【論文】

差分記述に基づく時系列データ管理構造

大沢 裕・金 景月

A Spatio-temporal Data Management Structure Based on Differential Script

Yutaka OHSAWA, KIM Kyungwol

Abstract: According to popularization of geographic information system (GIS), temporal data management becomes very important. When GIS is used in daily work, the most frequently accessed data is the present data. So, the access speed on the present data should not be lowered in spatio-temporal data management. Furthermore, the past data should also be able to be restored. This paper proposes a spatio-temporal data management method based on a geographic differential script file (GDSF) method. This method records the change of map in the form of graphical operation with the time print. When an old data becomes necessary, the data is restored by applying the GDSF onto the current map data. For executing spatial retrieval fast, this paper describes a combination method of spatial data structure GBD tree and the GDSF.

Keywords: 時空間データ (spatio-temporal data), 空間データ構造 (spatial data structure), 差分記述ファイル (differential script file)

1. はじめに

従来、地理情報システムは、2次元平面上や3次元空間中に配置されている地物や人工建造物の情報を扱ってきた。しかし、近年、これに時間軸を加えた3次元または4次元の時空間データを扱う要求が高まっている。例えば、設備管理監視システムにおける工事履歴の管理や、土地台帳における土地区分の分割・合併、土地利用変化等を管理したいという要求が高い。

4次元時空間の管理方式としては、3次元空間に時間軸を加えた4次元データとして扱う方式（黒木・牧之内, 1996）、データ管理構造自体に変化履歴を持たせる方式などが提案されている。まず、4次元データとしての管理は、バーチャルリアリ

ティーの様な映像生成分野での応用には適している。しかし、設備管理や土地管理などにおける地図情報は、変化が緩やかであり、空間的な広がりと同様に連続的な座標軸として時間軸をとらえることには無駄が多い。一方、寺岡ら（寺岡ほか, 1995）は、空間を領域分割型の木構造で管理し、データの追加・削除に伴う木構造の変化（ノードの分割や併合）をタイムプリントと共に残す方式を提案している。この方式は更に平衡木に拡張されている（中村・出木原, 1999）。このアプローチは、地図情報においては、現在の状況に対するアクセスが最も多いこと、時間軸における変化は離散的なもので十分であるという地図情報の特質に立脚しており、優れたものと思われる。但し、過去の木構造を全て保有している為、データ量の点で負担が大きい。また時間木と空間木を検索範囲に応じて適応的に切替える方式（仲ほか, 1996）も提案されている。

本論文では、上述の地図情報の特性、即ち、地理情報では現在のデータへのアクセス頻度が高いこ

大沢：〒338-8570 浦和市下大久保255
埼玉大学工学部情報システム工学科
Dept. of Information and Computer Sciences, Saitama University
Shimo-okubo, 255, Urawa, Saitama
e-mail: ohsawa@mm.ics.saitama-u.ac.jp

以下，2.において方式の概要と提案方式の様な空間管理が必要な理由について述べる．3.ではGBD木をベースとした時間情報管理方式について詳しく論じる．4.では実験結果を示す．最後に5.でまとめと，今後の展望について述べる．

GISを日常業務で利用している場合、日々現実世界の動きに合わせてデータの追加、削除が行われる。そのGISでもっともアクセス頻度が高い情報は最新の情報と考える。一方、過去の時点を指定して、そのときの「世界」の上で各種空間演算を行いたい場合がある。10年前の土地所有者を知りたい、または土地利用の状況を知りたいという要求である。より高度には、10年前の道路網においての、ある2点間の所要時間と、現在の所要時間とを比較したいなどの要求も考えられる。本論文で目指すデータ構造は、このような要求に応えるためのものである。即ち、通常は現在のデータを対象として何らオーバーヘッドなく処理を実行し、必要に応じて過去の世界を再現でき、その上で空間演算が実行可能なデータ構造である。

る。これは「現在データへのアクセス頻度が最も高い」という前提に反する。

この方式は、現在データへのアクセスに関してはオーバーヘッドがないという目的を満足するが、別の問題を生じる。つまり、図1に示すように過去のデータ上で空間演算を実行するためには、(1) 過去の世界を作り出し、その上で空間演算を実行し、(2) 再び現在の世界に戻す、という流れになる。GDSコマンドの適用により、データベースが過去の世界のものになってしまうため、演算が終了したあと(2)の処理が必須である。またGDS方式は、図葉またはメッシュ単位で作成されることから、空間検索を効率良く実行するためには別途空間データ管理構造を必要とする。

本論文では、空間データ管理のためのデータ構造

として GBD 木を用いる。GBD 木は R 木と同様に、階層的な多分木を用いて空間データを管理するデータ構造である。R 木 (Guttman, 1984) ではデータの挿入・削除の際にも、データ検索時にも、各ノードに付けられている MBR (Minimum Bounding Rectangle) が参照される。一方、GBD 木ではデータの挿入・削除の際には、各図形の MBR の中心点位置により、木構造上でそのデータが位置する葉ノードが決定される。空間検索は、R 木と同様に各ノードに付けられている MBR を参照して行われる。GBD 木ではこの役割分担により、データ挿入に伴いノード分割が生じる際の計算量を低減している。

GBD 木の詳細については文献 (大沢・坂内, 1991 ; 大沢ほか, 1992) に譲るが、空間データ管理の概略は以下のとおりである。

図形データは MBR の中心点 (代表点) 位置により分割管理される。まず、領域分割の基本は、図 2 に示すように、横 (X) 軸と縦 (Y) 軸により交互に 2 等分を繰り返して得られる領域を基本として管理される。この分割過程を 2 進数で表現したものを領域式と呼ぶ。例えば図 2 で網掛けを施した領域は、まず全体を X 軸で 2 等分した内の下側、次にその分割された領域を Y 軸で 2 等分した上側、更にその領域を X 軸で 2 等分した領域の上側であることから「下側」を「0」に対応させ、「上側」を「1」に対応させたとき、「011 *」と表現できる。ここで記号「*」は終了記号である。

図 3 (a) の空間データに対応する位数 3 の GBD 木を図 3 (b) に示す。図 3 (a) で点線は GBD 木の領域分割を表している。また a ~ k は図形を表しており、(a) と (b) で対応している。ここでは、各図形の MBR の形状を示している。

GBD 木では、ある分割領域に新しいデータが挿入された結果、予め定められているノードの容量 M (図 3 では $M=3$) を超えたとき、その分割領域は更に 2 つの領域に細分される。この分割は図 2 に示したルールで行われ、 $\lfloor M/3 \rfloor + 1$ 以上のデータを含む 2 つの領域に分割される。但し記号 $\lfloor x \rfloor$ は、 x より大きくない最大整数を表す。

GBD 木の領域分割では、領域式で表現可能な長方形 (または正方形) 領域と、それ以外の部分に分

図 2 領域式と領域分割

(a)

(b)

図 3 GBD 木における分割領域と木の対応

割されることになる。例えば、図 3 (b) の根ノードの 2 つのスロットにかかっている領域式「0 *」と「*」は、領域分割の結果「0 *」領域と、それ以外の領域に分割されていることを示している。また、ノード F が指し示す子ノードの 3 つのスロットに書かれている領域式、「00 *」、「0110 *」と「0 *」はこのノードにより「00 *」領域と「0110 *」領域、さらにそれ以外の「0 *」領域に分割されていることを示している。また、各ノードのスロットは、ある領域 R を包含する領域 Q が必ず R より右側に位置するように並べられている。

GBD 木への新しいデータの挿入は以下のように行われる。まず挿入する図形の MBR を求め、その中心点位置 (C) を決定する。次に、GBD 木の根ノードについて、その左端の-slot から領域式を調べていき、C を包含する slot を探す。更にその slot が指し示す下位のノードについても同様な処理を葉ノードに至るまで再帰的に繰り返す。もし、その葉ノードに空き slot があれば、そこにデータを挿入し処理を終了する。一方、既にその葉ノードの全ての slot にデータが登録されているときは、葉ノードの分割を行い、新しい葉ノードを指し示す slot を親ノードに挿入する。もし親ノードに slot の空きが無ければ、その親ノードも 2 つのノードに分割し、あらたに分割により作られた slot が上方に伝播していく。この機構は R 木などと同様である。

GBD 木の各 slot には領域式のほかにその slot より下位の木構造中に含まれる全ての図形を内包する MBR が置かれている。範囲検索などで下位の構造を調査する必要があるかをこの MBR と検索範囲の重なりの有無により判断する。

以上で述べた GBD 木の構造を整理する。

○GBD 木のノードは、M 個の slot を持つ

○葉ノード以外の slot は次の構成を持つ

〔領域式, MBR, 子ノードへのポインタ〕

○葉ノードの slot は次の構成を持つ

〔領域式, MBR, 図形の実体へのポインタ〕

3.2 GBD 木への GDS の付与

以上で述べた GBD 木を、過去の履歴を管理できるように拡張する。このために、GBD 木の全ての葉ノードに優先順位付き過去キューへのポインタを付加する。これは過去のデータに対応する GDS コマンド列を記録するためのものである。また、過去のデータについても 3.1 で述べたものと同じ処理により所属する葉ノードが決定される。各 GDS コマンドは、そのイベントが発生した時間 (T)、イベントの種類 (K)、図形形状をあらわすグラフィックデータ (G) からなる。

過去キューでは T が新しい順（現在から過去に向かう順）に優先順位が付けられ GDS コマンドが

並べられる。即ち、このキューからは、GDS コマンドを新しいものから順に取り出すことができる。

GBD 木の全てのノードに対して、そのノードより下位の葉ノードに置かれている過去のデータを内包する MBR を付与する。前節で述べた現在のデータに対応する MBR と、この過去のデータの MBR を区別するため、前者を MBRC、後者を MBRP と呼ぶ。

まず、現実世界の変化に合わせて現在のデータを管理している GBD 木から図形を削除する場合には次の一連の処理が行われる。

〔図形の削除〕

- (1) GBD 木上で削除されるデータが含まれる葉ノードを探す。この検索は領域式を用いて実行される。
- (2) その葉ノードで次の処理を行う。
 - (2-1) 葉ノードの slot からその図形を削除する。
 - (2-2) その図形を挿入する GDS コマンド（削除に対する逆コマンド）を作る。
 - (2-3) それに削除された時間を付与して、現在の葉ノードの過去キューにその GDS コマンドを挿入する。
- (3) この移動に伴う両 MBR の変化を反映させるため、根ノードに向かって MBRC と MBRP を調節する。

上記の処理では図形の削除により GBD 木のノードにアンダーフロー（使用 slot 数が予め定められている最少数を下回ること）が発生する場合がある。その際には 2 つのノードを 1 つに合併する処理が行われるが、ここでは説明が煩雑になることを避ける為、記述を省略している。

次に、新しいデータを GBD 木に投入する場合には、次の一連の処理を行う。

〔図形の挿入〕

- (1) GBD 木を根ノードからたどり、そのデータが挿入される葉ノードを決定する。
- (2) その葉ノードに次の手順でデータを登録する。
 - (2-1) 新しいデータを葉ノードに登録する。
 - (2-2) 新しいデータを削除する GDS コマンド（挿入に対する逆コマンド）を作成し、現在時間をタイムプリントとして、そのノードのキュー

に登録する。

- (2-3) もしこの挿入により葉ノードのオーバーフローが発生すれば、葉ノードのデータを2つのノードに分割し、キューのデータも新しいノードの領域式に合わせて分割する。
- (3) MBRC と MBRP の変化を根ノードに向かってたどりつつ調節する。もし、この挿入によりノードのオーバーフローが発生した場合には、通常の GBD 木と同様な方法で、新しく発生したノードを親ノードに挿入する。

図形の変形や移動などの変化は上に述べた挿入、削除を組み合わせることにより対応できる。つまり、旧形状や移動前の位置における図形を削除し、新形状や移動後の位置に対して図形を挿入することにより実現可能である。

この他道路名称変更などの属性の変化は、GBD 木には影響を与えないため単純である。図形の属性を変えると共に、その図形が置かれている葉ノードに付けられているキューに属性を変更する GDS コマンドを、タイムプリントと共に挿入すれば良い。

時系列管理に拡張した GBD 木の構造は以下の様になる。

○非葉ノードは M 個のスロットから構成される。
スロットは次の要素からなる。

[領域式, MBRC, MBRP, 子ノードへのポインタ]

○葉ノードは M 個のスロットと、過去キューへのポインタから構成される。

スロットは次の要素からなる。

[領域式, MBRC, MBRP, 図形の実体へのポインタ]

3.3 空間検索

GIS では様々な空間検索、即ち空間的な位置関係に依存した検索が行われる。それらの基本となる検索に、縦横が座標軸に平行な辺で囲まれた範囲に含まれる全ての図形を列挙する範囲検索がある。多くの空間検索は、この範囲検索をベースに実行することができる。

まず、GBD 木の上で、現在のデータを対象とした検索範囲 R の範囲検索は次のように実行できる。

[現在データの範囲検索]

- (1) GBD 木の根ノードへのポインタを p に代入する。検索結果を入れる集合 S を空にする。
- (2) p が指し示すノードで、使われている全てのスロットに対して以下の演算を行う。
- (3) スロットの MBRC と R との重なりをチェックする。重なりがあるとき以下の処理を行う
- (3-1) スロットが示す子ノードが葉ノードの場合、その葉ノードのスロットが示す図形データを調べ、真に R と重なるデータを集合 S に追加する。
- (3-2) スロットが示す子ノードが葉ノードでない場合、そのスロットが示す子ノードを p として (2) 以下を再帰的に呼び出す。

過去のある時点 (Tp) を指定して、その時のデータ上での検索範囲 R の範囲検索は以下の様に行われる。

[過去のデータ上での範囲検索]

- (1) GBD 木の根ノードへのポインタを p に代入する。検索結果を入れる集合 S を空にする。
- (2) p が指し示すノードで使われている全てのスロットに対して以下の演算を行う。
- (3) スロットの MBRC と R、及び MBRP と R との重なりをチェックする。何れかで重なりがあるとき以下の処理を行う。
- (3-1) スロットが示す子ノードが葉ノードの場合、その葉ノードの各スロットが示す図形データを全て作業メモリ (W) にコピーする。次に過去キューの GDS コマンドのタイムプリントを調べ、それが指定時間 Tp より新しいうち、W に適用する。これにより W 中のデータは“指定時間の世界”になる。W 中のデータを1つずつ調べ、真に R と重なるデータを集合 S に追加する。
- (3-2) スロットが示す子ノードが葉ノードでない場合、そのスロットが示す子ノードを p として (2) 以下を再帰的に呼び出す。

上記のステップ (3-1) の概念を図 4 に示す。

以上の検索アルゴリズムを用いることにより、現在のデータを“汚す”ことなく、過去の指定時刻の

図4 葉ノードにおける空間検索の流れ

データが復元でき、その上で空間検索を行うことができる。葉での検索終了後、W 中のデータは棄却する。

4. 実験結果

以上で述べたデータ構造は、現在データを対象とした検索に対しては、過去のデータをあわせて管理していることに関して何らオーバーヘッドなく実行することができる。これは手続き「現在データの範囲検索」が MBRP を参照しないこと、及び GBD 木の現在データの部分は過去データにより容量が増大しないことから明らかである。

一方、過去のデータの検索に際しては MBRP を考慮に入れなければならない。そこで、同一の検索範囲であっても、現在データの検索と比べてその検索の際にたどらなければならないノード数が増大する。この際に増大するノード数は、MBRP の広がり依存する。

この性能を評価するため、以下の実験を行った。ここで取り上げた時系列の実データを入手することは困難である（存在するものは非常に少なく、また公表されていない）ため、国土数値情報（FD マップ、KS5339）の道路をベースに実験用の時系列データを作成した。

まず、データセット中の道路セグメント（主として交差点間のリンク）に順序番号をつけた。次に擬似乱数により、(A) 全体に占める比率 p のデータを削除データとし、(B) 同様に p のデータを追加データ、(C) 残りの $(1-2p)$ から $(0.5-p)$ を選

び不変のデータとした。ここで (C) のデータ数をこのような値に定めている理由は、 p を変化させても GBD 木中の現在データの数を一に定めるためである。また 0.5 は p を変化させたときの上限を全データ数の $1/2$ に定めているためである。更に (A) と (B) には擬似乱数により過去のタイムプリントを付与した。

以上の処理で作られる実験データの個数は次のようになる。まず、現在は (B) と (C) のデータが存在し、(A) のデータが削除されている。また過去の指定された時点では、(C) の全てと、(A) のうちタイムプリントが指定時点以降、(B) のうち指定時点以前のデータが存在している。

R 木や GBD 木のような、階層的領域分割型のデータ構造を GIS に適用する場合、全てのノードをディスク上に実現する。従って、木をたどる処理に要する時間の多くはディスク上でのアクセス時間となる。個々のデータが検索範囲内に含まれるか否かの判定のような処理は、この時間に比較すると大幅に小さなものとなる。そこで、提案方式の検索時間を評価するためには検索に際してアクセスを要したノード数を調べるのが重要となる。

すでに述べたように、本データ構造では現在の情報の検索に際しては MBRC を、過去の情報については MBRC と MBRP を参照しつつ木のノードをたどる。一般に MBRP は MBRC を内包するため、過去の情報の検索では現在の情報の検索に比べて多くのノードをたどる必要がある。この比率が過去データ検索に要する検索時間の増大率となる。

図5 時系列データ比 (%)

図5は対象データの範囲（1次メッシュの範囲を用いている為、約80km四方）に対して、面積比が0.2%から5%の正方形領域中のデータを範囲検索する際にたどるノード数の比率を示している。つまり、（過去の時点のデータ検索の際にたどるノード数）/（現在のデータ検索の際にたどるノード数）を示している。具体的には、この比率が2であれば、過去のデータの検索には現在データの検索のほぼ2倍の時間（但し、ディスクのキャッシュ性能を無視した場合）が必要となることを表している。横軸は挿入または削除されたデータの全データ数に対する割合を示しており、先に述べた p の2倍の値である。例えば、この値が20%である場合、全データの内の2割が過去において挿入または削除されていることを表している。

まず、0.2%のように検索範囲が小さな場合は、図形のピックやノードに接続する他の線分を求めるような局所的な処理が対応する。一方、5%のように検索範囲が広い場合は、地図のズーム表示のような処理が対応する。

図5の結果から、検索範囲に関して次の性質がわかる。検索範囲が狭く、かつ時系列データの比率が高い（即ち、データの動きが大きい）場合には2倍以上の処理時間を要する。検索範囲が広くなるにつれ、この比率は下がり、全体の5%の範囲の検索では90%のデータが動きのあるデータでもこの比率が1.2倍程度に抑えられている。

図5では、全ての検索範囲において右肩上がりの傾向が見られるが、これは次の理由によるものであ

る。現在までに挿入や削除が行われたデータはMBRPを拡大する。従って、その頻度が高ければMBRCとMBRPとの面積の比率も大きくなり、MBRPを参照する過去データの検索でたどるノード数も増大することになる。

例えば、非常に大きな図形が過去データにある場合、MBRPがその図形により極端に拡大されることになり、検索処理時間は大幅に増大する。しかし、この状況は階層的領域分割型データ構造が共通して持つ欠点であり、対策としてはMBRを極端に拡大しないように図形が分割される。幸い、国土空間データ基盤など多くの地図データは図葉ごと、メッシュごと道路などの線が切断されており、極端にMBRを拡大することはない。

5. おわりに

本論文では、地理情報システムにおける時系列データの新しい管理方式として、GBD木とGDS方式を組み合わせることにより、過去の指定された時点のデータ上での空間検索を行う方式について述べた。

本データ構造は、現在のデータの検索に際しては過去データを管理していることによるオーバーヘッドのない検索が可能である。一方、過去のデータの検索に際しては、過去データの比率が増加するに従って、検索時にたどる必要のあるノード数が増加する傾向を示している。特に、検索範囲が小さい場合に、この傾向は顕著となるが、検索範囲が狭い場合には検索時間の絶対値が小さく、深刻な問題には

ならないと考える。

本論文では、空間情報管理方式として GBD 木を採用し、それに対する時系列管理の拡張について述べた。提案方式は GBD 木特有の性質により効率良い検索を行えるが、R 木や R* 木などの他のデータ構造に対して、単純には適用できない。そのような一般のデータ管理構造への拡張は今後の課題である。また、ここではデータ構造の観点から時系列データを扱っており、実際の GIS における時系列管理への適用も今後の課題である。

謝辞

本研究は、一部文部省科学研究費、創生的基礎研究 09NP1401（研究代表者：坂内正夫）の補助による。

参考文献

大沢 裕・坂内正夫（1991）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」, J74-D-I, 8, 467-475

大沢 裕・林英明・坂内正夫（1992）空間検索を重視した幾何情報処理環境, 「テレビジョン学会技術報告」, ICS82-35, 13-18

黒木 進・牧之内顕文（1996）単体複体の概念を用いた時空間データモデル Universe の設計, 「情報処理学会研究会報告」, 96-DBS-109, 221-226

仲 篤起・才脇直樹・辻本浩章・西田正吾（1996）時空間データの高速検索のためのデータ管理の一方式, 「高度データベース松江ワークショップ講演論文集」, 2, 349-355

中村泰明・出木原裕順（1999）, 時間属性をもった空間データの管理構造－PMD 木－, 情報処理学会論文誌, 40, SIG 5 (TOD 2), 54-68

寺岡照彦・丸山 稔・中村泰明・西田正吾（1995）空間検索を効率化した時空間データ管理構造の提案, 「電子情報通信学会論文誌」, J78-D-II, 9, 1346-1355

Guttman, A. (1984) R-Trees: A dynamic index structure for spatial searching, *Proc. ACM SIGMOD*, 47-57.

Ohsawa, Y., Kim, K. (1999) Geographic differential script file method for distributed geographic information systems, *IEICE Trans. INF & SYST*, E82-D, 1, 113-119.