移動オブジェクトの更新に適した領域分割形木構造

西川 嘉人[*1]　辻 俊明[*2]　金子 裕良[*2]　阿部 茂[*2]

１．まえがき

GPS などの位置情報取得技術と無線通信技術の進歩に伴い，車や人といった時々刻々と位置の変化する物体(以下，移動オブジェクトと呼ぶ)の位置情報の継続的な入手が容易となり，位置情報を用いた運行管理やセキュリティ管理，情報配信などのサービスが検討されている。

大量の位置データの管理構造として R-tree[1]が広く用いられてきた．しかし R-tree は本来大規模 CAD や地理情報システムなどの静的データの管理が目的であるため，データ更新における処理コストが大きい問題がある。

本論文では移動オブジェクトの最新位置管理における頻繁な更新を高速に処理する kdm-tree を提案する。kdm-tree は R-tree と異なる領域分割形であり，データ削除性能に優れた新しいノード分割アルゴリズムに特徴がある。

計算機実験において，kdm-tree は従来の手法に比べて更新速度は 25 倍以上となり，メモリコストや検索速度は同等以上の性能となった。

２．移動オブジェクト管理

移動オブジェクトは陸上や海上，空といった 2，3 次元空間に存在し，多くの場合地図情報と対応させた管理が行われる。移動オブジェクト管理の応用として，交通システムの効率化や事故回避への利用，海流調査，エリアの出入り管理などによるセキュリティ管理，位置情報に基づいた広告やサービス情報を配信するシステム(Location Based Service: LBS)などが考えられる。

携帯端末を用いた LBS の例として，ユーザが現在位置から付近のレストランや駐車場を検索するシステムや，逆に店舗側から付近のユーザに対してクーポンなどのサービスを提供するシステム，地域のイベントを知らせるシステムなどが考えられる。

これらの応用において，サーバは頻繁に生じる大量の位置情報更新を高速に処理し，かつ高速に検索を行う必要がある。

３．更新管理に適したデータ管理構造

データ管理とは検索を容易にするようにデータを蓄積管理することであり，そのための手法をデータ管理構造と呼ぶ。一般に大量データの管理においてデータ管理構造として木構造が用いられ，100 万個のデータから対象データを数十回程度の比較演算で検索することができる。

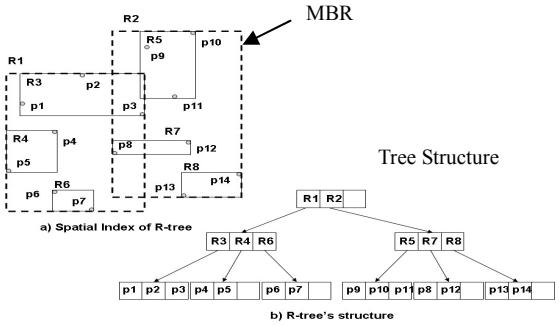


図１　R-tree の概略図

表１　木構造性質の比較

	k-d tree	quad-tree	R-tree	kdm-tree
管理方式	領域分割	領域分割	データ包括	領域分割
バランス	×	×	○	○
最大子ノード数	2	4	M	M
最悪メモリコスト	○	×	○	○

空間データの管理方式には全空間を分割して階層的に管理する領域分割形と，データを囲む最小長方形 MBR で管理するデータ包括形がある。静的データに対してデータ包括形の R-tree (図 1)が広く用いられてきた。しかし，R-tree で動的データを管理する場合は，移動オブジェクトの位置変化に伴って MBR を変化させる必要があるため処理コストが大きくなる。例えば，図 1 で p14 が右方向に移動した場合，R8 と R2 を拡大する必要がある。一方，領域分割形の手法は，データの位置変化に伴う領域変化は少ないが，削除処理性能が十分でない。

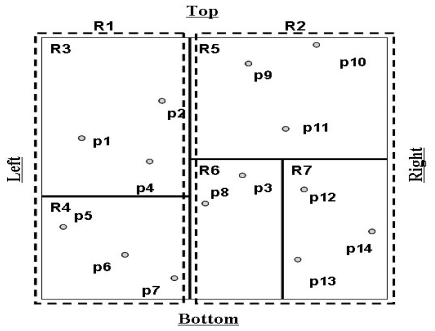
更新管理においてデータ管理構造に求められる要件として以下の 3 つが考えられる。

(1) データ包括形でなく領域分割形であること
(2) バランスする M 分木で，最悪メモリコストが保証されること
(3) 削除処理コストが小さい構造であること

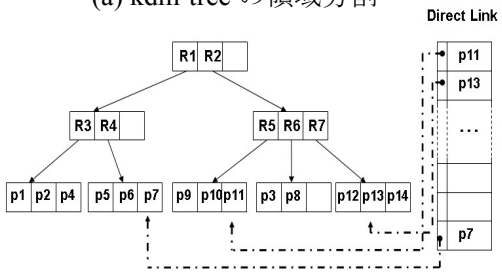
バランスとは根ノードからすべての葉ノードまでの距離が等しいことを指し，M 分木とは内部ノードが最大 M 個の子ノードを持つ木構造を指す。最悪メモリコストの保証とはデータ数によって木構造が使用する記憶領域の最大値が決まることである。a)～c)の要件を備えることで，挿入・削除時の木構造変化を抑えて更新を高速化し，検索性能を高く保つ。

表 1 は代表的な木構造の性質を比較したものである。k-d tree[2]と quad tree は領域分割形であ

*1 埼玉大学大学院理工学研究科数理電子情報系専攻 博士前期課程 2 年
*2 埼玉大学工学部電気電子システム工学科

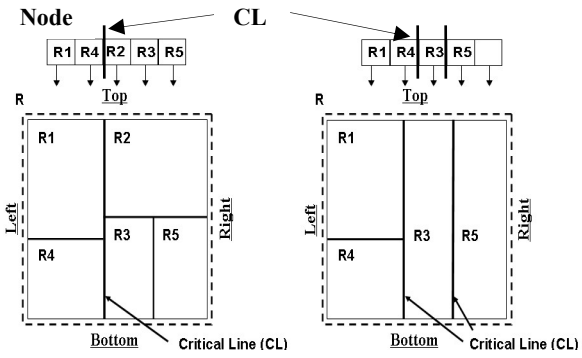


(a) kdm-tree の領域分割



(b) kdm-tree の木構造

図 2　kdm-tree の概略図



(a)kdm-tree の例 1　　(b)kdm-tree の例 2

図 3　kdm-tree の管理長方形

るが，バランスしないため不均一なデータ分布で検索性能が劣化する。R-tree や R-tree を拡張した RBU-tree[3]はデータ包括形であるため，データ位置変化に伴う木構造の変化が避けられない。また領域分割形で最悪メモリコストを保証する手法は削除コストが大きいか，もしくは削除により検索性能が劣化する問題がある。

４．提案手法　kdm-tree

kdm-tree(k-d tree for moving objects)は領域分割形の k-d tree をバランス M 分木に拡張し，削除性能に優れた新たなノード分割方式を用いる。

図 2 に kdm-tree の概略図を示す。ここでノードの管理長方形を 2 等分する線分を Critical Line (CL)と定義する。kdm-tree の各ノードは必ず 1 つ以上の CL を持つ。ノード分割時に CL を用いて 1 つのノードを 2 つに分割することで，兄弟ノードが管理する長方形の隣接性が高まり，削除時に兄弟ノードどうしが統合しやすいバランス M 分木となる。

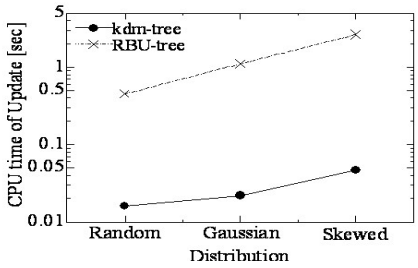


図 4　更新時間と分布

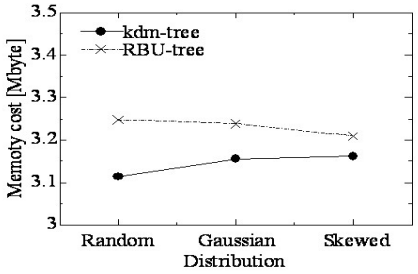


図 5　メモリコストと分布

図 3 に kdm-tree の管理長方形の例を示す。図 3 の内部ノードは最大 5 つの子ノードを持つとする。図 3(a)においてノード分割が生じた場合，{R1,R4}と{R2,R3,R5}の 2 グループに分けることでノードを分割する。図 3(b)のように CL が複数ある場合，分割時の分配バランスが良い CL を選ぶ(ここでは R3 の Left 辺)。

kdm-tree は隣接長方形を拡大することで統合を行う。図 3(b)は図 3(a)における R2 を削除した場合である。R3 が削除される場合，隣接する R3 と R5 を拡大して領域管理を行う。

５．計算機実験

10 万個の移動オブジェクトを二次元空間に分布させて更新を行う。データ分布を一様分布，正規分布，偏在分布としたときの更新時間を図 4 に，1000 回更新後のメモリコストを図 5 に示す。kdm-tree の更新時間は RBU-tree に比べ 1/25 以下となり，メモリコストは 2~4%程度少なくすむ。また検索性能も同等以上の性能であった。

６．まとめ

本論文では移動オブジェクトの更新に適したデータ管理構造に必要な要件を挙げ，それらを満たす領域分割形木構造 kdm-tree を提案した。

文 献

[1] A. Guttman, "R-tree: A dynamic index structure for spatial searching", Proc. ACM Special Interest Group on Management of Data, pp.49-59, 1984.

[2] J. L. Bentley, "Multidimensional binary search trees used for associative searching", Commun. ACM, 18, 9, pp.509-517, 1974.

[3] M. L. Lee, W. Hsu, C. S. Jensen, B. Cui and K. L. Teo, "Supporting frequent updates in R-trees: A bottom-up approach." In Proceedings of VLDB, 2003.