空間索引機構を用いた検索の評価

6W－5　古本幸彦　大保信夫
筑波大学

1　はじめに

近年、地理情報システム (GIS) の分野において、データベース管理システム (DBMS) の利用が活発化している。しかし、従来のDBMSでは空間的な情報を効率的に管理することは困難である [1]。そこで、空間索引機構と、それを活用するアルゴリズムが必要となる。

空間索引機構を効率的に使うためには、空間的な問い合わせを分類し、それぞれに応じた検索アルゴリズムを用いることが必要である。このような観点における空間的な問い合わせの分類として、以下が代表的である [3]:

- topological queries
  位相的な関係にあるオブジェクトを検索する問い合わせ
- set-theoretic queries
  集合論的な関係にあるオブジェクトを検索する問い合わせ
- metric queries
  距離の概念を伴った関係にあるオブジェクトを検索する問い合わせ

本稿では、空間索引機構としてskd tree[5] を取り上げ、上で挙げた3種の問い合わせに対する検索アルゴリズムを示す。特に、metric queriesの一種であるnearest relationshipの検索に関しては、シミュレーションによって、そのアルゴリズムの有効性を検証し、GIS分野における様々な空間的な問い合わせに対して、単一の空間索引機構でも有効に対応できることを示す。

2　空間的な関係

空間的なオブジェクトとして以下の3種を考える:

ポイント 座標で示される0次元のオブジェクト
ライン 座標の組で示される1次元のオブジェクト
リージョン 閉じた一連のラインの集合で示される2次元のオブジェクト

これらのオブジェクト間の関係を、1)Topological Relationships、2)Set-theoretic Relationships、3)Metric Relationshipsに分類し、これらを更に以下のように細分する。

1. Topological Relationships
   - boundary relationship
   - co-boundary relationship
   - adjacency relationship
2. Set-theoretic Relationships
   - containment relationship
   - intersection relationship
3. Metric Relationships
   - range relationship
   - nearest relationship

3　skd treeと検索アルゴリズム

多次元空間のオブジェクトを扱う索引機構として、代表的なものにkd treeが挙げられる [2]。しかし、kd treeはポイントのみをその対象としている。そこで、広がりを持つオブジェクト (ライン、リージョン) まで扱えるように拡張されたものがskd treeである [4]。

skd treeではintersection searchとcontainment searchという2種類の検索アルゴリズムが提供されている。intersection searchは検索領域に重なるオブジェクトを抽出し、containment searchは検索領域に含まれるオブジェクトを抽出する。

Topological Relationshipsの内、boundary relationshipとco-boundary relationshipの検索は、intersection searchアルゴリズムを、検索領域に重なるノードだけでなく、接するノードまで走査するように修正することによって可能である。そして、adjacency relationshipの検索は、boundary relationshipのアルゴリズムとco-boundary relationshipのアルゴリズムを組み合わせることによって実現できる。

また、Set-theoretic Relationships、及びMetric Relationshipsの内、range relationshipのアルゴリズムは、intersection search、又はcontainment searchを適宜使用することによって効率的に検索することが可能である。

しかし、これらの検索アルゴリズムでは、nearest relationshipの検索にはうまく対応できない。そこで、nearest relationshipの検索は以下の手順で行なう。

1. ノードが葉ノードならば、ノード内の全てのオブジェクトを調べる。
2. ノードが内部ノードならば、より近い子ノードから走査する。
3. 子ノードまでの距離が、現在見つかっているオブジェクトまでの距離よりも遠ければ、その子ノードは走査しない。

4　実験と評価

nearest relationshipの検索アルゴリズムの効率を評価するために、以下のような条件の基で、“ポイントPから最も近いラインセグメントを検索する”シミュレーションを行なった。

Evaluation of Search Algorithms with a Spatial Index
Yukihiko FURUMOTO, Nobuo OHBO
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305, Japan

1. 空間の大きさは10000 × 10000
2. オブジェクトはpointとlineのみとし、lineは全て軸と平行
3. オブジェクトの数は100000
4. ポイントとラインは空間上に一様に分散
5. コストはブロックアクセスの数によって評価 (ブロックのサイズは4096)
6. P は空間の中心付近に、7 × 7の格子状に設定し、49回の評価の平均を結果とする

4.1 シミュレーション 1

シミュレーション 1では、ラインの長さを変化させた。シミュレーションは2種類行ない、ラインの数を、それぞれ20000、80000とした。この結果を図1に示す。ブロックアクセスの数はラインの長さにほぼ比例して増加していることが分かる。これは、ラインが長くなると、最も近いラインが離れた葉ノードに格納され得るためである。

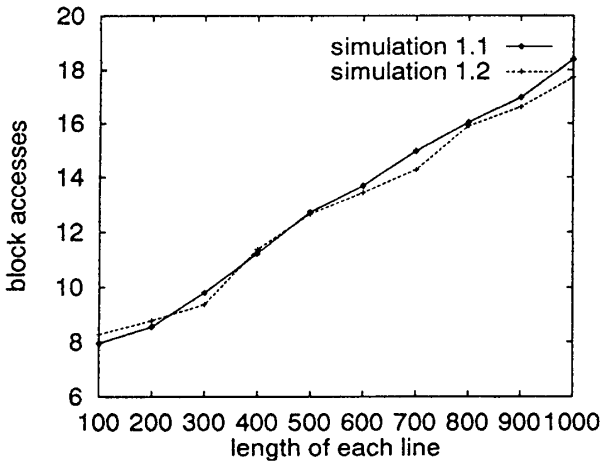

図 1: シミュレーション 1の結果

4.2 シミュレーション 2

シミュレーション 2ではポイントとラインの比率を変化させた。シミュレーションは3種類行ない、ラインの長さを、それぞれ100、200、300とした。この結果を図2に示す。ブロックアクセスの数は、ラインの比率が低い場合に非常に高い。しかし、ラインの数がある程度まで増加すると、ブロックアクセスの数はほぼ定数となる。これは、最も近いラインが P と同じ葉ノードに格納されているためである。

5 まとめ

本稿では、単一の空間索引機構によって様々な空間的な問い合わせに対応できることを示した。そして、skd treeを用いてTopological Relationships、及びMetric

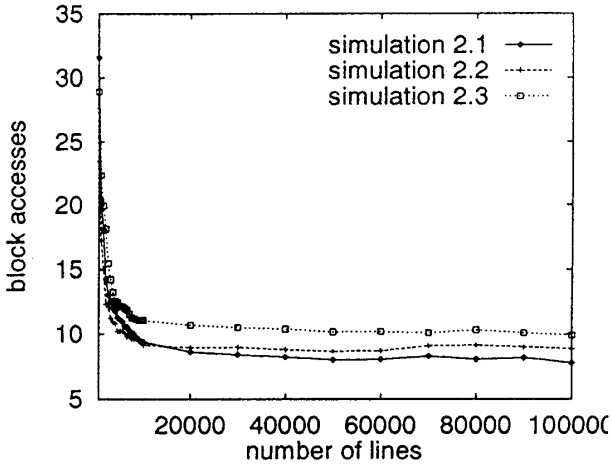

図 2: シミュレーション 2の結果

Relationshipsの検索を行なう効率的なアルゴリズムを示し、シミュレーションによってそのアルゴリズムの有効性を示した。そのアルゴリズムは一般のアプリケーションにも適用できるといえる。

本稿では、ポイントと固定長のラインが一様に分散している状況を仮定したが、今回とは異なった条件の基でのnearest relationshipアルゴリズムの評価を行なう必要がある。そして今後は、R-treeなど他の空間索引機構を用いて同様の評価を行なっていきたい。

また、日本デジタル道路地図協会作製の道路地図を用いて、シミュレーションの結果の正当性の検証を行なっている。

参考文献

[1] Kim, W., Garza, J., and Keskin, A., Spatial data Management in Database Systems : Research Directions. Proc 3rd Symp. on Large Spatial Databases, in LNCS Vol.692 1993 pp.1–13

[2] Bentley, J.L., Multidimensional binary search trees in database applications, IEEE Trans. on Software Eng. SE-5,4, 1979, pp.333–340

[3] Floriani, L.D., and Marzano, P., Spatial Queries and Data Models. Proc European Conf. on Spatial Information Theory, in LNCS Vol.716 1993 pp.113–138

[4] Ooi, b.C., Efficient Query Processing in Geographic Information Systems, in LNCS Vol.471 1990

[5] Ooi, B., McDonell, K.J. and Sacks-Davis, R., Spatial kd tree : An indexing Mechanism for for spatial Databases, Proc on IEEE Computer Software & Applications Conf. 1987 pp.433–438