

分散コンピューティング環境における空間結合問合せについて

横山 拓也[†] 石川 佳治^{††}[†] 名古屋大学 工学部 電気・電子情報工学科 情報工学コース ^{††} 名古屋大学 情報基盤センター

1 はじめに

近年、クラウドコンピューティングが注目されており、Google における MapReduce や GFS [2], Yahoo! が開発した Hadoop [1, 5] などの大規模分散コンピューティング技術の研究開発が進んでいる。本稿では、このような分散コンピューティング環境における空間情報 (spatial information) の処理方式について述べる。

例として、図 1 のデータを考える。ソーシャルネットワークサービスにおけるデータ処理を想定する。ファイル `usrs.txt` は、ユーザ情報を保持しており、ユーザ ID、ユーザ名、ユーザの住所の座標値などからなる。一方、`msgs.txt` は、ユーザが書きこんだメッセージの情報を保持する。ただし、ここでは前処理があり、地理情報 (例: 渋谷駅) がメッセージ中に含まれる場合、対応する座標値が抽出されているとする。図 1 では、たとえば 1, 2 番目の行は、メッセージ 1 が 3 月 10 日に記述され、その中には 2 つの点 (5, 69) と (25, 2) に対応する地理情報が含まれていたことを意味する。これらのファイルは分散ファイルシステムに格納されているとする。

```
> cat usrs.txt          | > cat msgs.txt
1 John (76, 77) ...      | 1 03/10/2010 (5, 69) ...
2 Mary (66, 69) ...      | 1 03/10/2010 (25, 2) ...
3 Mike (58, 32) ...      | 2 03/11/2010 (86, 69) ...
...                      | ...
```

図 1: 空間情報を含むデータ

ここで、SNS の各ユーザに対し、最近記述されたメッセージの中から、そのユーザの住所に近い (例: 2km 以内) 地理情報に触れているものを提示することを考える。SNS のデータは大規模であるため、バッチ処理で効率的に対応関係を抽出するものとする。この処理は、データベースの用語における空間結合 (spatial join) の機能があれば容易に実行可能である。しかし、分散コンピューティング環境では、一般に等結合 (equi-join) しか支援されていない。

そこで本研究では、この問題に対処するため、分散コンピューティング環境に適した空間結合処理方式を提案する。キー値による単純比較しか行えないという制約に対応するため、空間充填曲線を用いた処理方式を示す。本稿では特に、Hadoop [1, 5] の使用を想

定する。Hadoop 上に位置する、高レベルのデータ処理言語を含む大規模データ処理プラットフォームである Pig [3] との親和性の高い実現方式の提案を行う。

以下では、先の例で示したように、一方のファイル (`usrs.txt`) の各座標値に対し、ある一定の距離にあるエントリをもう一方のファイル (`msgs.txt`) から抽出し結合する空間結合について考える。しかし、空間結合の処理はこの形式に限定されるわけではない。以下で述べる手法は、矩形領域を設定して結合する場合などにも容易に一般化が可能である。

2 問合せの表現

Hadoop 環境では MapReduce 機能をじかに用いて Java などの言語で処理を記述することが基本である。しかし、これにはユーザの負荷が高いという問題がある。そこで本研究では、Hadoop 上に位置するプラットフォームである Pig の利用を考える。Pig [3] では Pig Latin と呼ばれる、データフロー処理を記述する高レベルのデータ処理言語が提供されている。本研究では Pig Latin を拡張して、空間結合条件を記述することを考える。

図 2 に空間結合の記述例を示す。最初に各ファイルを読み出し、FILTER 演算により 3 月 10 日のメッセージのみを抜き出している。そして JOIN 演算により、2 つのファイルの距離による結合処理を行っている。最後の DUMP 演算は結果の保存を指示している。

```
usrs = LOAD 'users.txt'
      AS (uid:int, name:chararray, uloc:point);
msgs = LOAD 'msgs.txt'
      AS (mid:int, date:chararray, mloc:point);
new_msgs = FILTER msgs BY date = '03/10/2010';
R = JOIN usrs, new_msgs WHERE within(uloc, mloc, 2.0);
DUMP R;
```

図 2: 拡張した Pig Latin による問合せ記述

この問合せの記述では、以下の 2 つの拡張を想定している。

1. LOAD 演算でファイルをロードする際に型の指定において、空間上の点データであることを示すための `point` データ型が指定できる。
2. JOIN 演算の中で空間結合の条件が指定できる: 具体的には `WHERE within(uloc, mloc, 2.0)` と指定している。

以下では、このような問合せをどのように処理するかについて述べる。

Spatial Join Processing in Distributed Computing Environments

Takuya Yokoyama[†], Yoshiharu Ishikawa^{††}

[†] Department of Information Engineering, School of Engineering, Nagoya University

^{††} Information Technology Center, Nagoya University

3 空間充填曲線について

Pig における結合処理はいわゆる等結合に限定されている。よって、以下では空間結合処理の一部を等結合で処理可能な形に変換し、Hadoop 上で実行するアプローチを示す。そのために空間充填曲線のアイデアを用いる。

空間充填曲線 (space-filling curve) [4] とは、空間上を埋め尽くす曲線であり、ヒルベルト曲線や Z オーダなどが知られている。2 次元空間をセルに分割したとき、空間充填曲線の順序に従ってセルに番号を付与すると、2 次元空間のセル集合を 1 次元の番号で順序付けできる。空間上で近くに位置するセルには一般に近い値が付与されるという特徴がある。本稿では、Z オーダ (Z-ordering) を例として考える。

Z オーダでは、 $2^n \times 2^n$ に区切られた 2 次元空間のセルを、アルファベットの Z のような形で順番にたどる。たどった順に 0 から $2^n \times 2^n - 1$ までの番号を付けることで、2 次元空間を 1 次元のデータで表すことができる。 n を Z オーダの次数と呼ぶ。図 3 に 2 次の Z オーダの例を示す。Z オーダの番号付けは簡単な規則に則っており、セルの位置と Z オーダ値の対応が容易に計算でき、異なる次数の Z オーダ値の対応付けが容易という利点がある。

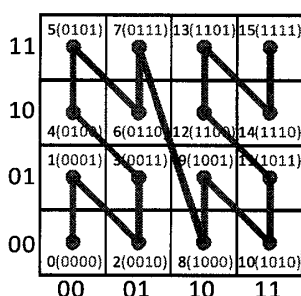


図 3: 2 次の Z オーダ

4 問合せ処理

図 2 に示したような問合せを処理するため、本研究ではプリプロセッサを構築する。プリプロセッサは拡張された Pig Latin 問合せを受け取り、これを Pig Latin の記述に展開する。図 4 に展開した問合せのイメージを示す。

```
usrs = LOAD 'users.txt'
      USING GeoLoadFunc(2, 2.0)
      AS (uid:int, name:chararray, uloc:chararray, uz:int);
msgs = LOAD 'msgs.txt'
      USING GeoLoadFunc(2, 0.0)
      AS (mid:int, date:chararray, mloc:chararray, mz:int);
new_msgs = FILTER msgs BY date = '03/10/2010';
tmp = JOIN usrs BY uz, new_msgs BY mz;
R = FILTER tmp BY within(uloc, mloc, 2.0);
DUMP R;
```

図 4: 展開した問合せ

LOAD 演算子で USING が用いられている。これは、ユーザ定義のデータロード関数を用いるための記述である。GeoLoadFunc は、1 番目の引数で指定された属性（この例の場合値が 2 なので 3 番目の属性）を点データとして解釈し、Z オーダ値を生成しタプルの末尾に付与する。2 番目の引数は、Z オーダ値を求めるときに距離をどれだけ考慮するかを表す。usrs については $r = 2.0$ という値が指定されている。これは、図 5 に示すように、距離検索のために対象の点から距離 r の円に交わるセルを考慮する必要があることを指示している。この図の場合、ユーザはセル 8 内に位置しているが、距離による結合のためにはセル 2, 3, 8, 9 を考える必要があることを意味している。結果として、usrs.txt の 1 行がこの場合 4 タプルに展開され、usrs に代入される。一方、msgs については $r = 0.0$ という値であるため、セルの展開は行われない。

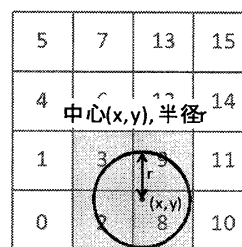


図 5: セルの展開処理

図 4 の tmp = JOIN ... の行においては、Z オーダの値による結合処理が行われる。これは等結合であるため Hadoop で問題なく処理できる。ただし、この結合結果には空間結合の条件を満たさない（距離が 2.0 より大の）ペアも含まれる。このため、FILTER 演算であらためてフィルタリング処理を行う。ここで within はユーザ定義のフィルタリング関数であり、文字列のペアを解釈して距離条件を判定する。

5 まとめと今後の課題

本稿では、Hadoop 上での空間結合処理を表現し処理するためのアプローチを提案した。今後はその実装を進め、また、最近傍問合せへの拡張なども図りたい。

謝辞

本研究の一部は科学研究費 (21013023, 19300027) の助成による。

参考文献

- [1] Hadoop homepage. <http://hadoop.apache.org/>.
- [2] 西田圭介. Google を支える技術: 巨大システムの内側の世界. 技術評論社, 2005.
- [3] Pig homepage. <http://hadoop.apache.org/pig/>.
- [4] Hanan Samet. Object-based and image-based object representations. *ACM Computing Surveys*, Vol. 36, No. 2, pp. 159–217, 2004.
- [5] Tom White. *Hadoop: The Definitive Guide*. O'Reilly, 2009.