree-Based Spatial Index for Peer-to-Peer Environments, Proc. of the International Workshop on Peer-to-Peer Computing and Databases in Conj. with the 9th International Conference on Extending Database Technology (EDBT 2004), pp. 516-525 (2004).
[11] 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. SIG18(TOD28), pp. 1-15 (2005).
[12] E. Tanin, A. Harwood, and H. Samet: Using a Distributed Quadtree Index in Peer-to-Peer Networks, The International Journal on Very Large Data Bases (VLDB), Vol. 16, No. 2, pp. 165-178 (2007).
[13] M. Ohnishi, M. Inoue, and H. Harai: Incremental Distributed Construction Method of Delaunay Overlay Network on Detour Overlay Paths, Journal of Information Processing (JIP), Vol. 21, No. 2, pp. 216-224 (2013).
[14] Y. Teranishi, Susumu Takeuchi, and Kaname Harumoto: HDOV: An Overlay Network for Wide Area Spatial Data Collection, Proc. of the 26th ACM Symposium on Applied Computing (SAC 2011), pp. 506-513 (2011).
[15] 川上朋也: 複数の異なる時間間隔に基づく構造化オーバレイネットワーク構築手法の検討, 2018 年度情報処理学会関西支部 支部大会, G-22, pp. 1-3 (2018).
[16] T. Kawakami: A Construction Method for Structured Overlay Networks Based on Multiple Different Time Intervals, Proc. of the 2019 World Congress on Information Technology Applications and Services (World IT Congress 2019 Jeju), pp. 81-86 (2019).
[17] 川上朋也: 複数の異なる時間間隔に基づく構造化オーバレイネットワーク構築手法の評価, 2019 年度情報処理学会関西支部 支部大会, C-06, pp. 1-6 (2019).
[18] R. Jain, D.-M. Chiu, and W. Hawe: A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems, DEC Research Report TR-301 (1984).

[19] 山下高生, 栗田弘之, 高田直樹, 多様なデータサイズ分布を持つ Zipf 分布型処理要求に対する負荷分散とインメモリデータ・サイズ近似的最小化, 情報処理学会論文誌, Vol. 58, No. 12, pp. 1977-1992 (2017).
[20] X. Shao, M. Jibiki, Y. Teranishi, and N. Nishinaga: A Virtual Replica Node-Based Flash Crowds Alleviation Method for Sensor Overlay Networks, Journal of Network and Computer Applications, Vol. 75, pp. 374-384 (2016).
[21] X. Shao, M. Jibiki, Y. Teranishi, and N. Nishinaga: An Efficient Load-Balancing Mechanism for Heterogeneous Range-Queriable Cloud Storage, Future Generation Computer Systems, Vol. 78, Part 3, pp. 920-930 (2018).
[22] S. Fujita: Flash Crowd Absorber for P2P Video Streaming, IEICE Transactions on Information and Systems, Vol. E102.D, No. 2, pp. 261-268 (2019).