D-015

空間インデックスＧＢＤ木の高速初期構築法
High-speed Initial Construction Method of Spatial Index GBD-Tree

根岸　幸生† 川崎　洋† 大沢　裕†
Yukio Negishi Hiroshi Kawasaki Yutaka Ohsawa

1. はじめに

現在、地理情報システムの分野では、大容量の大縮尺ディジタル地図の普及や、過去の地物をすべて管理し、任意の時点での地図を再現できる時空間 GIS の普及などにより、大規模空間データを管理する必要がある。大規模空間データを管理する手法としては、R 木 [3] や k-d 木 [2]、GBD 木 [1] などがある。この中でも GBD 木はデータ挿入や削除といったデータ管理のための補助情報と、データ検索のための補助情報を別々に持つことにより、検索性能を犠牲にすることなく高速なデータ更新が可能である。しかし、昨今の大規模空間データを GBD 木で管理する場合には、従来のインデックス作成方式では、初期構築の際に長い処理時間がかかる。筆者らは、すべてのデータをまとめて投入し、バッチ処理的に GBD 木のインデックスを高速に構築する方式を提案する。また、本手法を用いて、当研究グループで開発している時空間情報管理システム STIMS[4] の空間インデックスの作成実験により、この手法を評価した。

2. GBD 木

GBD 木は R 木や k-d 木と同様に階層的にデータの存在する空間の分割を行い、その分割過程を木構造で管理する図形データ管理構造である。図 1 は GBD 木の構造を示している。GBD 木の木構造は N 個のスロットを持つノードにより構成されている。各スロットには子ノード (葉ノードでは図形データ自身) へのポインタと、その子ノードが対応する空間を表す領域式、および子ノードを根とする部分木の全データを含む外接長方形 (MBR) を持つ。領域式は図形の空間上での位置と大きさを表現するものである。領域式、ならびに GBD 木の詳細については文献 [1] を参照されたい。

3. バッチ処理によるインデックス作成方式

従来、GBD 木にデータを追加する場合には、図形データを一つずつ投入する手法を用いてきた。しかし、この手法では昨今の大容量の地図データの空間インデックスを作成する場合には長い処理時間を必要とする。このため、すべてのデータをあらかじめ領域式を用いてソートしておき、バッチ処理的に空間インデックスを生成する方法を提案する。この処理は以下の手順で行われる。

1. GBD 木に挿入するすべての図形の中心点座標の領域式を求め、ルートノード直下の葉ノード（L）に、すべての図形を挿入する。
2. ノード L のスロットを領域式の値で昇べきの順にソートする。
3. ノード L のスロット中の図形の領域式を参照し分割するための領域式 (VD) を求める。この領域式の求め方は後述する。
4. 新しい葉ノード（NL) を作成し、ノード L のスロット中の図形のうち領域式 VD に含まれるものをノード NL のスロットに移動させる。
5. ノード L とノード NL のスロット数を調べ、スロットの数が最大スロット数以上ならば、4–5 の処理を再帰的に繰り返す。
6. すべての処理が終了した後、ルートノードからツリーをたどり、GBD 木のインデックスをディスクに書き込む。

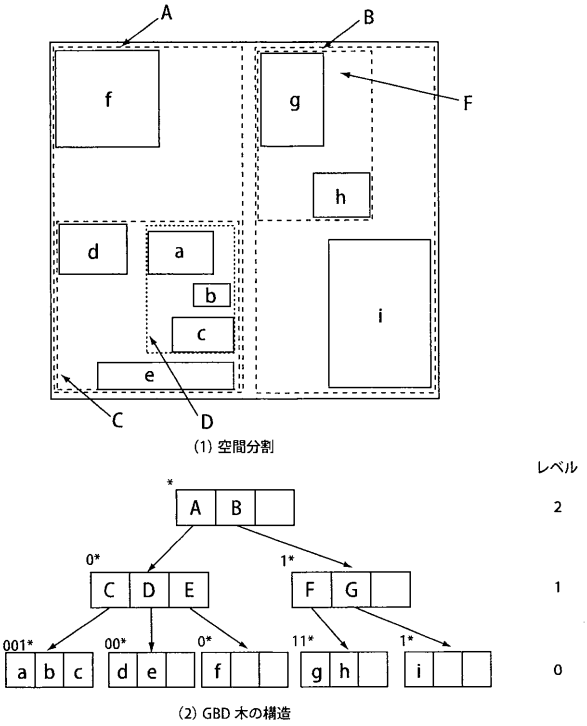

図 1: GBD 木の概要

3 において決定される領域式 VD は次のように求める

1. 領域式を最初のスロットの領域式で初期化する。これを TD とする。
2. 2 番目のスロットから順に調べ、TD と各スロットの領域式の共通部分を求める。たとえば TD が「001101…」であり、スロットの領域式が「001111…」であった場合、共通部分は「0011*」となる。
3. TD を求めた共通部分の領域式で更新する。

†埼玉大学工学部情報システム工学科

4. すべてのスロットを調べた後、TD の末尾に 0 を足したものが求める領域式 VD になる。

4. 性能の評価

4.1 各手法による性能評価

前章にて述べた手法を実装し実験により評価した。実験は、時空間情報管理システム:STIMS のデータファイルを読み込み、空間インデックスの作成にかかった時間を測定した。実験には、表 1 に示した 4 種類の数値地図を用いた。計算機へのインプリメントでは領域式を 64 ビット、領域式の長さを 8 ビットで表現した。実験には、今回提案した手法と従来の手法について、空間インデックス作成にかかった時間を測定した。実験結果を表 2 に示す。また、本実験では、GBD 木の最大スロット数を 50 とした。このスロット数は現在、STIMS で用いてられている最大スロット数である。

表 1: 実験に用いた数値地図

数値地図	縮尺	データサイズ	オブジェクト数
JMCマップ KS5339	1/200,000	840KB	8,583
数値地図 25,000 さいたま市	1/25,000	1.4MB	27,074
トロピカルテクノセンター 沖縄市	1/2,500	10MB	151,699
J-Mapple 200,000 日本全国	1/200,000	120MB	610,756

表 2: 各手法ごとの処理速度

数値地図	提案手法	従来手法
JMCマップ (1/200,000) KS5339	0.5	1.6
数値地図 25,000 さいたま市	1.5	5.2
トロピカルテクノセンター 沖縄市	9.1	31.6
J-Mapple 200,000 日本全国	38.1	148.3

単位［秒］

提案手法は、どの数値地図においても良好な結果が得られていることがわかる。また、従来手法との比較では、どの数値地図においても約 3 倍程度、速度が向上した。

4.2 最大スロット数を変更した場合の性能評価

次に、ノードあたりのの最大スロット数を変化させた場合についての評価実験を行った。実験内容は、先の実験と同様に空間インデックスファイルの作成にかかった時間を測定した。実験に用いた数値地図は表 1 の沖縄市のデータを用いた。測定はそれぞれの手法について、スロットの最大数を 10 から 500 まで変えて測定した。実験結果を図 2 に示す。

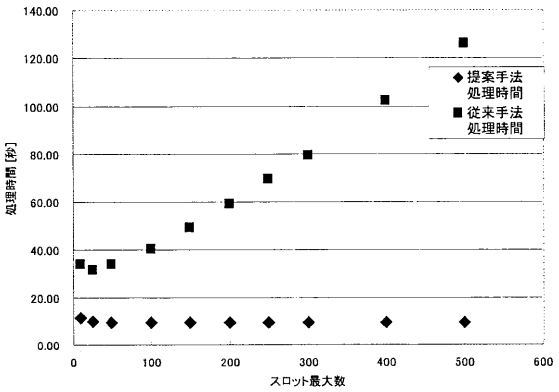

図 2: スロット数を変更した場合の処理速度

提案手法は、スロット数が多くなるに従い処理時間が緩やかに減少している。一方、従来の手法ではスロット数が多くなるに従い処理時間が線形的に増加することがわかる。これは、提案手法では、すべてのスロットを領域式でソートし、その後、ノード分割をしているのに対し、従来の手法ではデータを投入するごとに、ノード内のスロットを領域式順にソートするため、ノードあたりのスロット数が多いほど処理時間が増加する。

5. まとめ

本稿では、GBD 木の空間インデックスの初期構築を高速化するための手法について述べた。

提案手法は従来の手法より約 3 倍ほど高速にインデックスを構築することが可能である。またスロット数が増えた場合にはさらに差が大きくなる。このため、大規模データや精度の高い大縮尺の地図データを管理する場合には本手法が有用である。

今後の課題としては、提案した手法ではスロットが使用される率の低いノードがわずかに存在する。また、空間の分割についてもより最適な分割方式を考えていく必要がある。

参考文献

[1] 大沢裕, 坂内正夫: 2 種類の補助情報により検索と管理性能の向上を図った多次元データ構造の提案, 電子情報通信学会論文誌, Vol. J74 No. 8 pp. 467–475 (1991).

[2] Bentley J.L.: "multi Dimesional Binary Search Trees Used for Associative Searching", Commun. ACM, 18, 9 (1975).

[3] Antonin Guttman : "A Dynamic Index Structure for Spatial Searching", Proc. of the 1984 ACM-SIGMOD International Conf., pp. 47–57 (1984)

[4] 大沢裕、長島敦、トポロジー暗示型時空間情報管理システム：STIMS、第１２回機能図形情報システムシンポジウム講演論文集、pp. 27-36,2000