時間と位置を考慮した オーバーレイネットワークの提案

洞井晋一　松浦知史　藤川和利　砂原秀樹

奈良先端科学技術大学院大学 情報科学研究科

概要

センサ機器の発達と低価格化により、高精度なセンサデータを高密度で収集することが可能である。このようなセンサデータを広域ネットワークを介して共有することにより、環境問題への取り組みやビジネスへの応用などが期待できる。本研究ではセンサデータを効率よく利活用可能とすることを目的とし、時間や位置の連続性・局所性を考慮したオーバーレイネットワークを提案する。センサデータを扱うためには時間や位置などの連続量を扱え、かつ時間の周期性も考慮する必要がある。しかし、代表的なオーバーレイネットワークである DHT では、ハッシュ関数に基づいたデータ管理を行っているため完全一致検索しか行えず、時間や位置などの連続量の範囲検索を行う場合、時間・位置に応じて個別のクエリが大量に発生すると考えられる。本研究では時間に対してセンシングデータを集約することで高速な範囲検索や周期検索の行えるオーバーレイネットワークを提案する。また、提案手法を評価するために複数の計算機を用いたシミュレーションの可能なシミュレータを作成した。

Overlay Network Considering the Time and Location of Data Generation

Shin'ichi Doui, Satoshi Matsuura, Kazutoshi Fujikawa, Hideki Sunahara

Nara Institute of Science and Technology
Graduate School of Information Science

1　背景

現実世界における事象とは膨大な情報であり、その中には多くの有益な情報が含まれている。身近に挙げられるものとして大気中の情報がある。気温・湿度・気圧といった気象に関する情報や、二酸化炭素濃度や窒素酸化物濃度といった情報である。これらの情報は主に気象予報や環境問題対策に用いられてきたが、防災への利用や屋内の空調管理、初等教育への応用といった目的に対して有益な情報になり得る。また大気中の情報以外にも、自動車の位置・速度・方向といった情報も考えられる。自動車の情報を大規模に収集することで渋滞予測・回避といったことが実現できる。一般的にこうした現実世界に存在する情報は、人間が社会生活を営む上で有益な情報になり得る。

現実世界の情報はセンサ等を用いて取得することができる。大気中の情報は気象センサによって収集可能であり、自動車の位置などはGPSセンサを用いて収集可能である。また、こうしたセンサは高機能化・低価格化している。高機能化したセンサは様々な情報を収集することが可能であり、またネットワー

クへの接続性も期待できる。例えば Vaisala 社の高機能センサでは温度・湿度・気圧・風速・風向・雨量の情報に加えて CO_2 濃度の計測が可能である。風向・風速の計測は超音波のドップラー効果を利用しており短時間の微小な変化も捉えることが可能である。さらに、Echelon 社の iLon システムを用いてインターネットへの接続も可能にしている。このようなセンサは、大量生産による更なる低価格化が期待でき、従来よりも高密度なセンサの設置を期待できる。センサから生成される情報をインターネットを用いて共有することにより、現実世界の事象に含まれる情報を広く利活用できると期待できる。

本研究では Live E!プロジェクト [1] における教育利用を想定環境の一つとしている。Live E!プロジェクトでは公共サービス・教育・ビジネスに利用可能な環境情報の基盤構築を目指している。このプロジェクトの一部に、環境情報を教育に利用するため各小中学校に高機能センサを設置することが考えられている。これまで設置されていた百葉箱に代わり、高機能センサを設置することで科学的好奇心の高揚を促す目的がある。こうした取り組みではセンサの生成した情報を各小中学校に設置された計算機を用いて管理している。また、教育利用だけではなく、農業利用におけるセンシングデータや防災対策用のセンシングデータなども局所的に管理されると考えられる。センシングデータが取得された場所の情報である限り、こうした局所性は避けられない。しかし、環境情報を広く利用して多分野へ応用するためには、センサの生成した情報は広く共有されることが求められる。このように本研究では、センサから提供された情報が局所的に管理され、かつ広く共有されるべき環境を想定している。

環境情報の共有基盤を構築したとき、利用者による検索は、例えば「北緯 x 度，東経 y 度を中心に 1km 圏内の、T_1 時から T_2 時までの T_{span} 分間隔の温度情報」といったものが考えられる。これは、ある場所における今後の気温変動を過去の推移から予測する場合などが考えられる。センシングデータが複数の計算機によって局所的に管理されている場合、このような検索要求に答えるためにはその地域の全ての計算機に問い合わせる必要がある。

本研究ではこのような想定環境においてセンシングデータの効率の良い共有を目的とする。効率の良い共有とは、センシングデータの量やそれに対するユーザからのアクセスが負荷分散され、検索を行ったときに高速に情報の収集が可能な共有を指す。複数の計算機がそれぞれ局所的にセンシングデータの管理をしながら広域で共有するために、オーバーレイネットワーク技術を用いた情報の共有を行う。オーバーレイネットワークは既存のネットワーク上に仮想的なネットワークを構築し、耐故障性の向上や、スケーラビリティの確保、管理コストの低減、アプリケーションレベルでの可用性向上などを実現する技術である。本研究の想定環境では情報を生成し管理する主体が様々な場所に離散している。そこでこれらの主体を結ぶオーバーレイネットワークを構築し、情報の共有を行う。また、効率の良い共有を行うためにセンシングデータの生成された位置と時間を重要な要素として取り入れる。

本稿では 2 節において関連研究について述べ、要求事項をまとめる。3 節において提案手法を述べ、4 節ではシミュレータの作成について述べる。5 節においてシミュレータを用いて行った実験と評価について述べ、6 節においてまとめと今後の課題について述べる。

2　関連研究

センシングデータの効率の良い共有を目指すため、本研究ではオーバーレイネットワークを構築する。想定環境では数万台の計算機がオーバーレイネットワークの参加ノードとなる可能性がある。例えば小中学校に高機能センサを設置し、各学校に計算機を設置した場合を考える。小中学校の数は全国で約 3 万校であることから、3 万台のセンサと 3 万台の計算機が想定される。このセンサは計算機にセンシングデータを格納し、計算機がオーバーレイネットワークを構築することで全国での共有を行う。また、小

中学校以外にもオーバーレイネットワークに参加するノードが増加することを考え、ノードの増減に対応する規模拡張性を兼ね備える必要がある。また、ノードが増加してもデータの検索を高速に行える必要がある。

オーバーレイネットワークの構築手法には様々なものがある。特に代表的なものとして分散ハッシュテーブル (DHT) を用いたもの [2] がある。DHT を用いた手法では、データとノードからハッシュを用いて識別子（ID）を生成する。この ID を用いてノードへのデータの分配、利用者のデータの検索などを行う。しかし、DHT を用いた手法では完全一致の検索しか行うことができない。本研究で取り扱うようなセンシングデータの検索は、「北緯 x 度，東経 y 度を中心に 1km 圏内の、T_1 時から T_2 時までの T_{span} 分間隔の温度情報」といった検索である。これは「位置の範囲」「時間の範囲」「時間の間隔」を特定して検索を行う。完全一致の検索では特定の位置と時刻を指定した情報に特化してしまうため、範囲や周期の検索を行うことは検索速度の低下や、検索クエリーの増加につながる。本研究では完全一致の検索に加え、範囲を指定した検索と間隔を指定した周期検索を実現しなければならない。

連続量を扱うことができるオーバーレイネットワークの研究としては、SkipNet[3] や Mercury[4] などが挙げられる。これらの手法では連続量に対して範囲検索を行うことが可能であり、連続量として位置や時間を扱えば範囲検索が可能である。また、位置の局所性を考慮した LL-net[5] や Mill[6] といった手法もある。しかし、これらの手法を用いて時間の連続量を扱うことは難しい。まず、時間は無限に単調増加し続けるという特徴がある。位置のように限られた連続量に対しては固定の識別子を割り当てることで連続量として管理できる。しかし、無限に増加する時間に対して固定の識別子を割り当てることは難しい。また、時間には周期的に扱われるという性質もある。上述の検索例のように「T_{span} 分間隔の温度情報」といったようなものや、「毎朝 7 時の風向の情報」であったり、自動車の情報として「日曜日の渋滞状況」といったものが考えられる。こうした周期的な情報を検索する場合、時間を連続量として扱えば検索速度の低下が予想できる。また、現在からの時間差によって情報の利用方法も変化する。現在のセンシングデータ、つまりリアルタイムに取得されているセンシングデータは一つのセンシングデータとして利用されるのに対し、現在からの時間差が長いセンシングデータは、ある程度の範囲でまとめて利用される。現在からの時間差によって利用の方法が違うデータに対し、線形な連続量として扱うことは現実の利用において効率が悪いと考えられる。

要求事項をまとめると以下の通りとなる。

- 規模拡張性に優れたオーバーレイネットワークを構築する
- 位置の局所性を考慮する
- 無限に増加する時間を扱うことができる
- 時間に対して範囲検索・周期検索が行える
- 時刻差によるデータの利用方法の違いに対応する

本研究ではこれらの要求事項を踏まえ、既存の手法よりも高速に情報の検索が可能であり、負荷分散に優れたオーバーレイネットワークの構築手法を提案する。

3 提案手法

本節では本研究の提案手法について説明する。まず、3.1 において提案手法の概要を説明する。次に 3.2 において時間の取り扱いについて説明し、3.3 において三次元の ID 空間を分割する手法、3.4 において検索の手法を説明する。

3.1 概要

本研究の提案手法では CAN[7] に似た ID 空間を構築する。CAN では d-次元の直交座標空間を構築

し、オーバーレイネットワークの ID 空間として用いている。本研究では三次元の直交座標空間を構築し、オーバーレイネットワークの ID 空間として利用する。この三次元はそれぞれ、経度・緯度・時刻を表している。経度を X 軸、緯度を Y 軸、時刻を Z 軸と定め、空間内における任意の座標は (x, y, z) によって表される。この座標空間にノードを割り当て、お互いに空間を分割保持する。ノードは現実世界の経度・緯度を持っており、これを (x, y) とする。現実世界において近い位置のノードはオーバーレイネットワーク上の ID が近い値になり、お互いに協調しながらデータの共有を行う。

ノードはオーバーレイネットワークに参加する場合、まず参加する座標を決定する。オーバーレイネットワークに参加している一つののノードに対して、参加する座標を管理しているノードの検索を依頼する。そして、座標を管理しているノードに対して参加の要求を送信する。参加の要求を受け付けたノードは自分の保持している空間の半分を参加するノードへと割り当てる。こうしてノードは ID 空間における位置 (x, y, z) と、保持する空間を割り当てられる。

センシングデータは取得された経度・緯度・時刻を基準に (x, y, z) を決定する。ノードは管理している ID 空間内のセンシングデータを保持する。また、Chord や SkipNet と同様に、ノードの座標から 2 の累乗離れた地点への到達性を持つことで任意の座標まで $O(\log N)$ の検索回数で到達可能である。

本研究ではこの ID 空間に加えてセンシングデータの集約手法を提案する。センシングデータは取得された時間について集約が行われ、検索速度の向上を図る。また、集約を 2 の累乗ごとに行うことで範囲検索とともに周期検索の高速化を行う。

3.2 時間軸の取り扱い

一般的なオーバーレイネットワークでは、ノードとデータに対して ID を割り当て、ID を用いてデータの検索やノードの発見を行う。データの発生時刻を時間軸の ID と考えることで、範囲検索や周期検索

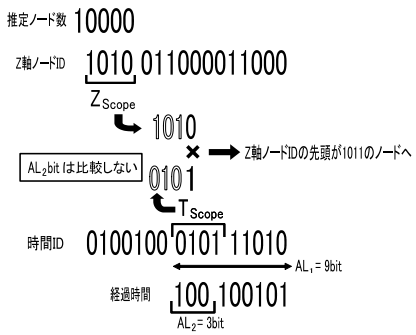

図 1: センシングデータ保持判別の一例

は可能である。しかし、時間軸を経度や緯度と同様の連続量として取り扱うと様々な不都合が生じる。例として 64 ビットの ID 空間を時間軸として取り扱うことを考える。64 ビットの空間では 0 から約 1.8×10^{19} までの数を取り扱うことができる。この数字を秒数として考えると 0 年から約 5800 億年までの時間を取り扱える。しかし、現実にセンシングデータが取得されるのはこの空間の一部であるため、大半の ID は使用されない ID になってしまう。オーバーレイネットワークで用いられている ID に対してノードを割り当てるという手法を用いれば、大半のノードがセンシングデータを保持しないことになってしまうため、負荷分散の点で好ましくない。逆に ID の空間を 0 年から 10 年という形で狭く定めると、ID の不足ということも考えられる。

本研究ではこうした問題を解決するために以下のような段階的な集約を行う。まず、ノードの個々の ID を決定し、次にデータの ID を決定する。この二つの ID を比較して一致した場合はデータを保持し、一致しない場合は一致するノードへとデータを譲渡する。データの ID の決定方法には、データ発生時刻からの経過時間などを用いて決定し、経過時間が長くなるほど集約の母数が大きくなる。具体的には以下のようにして ID を決定していく。また、ノードがセンシングデータを保持するかどうかを判別する例を図 1 に示す。

ノードの ID を決定　ノードは割り当てられた領域の Z 軸の大きさから Z 軸方向に存在するノードの数を推定する。このノードの推定数は、各ノードの領域が 2 分の 1 ずつ分け与えられるため、2 の累乗の数になる。このノードの推定数の対数を取り N_z とする。ノードの推定数は 2^{N_z} となる。ノードは Z 軸上の座標を 2 進値で表し、先頭 N_z ビット分を有効な Z 軸 ID として定める。これを Z_{Scope} とする。図 1 では推定ノード数が 16 ノードであるため、$N_z = 4$ となる。よって、Z 軸座標の先頭 4 ビットを取り、$Z_{Scope} = 1010$ となる

集約レベルの決定　センシングデータをどのノードに格納するかは集約レベルによって異なる。集約レベルはセンシングデータの生成された時刻 T との差によって段階的に増えていく。T から集約が行われる時刻までの差を T_{Span} とする。この T_{Span} を 2 進値で表したときの、最上位ビットまでのビット数を集約レベル 1（以下、AL_1）とする。AL_1 はセンシングデータが何段階目の集約レベルにあるかを示しており、時間が経過するに従い AL_1 は増加する。また、T_{Span} の最上位ビットから次に 1 となっているビットまでの距離を集約レベル 2（以下、AL_2）とする。AL_2 は AL_1 が増加するまでの残り時間を表している。図 1 では $AL_1 = 9$ ビット,$AL_2 = 3$ ビット と定まる。

データの ID を決定　センシングデータは生成時刻の下位 AL_1 ビットのうち、上位 N_z ビットを有効な ID として定める。これを T_{Scope} とする。AL_1 の値が大きくなれば T_{Scope} が左側へとシフトし、より大きな範囲での集約が行われる。図 1 では下位 9 ビットのうち、先頭の 4 ビットを抜き出して 0101 がデータの ID となる。

ID の比較　ノードがセンシングデータを保持するかどうかは Z_{Scope} と T_{Scope} の比較によって決定する。このとき、二つの ID の上位 AL_2 ビットを除いて比較を行う。比較を行う二つの ID が一致しなかった場合は次のノードへと渡される。次のノードの ID は、Z 軸座標の先頭を T_{Scope} の上位 AL_2 ビットを除いた ID に置換することで得られる。AL_2 が上がる度にセンシングデータは移動が行われ、AL_2 が最大値になったノード、すなわち Z_{Scope} と T_{Scope} が完全一致するノードにおいてセンシングデータは一番長く保持される。図 1 では下位 1 ビットの比較となるが、一致しないために次のノードへと渡される。次のノードの ID は Z 軸座標の先頭 4 ビットを置換し、1011 を先頭とした ID にデータを送信する。

3.3　三次元 ID 空間の分割

本節では三次元の ID 空間をノードによって分割する方法を示す。本研究で用いている空間の分割方法は CAN と似た手法を用いている。しかし、本研究では位置の情報である (x, y) が固定の値に定められているため、座標の配置によっては領域の半分を分け与えることができない場合がある。こうした場合に本研究では時間を表す Z 軸の座標を変更することや、実体を伴わないダミーノードを配置することによって解決を行う。基本的な空間分割のアルゴリズムは以下の通りである。

空間分割のアルゴリズム

1 新規ノードは (x, y, z) を定め、その座標を管理しているノードに対して参加の要求を行う

2 参加の要求を受けたノードは X 軸または Y 軸による分割を試みる

3 分割後の領域が条件 a,b （下記に示す）を満たさない場合は z 軸による分割を行う

4 条件 b を満たし、条件 a を満たさない場合は Z 軸における座標の移動を行う

5 Z 軸による分割が条件 b を満たさない場合はダミーノードの作成を行い、手順 1 からやり直す

分割条件

a 一つの領域には一つのノード

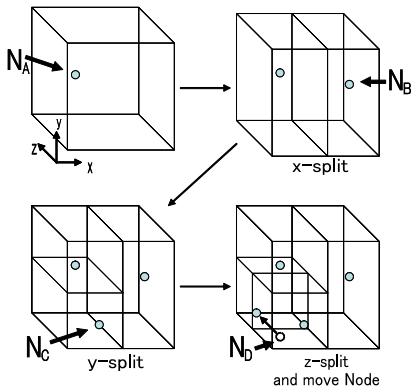

図 2: 空間分割

b 領域の一辺は他の辺の半分より長い

実際に分割が行われていく例を図 2 に示す。図 2 ではまずノード N_A が ID 空間に存在している。そこに N_B が参加の要求を行い、X 軸による分割が行われる。次に N_C が N_A に対して参加の要求を行う。X 軸による分割も可能であるが、条件 b を満たさないため y 軸による分割を行う。最後に N_D が N_C に参加の要求を行う。X 軸による分割も Y 軸による分割も条件 b を満たさないため、Z 軸による分割が行われる。N_D の座標がそのままでは a を満たさないため、N_D の Z 軸における座標を変更する。

ダミーノード　上記のアルゴリズムで空間を分割した場合、Z 軸の分割においても条件 a,b を満たさない場合がある。こうした場合、参加の要求を受けたノードはダミーノードを作成する。ダミーノードは x 軸または y 軸に分割可能な座標に割り当てられ、通常のノードと同じように空間の半分を保有する。ダミーノードによって空間を分割した状態で再度新規ノードに領域の割り当てを試みる。この場合も領域の分割を行えない場合は更にダミーノードを作成し、領域の割り当てを繰り返す。ダミーノードが 2 つ作成されると条件 b を必ず満たすため、領域の割り当てが可能になる。ダミーノードは通常のノードと同様に割り当てられた ID 空間を管理するが、以下の部分が通常のノードとは異なっている。

- ダミーノードは作成したノードのコンピュータ資源を利用する
- ダミーノードに参加要求が行われた場合はダミーノードの全領域を新規ノードに割り当て、ダミーノードは消滅する

3.4　センシングデータの検索手法

本節ではセンシングデータを位置と時間を用いて検索する手法を述べる。ID 空間内における任意の座標を検索する手法として、CAN に用いられているような隣接したノードを検索する手法と、Chord に用いられているような 2 の累乗の位置に対して経路制御表を持つ手法を用いる。これらの手法を用いることにより、ノード数 N に対して任意の座標の検索を $O(\log N)$ の回数で行うことができる。このことから、集約されたセンシングデータが存在する座標の位置を特定できれば、任意のデータに対して高速な検索が可能になる。センシングデータが生成された経度・緯度は時間による集約が行われても変化することがないため、検索時にはセンシングデータが配置されている時間軸上の座標のみを特定すればよい。

本研究ではセンシングデータを検索する手法として以下の三種類の検索を提供する。

- 時間を指定したデータの検索（単一検索）
- 時間の範囲を指定したデータの検索（範囲検索）
- 時間の間隔を指定したデータの検索（周期検索）

これらの三種類の検索について、検索対象となるセンシングデータがどの座標に存在しているかを特定する。

単一検索　単一検索では特定の時刻 T のセンシングデータを検索する。まず、オーバーレイネットワークに参加しているノード A に対し、単一検索の要求と時刻 T を伝える。ノード A はまず該当するセンシン

グデータを持っていないかどうかを調べる。持っていなかった場合、センシングデータを集約する手法を用いて、T から次のノードの座標 B を得る。このとき、AL_2 による ID の作成は行わず、AL_1 を AL_1-1 とし Z 軸座標の先頭に T_{Scope} を置換することで次のノードの座標 B を得る。この座標を管理しているノードを検索し、発見したノードに対して単一検索の要求を行う。発見したノードをノード B とする。

次にノード B は検索要求を受け取り、センシングデータを保持している場合はそれを返す。しかし、AL_2 が N_z よりも小さい値の場合は次の集約ノードにデータが移動しているため、その ID に対して検索要求を転送する。また、センシングデータの発生からあまり時間が経っていない場合は、まだ集約が行われずに複数のノードがセンシングデータを保持している可能性がある。このような場合、保持している可能性のあるノードに対して検索要求を転送する必要がある。センシングデータが発生した直後であれば、全ノードに対して要求を転送する必要がある。

範囲検索　時間の範囲を指定した検索では、時刻 T_S から時刻 T_E のセンシングデータを検索する。まず、T_S と T_E の時刻についてセンシングデータを保持しているノードを検索する。これは単一検索を用いる。T_S を保持しているノードをノード S、T_E を保持しているノードをノード E と呼ぶ。ノード S とノード E が同じノードである場合、T_S から T_E のセンシングデータは一つのノードに集約されているため、ノード S に問い合わせるだけで検索は終了する。ノード S とノード E が異なるノードの場合、T_S と T_E の中央値についてノードの検索を行う。発見したノードと保持しているノードを比較しながらさらに中央値について検索し、T_S から T_E の値を保持している全てのノードを探索し、センシングデータを得ることができる。

周期検索　時間の間隔を指定した検索では、検索を行う時間範囲 T_S と T_E を定め、検索の間隔である T_{Span} を定める。T_S と T_E で範囲検索を行えば T_{Span} 間隔のセンシングデータを得ることができる。しかし、T_{Span} の間隔が大きければ、該当するセンシングデータを持っていないノードに対しても検索を行ってしまう。そこで、まず T_S に対して単一検索を行う。T_S を保持しているノード S が $T_S - T_{Span}$ の時刻に対して検索を行い、T_E までの検索を行う。時間が経過するにつれてセンシングデータは集約されているため、検索対象のノードの数は減り全体の一部のノードに問い合わせるだけで検索が可能である。

4　シミュレータの作成

本節では提案手法を評価するために作成したシミュレータについて述べる。まず、4.1 においてシミュレータの概要と必要な機能について述べる。次に 4.2 においてノード間の通信部分、4.3 においてノードの非同期参加方法、4.4 においてセンシングデータの擬似的な発生方法、4.5 において三次元空間の可視化について詳細な部分を述べる。

4.1　概要

本研究では学内の個人常用端末を利用することを想定している。奈良先端科学技術大学院大学情報科学研究科には Sun Microsystems の Java Workstation W2100z が 40 台導入されている。この計算機を用いて 1 台あたり 1000 ノードをシミュレートし、合計 4 万ノードをシミュレートする。TCP/IP を用いて通信を行いながら全体が同一のオーバーレイネットワークに参加することで、実利用に近い環境でシミュレーションを行う。このような環境で実験を行うためにシミュレータには以下のような機能を実装した。

1. TCP/IP 通信によるノード間の通信

   複数の計算機間を TCP/IP で通信するために、ノード間の通信に TCP/IP を用いた。そのためノードは待ち受け用の TCP ポートを一つ保持し、同一計算機内の通信であっても TCP/IP を用いる。

2. 非同期のノード参加

   逐次的にノードを参加させた場合、多くのノードをシミュレートするには長い時間を要し、また効率も悪い。三次元 ID 空間において離れた位置への参加は同時に実行することが可能であるため、ノードが非同期に参加できるようにノードの状態管理を行った。

3. センシングデータの生成

   オーバーレイネットワークに参加したノードは、定期的にセンシングデータを生成する。このデータの内容はランダムな値が入っており、センシングデータとしての意味は無い。シミュレータでは 1 ノードにつき 1 つのスレッドがセンシングデータの生成を行う。

4. 三次元 ID 空間の可視化

   三次元空間の分割状況や生成データの集約状況などを確認するために、三次元 ID 空間の可視化を行った。可視化された三次元 ID 空間にはノードとデータを配置し、空間がどのように分割され、データがどのノードに保持されているかが確認できる。

以下に、それぞれの項目について詳細な手法を述べる。尚、実装には Java 1.5 を用いた。

4.2 ノード間の通信

本研究では他ノードの情報として経路制御表と近隣ノード表を保持している。経路制御表は三次元 ID 空間において、2 の累乗離れた座標のノード情報である。経路制御表は検索を高速に行うために利用される。近隣ノード表は三次元 ID 空間において隣接しているノードの情報である。経路制御表に格納しているノードの情報は確実ではないため、経路制御表では検索できない場合がある。その際に、近隣ノード表によって次のノードを求め、ノードへの到達性を確保する。

本研究ではノード間の通信に TCP/IP を用いているため、ノードは TCP の待ち受けポートを一つ所持している。ノードはオーバーレイネットワークに参加すると、待ち受けポートからの通信を待つスレッドを立ち上げる。このスレッドは他ノードから通信の要求があった場合に、そのノードとの通信用のスレッドを立ち上げる。元のスレッドは再び Listen の状態に移り、他のノードからの通信を待つ。他ノードとの通信用スレッドは、明示的に相手側から Close されない限りは TCP のコネクションを保持する。しかし、一度の検索で使用されただけでほとんど使用されないコネクションである場合、リソースを開放しておくことが望ましい。そこで、近隣ノード表に記載されていないノードに限り、一定時間が経過しても情報の送受信が無い場合はコネクションを切断し、スレッドを終了する。

4.3 非同期のノード参加

ノードが非同期に参加しても可能なように、オーバーレイネットワークに参加するノードは二つの状態の切り替えを行う。一つはどのような要求にも答えることが可能な状態であり、StableState と呼ぶ。もう一つは他からの要求には一切答えない状態であり、LockState と呼ぶ。新規参加ノードがオーバーレイネットワークに参加しているノードに対して参加の要求を行ったとき、要求を受けたノードは LockState へと状態を変更する。これは二つ以上のノードに対して同時に領域の割り当てを行うことが難しいためである。更に、要求を受けたノードは近隣ノード表に記載されているノードに対して LockState への要求を行う。これを受けたノードは状態が StableState であれば LockState に状態を変化する。これは領域の分割を行っているときに、周辺のノードが領域の分割を行った場合に不都合が生じるためである。この LockState への変更要求が拒否された場合は、直ちに周辺のノードを StableState に戻し、参加の要求を拒否する。近隣ノード表に記載された全ノードを LockState に変更することが出来た場合のみ、ノード参加の手続き

を進めていくことができる。参加を受け付けたノードは新規参加ノードに領域を割り当て、近隣ノード表を再構成する。これらの手続きが終了した後、近隣ノード表に記載されたノードの LockState を StableState へと変更し、最後に自分の状態を StableState に変更する。こうした状態の変更を行うことで非同期にノードがオーバーレイネットワークに参加しようとしても、正常に参加の処理を受け付けることができる。

4.4 センシングデータの生成

ノードは参加が終了するとセンシングデータ生成用のスレッドを作成する。このスレッドでは各ノード固有の時間でセンシングデータの生成を行う。生成されるデータは以下の要素を持っている。

- 生成時刻
- 生成したノードの IP および待ち受けポート番号
- 生成した経度・緯度
- 生成データ

これらの要素のうち、生成データにはランダムな値を格納している。生成した経度・緯度は生成したノードの X,Y 座標の位置を格納している。ノードはこのデータを生成するとひとまず保管ファイルに追加する。次に、保管ファイルに格納されているセンシングデータに対して集約を開始する。保持できないデータを発見した場合は、保持可能な Z 軸の座標を特定する。生成データに格納されている経度・緯度情報と、Z 軸の座標により、生成データを保持可能な三次元 ID 空間の座標を決定する。座標を管理するノードを検索によって特定し、センシングデータをそのノードへと送信する。尚、これらの集約を行っているときに他のノードからデータの送信があった場合、キャッシュファイルに格納し、次の集約のときに保管ファイルに追加する。

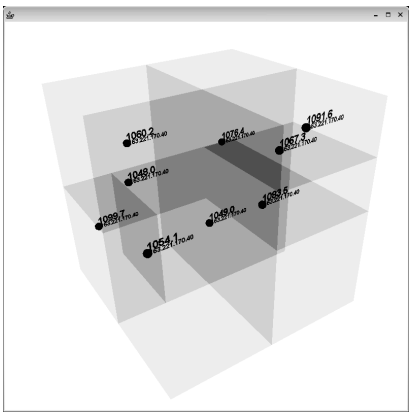

図 3: 三次元 ID 空間の可視化

4.5 三次元 ID 空間の可視化

本研究で提案する ID 空間を Java3D を用いて可視化した。ID 空間がノードによって分割されていく様子や、データが集約されていく様子などを確認することができる。時刻の範囲を指定すればその時間に発生したデータの座標の位置を確認することが可能であり、どのようにセンシングデータが集約されていくかを可視化している。シミュレータにより可視化している様子を図 3 に示す。

5 実験と評価

本節ではシミュレータを用いて行った実験およびその結果から、提案手法に対する評価について述べる。5.1 では評価実験について、5.2 では実験の結果について、5.3 で結果に対する考察について述べる。

5.1 評価実験

作成したシミュレータを用いて評価実験を行った。本研究の提案手法ではセンシングデータを集約し、生成された時刻に応じて様々なノードが保持する。集約先のノードは時刻によって変化するため、一つのノードが保持しているセンシングデータの量は時刻

によって大きく異なると考えられる。また、ハッシュを用いた手法では格納されるノードがランダムに選択される。一つのノードが保持しているセンシングデータの量は均一にならず、平均値から分散して分布すると考えられる。提案手法の分散がハッシュを用いた手法の分散に準ずるものであれば、提案手法がセンシングデータを分散配置させているといえる。

このようなセンシングデータの分散配置を評価するために、提案手法においてノードが保持するセンシングデータ量の標準偏差と、乱数を用いて格納した場合の標準偏差について比較を行った。まず三次元 ID 空間のオーバーレイネットワークを構築する。次にノードは定期的にセンシングデータを生成する。センシングデータは以下の情報を保持する。

- 生成された時刻
- 生成したノードの位置
- 乱数値

ノードは生成された時刻と生成したノードの位置を元に時間軸方向に集約を行う。乱数値はセンシングされた情報を表しており、実験上においては意味を成さない値とする。

提案手法ではアルゴリズムに従ってセンシングデータの集約が行われていくのに対し、乱数を用いた方法では格納先のノードを乱数から決定する。乱数値は Java に用意されている Random クラスを使い、線形合同法によって擬似乱数を生成する。格納先のノードはセンシングデータを受け取り、それを保管する。保管されたデータはそれ以上移動することはなく、最後までそのデータが保持される。このようにしてセンシングデータを次々と生成し、一定時間ごとにノードの保持しているセンシングデータの量を計測する。全ノードのこれらの情報を集め、平均値を出した上で標準偏差を計算する。

また、同様の実験をノードの位置に粗密が発生する場合においても行った。現実のセンサ配置を考えた場合、センサが設置されている位置には粗密が発生すると考えられる。本研究では空間の分割によってその粗密を補う。この手法の有効性を確認するために、ノードの配置に粗密を発生させ標準偏差の値を収集した。

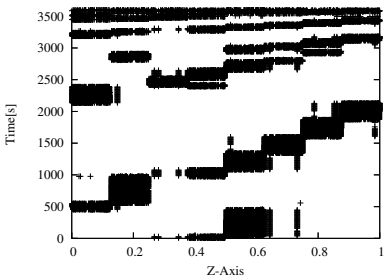

図 6: 保持データの分布

次に、センシングデータの集約を確認するためにセンシングデータの配置を確認した。シミュレータを動作から 1 時間後に停止し、各ノードが保持しているセンシングデータの時刻を確認し保持しているセンシングデータが Z 軸上において集約されているかを確認した。

5.2 実験結果

ノード数 1000、データの生成間隔 10 秒で 1 時間の計測を行った結果を図 4 に示す。また、中心点 $(0.5, 0.5)$ から半径 0.3 の円内に、円外のノードの 8 分の 1 を再配置し、同様の計測を行った。結果を図 5 に示す。最後に 1 時間後にシミュレータを停止し、各ノードが保持しているデータを集計した。横軸に Z 軸上の位置、縦軸に時刻を取り、結果を図 6 に示す。尚、保持データ量の分布は提案手法においてノードを分散配置した場合のものである。

5.3 考察

図 4、図 5 の両方において、乱数を用いてデータを配置した場合は線形に標準偏差の値が増加しているのに対し、提案手法では瞬間的に標準偏差の値が大きく増加していることがわかる。これは AL_1 の増

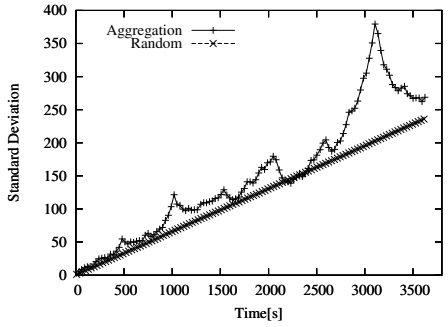

図 4: 保持データ量の標準偏差の変化（分散配置）

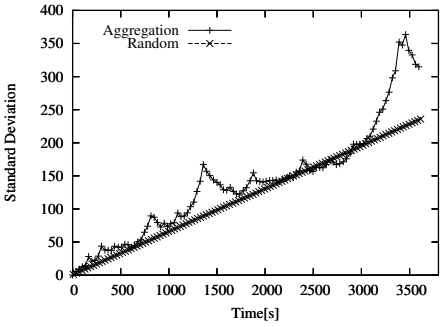

図 5: 保持データ量の標準偏差の変化（集中配置）

加による集約が行われたとき、これまで保持していたセンシングデータの多くを他のノードに移動させるためと考えられる。時刻 2700 から時刻 3500 にかけての値が大きく、この時間帯にセンシングデータの集約が行われたと考えられる。

また図 6 では、発生時刻が 3500 付近のセンシングデータ、つまりシミュレータの停止直前のデータはほぼ全ノードが保持しているのに比べ、発生時刻が過去のものになるにつれて保持しているノードの数が減っている。これは時刻によるセンシングデータの集約が正常に行われたことを示している。また、時刻 0 から 1000 のセンシングデータを保持しているノード数と、時刻 1000 から 2000 のセンシングデータを保持しているノード数では 2 倍近くの開きがある。そのため、時刻 1000 のデータを境に集約が行われたと考えることができる。

AL_1 の集約が行われるのは 2 の累乗ごとであるため、実験を行った時間では時刻 $512, 1024, 2048$ などが当てはまる。また、ノードの数が 1000 であることから、Z 軸に存在する推定ノード数は 10 ノードと計算される。そのため N_Z は 3 または 4 になると考えられる。AL_2 による集約は経過時間の上位 N_Z ビットが有効であるから、実際に集約が始まるのは $544, 576, 1088, 1152, 2176, 2306$ の時刻になる。これらの時刻は実験結果の標準偏差が増加を始めた時刻とほぼ一致するため、理論上と同様の集約が行われていることが確認できる。

提案手法と乱数による手法を比較すると、集約が行われるときに標準偏差の値が大きく開くが、時間経過と共にその差が小さくなっていることがわかる。集約が行われることによって各ノードの保持しているセンシングデータの量には差が生じるが、その分布は乱数を用いる手法に準ずるものであることがわかる。つまり、保持しているセンシングデータ量のノード間の公平性という点では、ハッシュ関数を用いる手法と比べても公平性を保っていると考えることができる。

また、提案手法ではセンシングデータを集約しながら保持しているため、受け取ったノードによる圧縮を行えることが期待できる。例えば時刻 500 から 1000 までのセンシングデータが全て同じだった場合、時刻の範囲とセンシングデータの内容、個数を記録するだけでよい。こうした圧縮は時刻 500 から 1000 までのセンシングデータが異なるノードに格納されている状況では行うことができない。また、センシングされる情報は時間経過によって劇的に変動することが少ないため、こうした圧縮はより有効になると期待できる。

6 まとめと今後の課題

ユビキタスネットワーク社会においてセンシングされる情報を共有することは大きな可能性を秘めている。センシングデータを共有するために、本研究

では時間と位置に基づいてオーバーレイネットワークを構築した。情報の集約を行うことでセンシングデータの高速な検索を行うと共に、ノードが保持するセンシングデータ量の公平性を確認することができた。今後の課題として以下のものが挙げられる。

他の手法との検索速度の比較　提案手法と他の手法との検索速度の比較を行う必要がある。他の手法としてはChordに代表されるようなDHTを用いた手法、SkipNetなどの範囲検索が可能な手法が挙げられる。DHTを用いた手法に対しては範囲検索・周期検索が高速になると考えられる。また、情報の集約を行っているため、範囲検索が可能な他の手法に対しても範囲検索・周期検索が高速になると考えられる。これらの比較を行うために、センシングデータに対する実際の検索パターンをモデル化し、提案手法がそのモデルに対して有効であることを示す必要がある。

集約にかかるコストの算出　提案手法では集約を行うことによって定常的にセンシングデータがネットワーク上を流通する。この定常的に流れているセンシングデータの量を集約にかかっているコストと考え、このコストを算出して現実に許容できる範囲を示す必要がある。センシングデータが増加すればコストが増加し、オーバーレイネットワークを輻輳する恐れがある。そのため、ある程度時間が経過したデータは集約を行わずにノードに滞留させたり、オーバーレイネットワーク上にスーパーノードを配置し、そこで全情報の集約を行うことなどが考えられる。

以上のようなことを複数計算機による大規模な実験によって評価する必要がある。想定される環境に合わせて参加ノードは3万ノードとし、より長時間の実験を行うことで、実利用に近い評価を取得していく。今後は実験によって得られる評価を考慮しながら、実利用に向けて更に研究を進めていく。

参考文献

[1] Live E! Project, http://www.live-e.org

[2] I.Stoica, R.Morris, D.Karger, M.Kaashoek, and H.Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *ACM of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 149-160,Jun 2001.

[3] N.Harvey, M.Jones, S.Saroiu, M.Theimer, and A.Wolman. Skipnet: A scalable overlay network with pratical locality properties. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, Mar 2003.

[4] A.Bharambe, M.Agrawal, and S.Seshan. Mercury: Supporting scalable multi-attribute range queries. In *Proceedings of the ACM SIGCOMM 2004 Technical Conference*, pages 353-366,Oct 2004.

[5] 金子雄，春本要，福村真哉，下條真司，西尾章治郎，“位置情報に基づくP2Pネットワークにおけるエリアの階層化”，電子情報通信学会第16回データ工学ワークショップ論文集，1B-o1，2005.

[6] S.Matsuura, K.Fujikawa, and H.Sunahara. Mill: An Information Management and Retrieval Method Considering Geographical Location on Ubiquitous Environment In *The 2006 International Symposium on Applications and the Internet SAINT2006 Workshop-IPv6*, pages 14-17, Jan 2006.

[7] S.Ratnasamy, P.Francis, M.Handley, R.Karp, and S.Shenker. A Scalable Content-Addressable Network. In *Proceedings of the ACM SIGCOMM 2001 Technical Conference*,pages 161-172, August 2001.