Index Structure for Managing Multi-levels of Road Networks on Distributed Environment

Jun Feng
Graduate School of Engineering
Nagoya University
Furo-cho, Chikusa-ku, Nagoya
Aichi 464-8603, JAPAN
feng@watanabe.nuie.nagoya-u.ac.jp

Toyohide Watanabe
Graduate School of Engineering
Nagoya University
Furo-cho, Chikusa-ku, Nagoya
Aichi 464-8603, JAPAN
watanabe@nuie.nagoya-u.ac.jp

Abstract

With a view to attaining the sharability and consistency of map information under distributed environment, we proposed a Multi-level/Multi-theme map information model to maintain maps in consistency with original source datasets. However, the distributed management of spatial datasets results in a complex maintenance processing, especially when the modification refers to several datasets. To solve this problem effectually, in this paper we propose an index structure, MOR-tree (Multi-levels-Object-Relation tree), for organizing integrated maintenance procedure. MOR-tree is an extension of R-tree index structure with the ability of indexing spatial objects of multi-levels in one hierarchy and records relations among objects at different levels. The performance of MOR-tree is also evaluated with a prototype system in this paper.

1. Introduction

Map information is the basis of Geographic Information System (GIS). For a country-wide GIS, there is a need for consistent map information of many themes at different levels of detail, even though the information is collected and maintained by different authorities. This issue is related to the integrated management of information of multiple datasets on different scales and the multi-representation of spatial objects.

Because the generalization of map information cannot be realized automatically [7], based on a trade-off between storage and computation, many models have been proposed [5, 8] to manage multi-scales of map objects. The model of [5] assures that maps can be displayed customarily by storing spatial information of entities for every scale. However, to keep information contents consistent among multiple scales is complicated. A hierarchical structure for managing map objects at different scales is proposed in [8] for rapid map zooming. Because the structure is too simple to process many relations among maps of different scales, the maps cannot be displayed customarily.

In view of the multi-representation of spatial objects, a number of so-called reactive data structures have been proposed from the point of geometry generalization: e.g., Reactive-tree [7] and PR-file [1]. A Reactive-tree assigns an *importance* value to every geometric object on one most detailed map, and objects are stored in proper levels of the tree according to its importance value. It is effective to access objects with different importance values. However, the different appearances of one object depend only on the "on the fly" drawing algorithm and the relations among objects with different importance values are ignored. The PR-file is also based on one most detailed map. In a PR-file, one geometric object is broken into multiple objects and each object is assigned with a priority value by making use of a line simplification algorithm. Each priority value corresponds to a map scale. For each scale, objects which have higher priority values are not retrieved. However, in order to achieve its desirability, the overall structural design as well as insertion and deletion algorithms is rather complex [2].

So, we proposed Multi-level/Multi-theme (M^2) map information model [3] to support extraction of information satisfying a given level of detail for a specific region or themes. Our model is powerful to integrate various scales of maps uniformly. However, there are more kinds of relations among levels than those coped by Reactive-tree or PR-file. To achieve an efficient management of these datasets, in this paper we propose

0-7695-1916-4/03 $17.00 © 2003 IEEE

Multi-levels Object-Relation structure (MOR-tree) for indexing information of multi-levels of road networks on a distributed environment.

The rest of this paper is organized as follows. Section 2 describes our M^2 map information model. Section 3 focuses on MOR-tree. Section 4 outlines a prototype system and presents the experimental results of performance evaluation. Finally, Section 6 concludes our work.

2 Framework

M^2 map information model is proposed to manipulate countrywide integrated maps of different scales for different themes. The model can be regarded as a forest consisting of two kinds of hierarchies: one is a directory tree, which is obtained by recursively decomposing map regions into a sequence of increasingly finer tessellations with regard to the granularities of administrative units; and another is theme trees, which are obtained by uniquely dividing spatial objects into different themes and then in every theme, by dividing them into different scales in regard to the important level, ownership or display needs of them [9].The information model is powerful to integrate various scales of maps uniformly and possesses advanced extensibility [3].

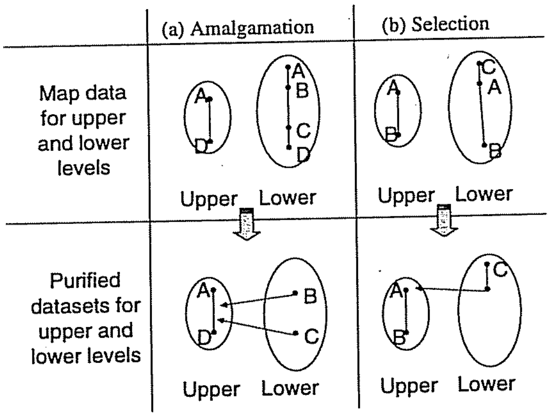

Figure 1. Purification

Under this model, map objects of a theme are assigned to multi-level datasets without repetition, the dataset in the lower level is a supplement of the dataset in the upper level. For a particular requirement, map information can be prepared dynamically by using generation functions. For example, to generate a city-level road map, we can combine the information inside the region managed in country-level and prefecture-level datasets with the city-level datasets.

Typical situations of this assignment for road objects in two levels are given in Figure 1. In Figure 1(a), the relation between the map objects in upper and lower levels is amalgamation: map object AD is represented as a line (AD) in the upper-level map and as lines (AB, BC, CD) in the lower level— an object in the upper level can be regarded as the merging of adjacent objects in the lower level. The information about nodes A and D is the same in both levels and the relation between A and D is different. So, under our model the information in the lower-level map can be split into two parts: one part is the same information as that in the upper level; and another part is the information which only belongs to the lower level. These two parts are called purified datasets, and are managed in the upper and lower levels, respectively. Considering the relations between two datasets are only needed by the process of generating maps of the lower level, these relations are managed in the purified dataset of the lower level. In (b), there is a selection relation between the maps in the two levels: map object AB is displayed in two levels, but AC is not needed in the upper level— the information in the upper level is a "selection" of that in the lower level. Therefore, the information in the purified datasets of the upper and lower levels is object AB and object AC, respectively.

3 Multi-levels of Object-Relation Index Structure

In M^2 model, map objects are split into purified datasets of multiple levels without redundancy. For the perfect sharability and consistency of these datasets, the integrated maintenance of them needs to be considered: i.e., modifications on a map maybe result in a series of operations— modifying of the objects in every referring datasets and resetting of the relations among them if needed. Therefore, an efficient access method, called Multi-levels of Object-Relation Index Structure (MOR-tree in short), for these datasets and the relations among them is proposed.

3.1 Design Overview

Besides the ability of accessing spatial objects in one level (in one dataset) efficiently just like other spatial index structures (e.g., R-tree [4]), MOR-tree is designed to be able to differentiate the levels of objects and manage the relations among objects of multiple levels.

To differentiate the levels of road objects, a main hierarchy is proposed. Similar to Reactive-tree [7], the main hierarchy assigns a logical importance value to every object. The logical importance value is a natural number in agreement with the map level: e.g., 0 for objects in country-level datasets, 1 for those in prefecture-level datasets, and so on. The objects with the higher importance values are stored in the higher levels of the hierarchy. Another kind of hierarchy, called a composition hierarchy, is proposed to keep the relations among levels, which are pointed by the leaf nodes of the main hierarchy. The main hierarchy is based on R-tree, to achieve the outstanding spatial access performance. Each node in the hierarchies contains a number of entries. There are three kinds of entries: tree-entries, object-entries and composition-entries. The internal nodes may contain the first two kinds of entries, in contrast to R-tree. The leaf nodes contain object-entries. The object-entry points to a composition hierarchy, which consists of composition-entries. Three kinds of entries have the following forms:

1) Object-entry has the form (*MBR*, *flag*, *comp-ptr*), where *MBR* is the minimal bounding rectangle of the composition hierarchy; *flag* is a natural number that indicates the importance level; and *comp-ptr* contains a reference to a composition hierarchy;

2) Tree-entry has the form (*MBR*, *flag*, *child-ptr*), where *child-ptr* contains a reference to a subtree. In this case *MBR* is the minimal bounding rectangle of the whole subtree and *flag* is the least importance level of the child-node.

3) Composition-entry has the form (*comp-id*, *n-ptr*, *nl-ptr*), where *comp-id* is the identifier of object's composition; *n-ptr* contains a reference to the next composition of the parent node object: e.g., the object is a road segment, the first composition of it is one of the end points, *n-ptr* points to the next point on the same object; and *nl-ptr* contains a reference to the composition of the parent node object in the lower level: e.g., the intersection between the upper-level road and lower-level road.

3.2 Creation Process

The index is created based on datasets of multiple levels. When a map is generated for a region on specific level of scale, the indexes are created on the referring datasets. Here, we give the algorithm for two levels:

1) Create bottom nodes of main hierarchy based on the objects in the lower level;

2) Create composition hierarchy for objects from the upper level: *nl-ptr* points to the intersection node, which is the intersection between upper-level and lower-level roads;

3) Create main hierarchy based on the nodes generated by 1) and 2).

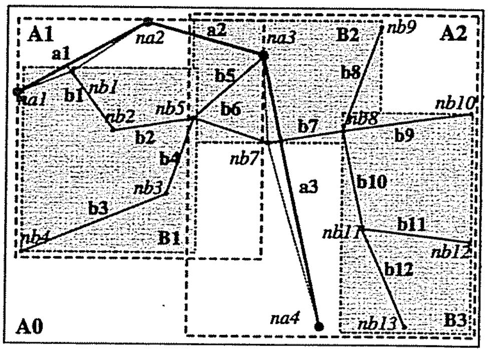

Figure 2. Example of two-levels of road networks and MBRs for road segments

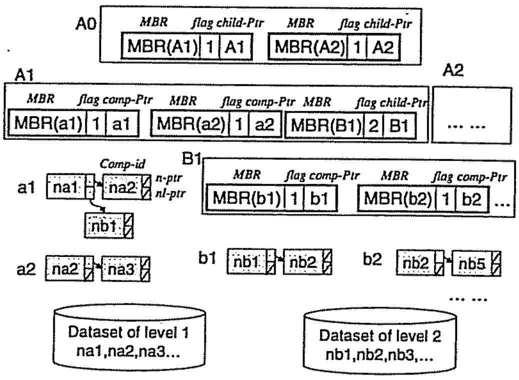

Figure 3. Index structure for two-levels of road networks: Ai: internal nodes of main hierarchy; Bi: leaf nodes of main hierarchy; ai, bi: composition hierarchy; nai, nbi: Composition-entries; i(1 or 2): flag of Object-entries or Tree-entries

An example for two-levels road network is given in Figure 2. The bold lines (a1, a2 and a3) represent the road segments managed in the upper level, and lines b1-b12 represent the road segments managed in the lower level. The dotted lines between *na1* and *nb1*, *nb1* and *na2* are generated by the inheritance function, which propagates a1 to the lower level and splits the original road segment into two ones. a1 is the road segment managed in the upper level. The dotted lines between *na3* and *nb7*, *nb7* and *na4* are so for a3. The index for the road datasets in Figure 2 is given in Figure 3, and *MBR*'s generated for the index are illustrated with dotted boxes(for simplicity, in the example the maximum number of entries in a main-hierarchy node is 4) in Figure 2. At the step 1, we generate bottom nodes B1-B3 based on the objects in level 2 (the gray boxes in Figure 2); At the step 2, the composition hierarchies corresponding to objects in level 1 are created. MBR for every object-entry is based on all the objects inside the corresponding composition hierarchy.

3.3 Example of Update

There are importance flags in object-entries and tree-entries: the index structure provides an integrated way to access datasets in different levels; the relations among levels are represented in the composition hierarchies; and the simultaneous modification to multiple levels is possible.

We give an example of deleting the road segment between *nb1* and *na2*. The operation refers to datasets in two levels. Before the update, *nb1* belongs to the level 2, and *na1* and *na2* belong to the level 1. The road segments *na1-nb1* and *nb1-na2* are generated in the level 2; however, after the update, as the node *nb1* becomes an end-point of the upper-level road, the road segment managed originally in the upper-level a1 (*na1-na2*) should be replaced by *na1-nb1*, and the information about *nb1* should be managed by the upper level. The update steps are given as follows:

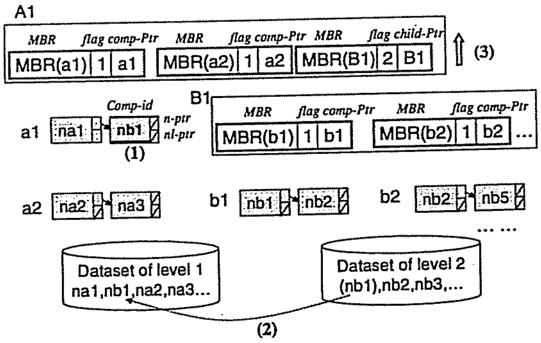

Figure 4. Update on index structure: (1) replace na2 with nb1; (2) copy information of nb1 from level 2 to level 1; (3) reset pointers and adjust main hierarchy.

(1) Descend the main hierarchy, locate the referred object-entry of object in upper level. Locate the modification on the composition hierarchy, replace *na2* with *nb1* in the composition hierarchy (see Figure 4);

(2) Modify the corresponding datasets. Copy the node *nb1* from level 2 to level 1, and make up the original one as a virtual node, which points to the real one in level 1;

(3) Adjust the main hierarchy.

4 Experimental Evaluation

In this section, we introduce our prototype system and evaluate the index structure proposed for maintenance.

Our prototype system is developed in Java on an SGI O2 R5000 SC 180 entry-level desktop workstation. The system manages themes of country, prefecture and city levels, based on the maps of Japan (country), Aichi Prefecture and Ichinomiya City. We have also developed the inheritance function, overlay function and query function on the road theme and the architecture theme. There are 30,663 road segments managed in the system for representing 34,235 road segments in the real world. These road segments are assigned to the country-level (country-wide highways and national roads), the prefecture-level (prefecture roads and main local roads) and the city-level (city-roads).

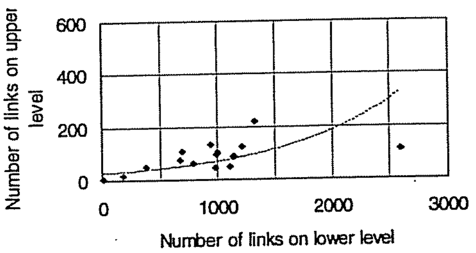

Figure 5. Data distribution in the upper level and lower level of experiment datasets (dotted line is an approximate curve for data distribution)

To evaluate MOR-tree, we compare the performance of this index structure with that of multiple datasets arranged by R-trees in every level, respectively. The comparisons are based on the time of creating index structures for multi-levels of datasets and the modification time for multi-levels of datasets based on these index structures. As the performances are related to the data distributions among multiple levels, the experiments are based on some subsets of data in our prototype system. The approximate curve in Figure 5 represents the data distribution of some datasets among multiple levels.

1) *Creation time for index structures:* To create an MOR-tree index structure for the road datasets in our prototype systems, the road objects in the lower level are arranged in the bottom level, and the objects in the upper levels are arranged in the upper levels in the tree. We use the Sort-Tile-Recursive algorithm [6] for packing the trees. This algorithm clusters rectangles in an attempt to minimize the number of nodes visited while processing a query. As the links in the lower level are much more than those in the upper levels, the creation time of MOR-tree is shorter than that of R-trees (Figure 6).

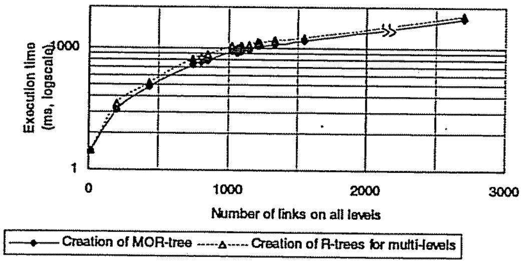

Figure 6. Creation time of two index structures

2) *Modification to multi-datasets:* As the relations among multiple levels are managed by MOR-tree, the modification to multi-levels of datasets can be realized integratedly without searching the relations among them from the datasets. The delete operation similar to the example in Section 3.3 was demonstrated, and the result is given in Figure 7. We can observe that the MOR-tree outperforms R-tree strucutures in applying modifications to several related datasets.

5 Conclusion

In order to attain the information sharability and consistency, we proposed the framework for managing multi-levels of road networks in distributed environments. The method is powerful to integrate map information in various levels uniformly in comparison with

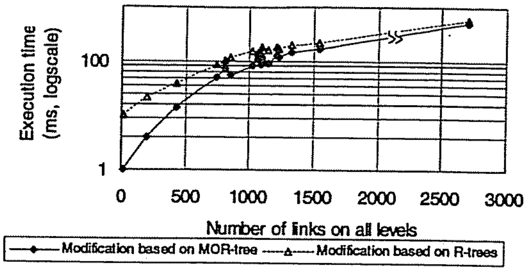

Figure 7. Modification time to multi-levels

those in the traditional GIS. For the integrated maintenance of the map datasets, we proposed an index structure called MOR-tree.

We have successfully assigned the road networks of Ichinomiya city of Japan to the road theme among three different levels in our prototype system with inheritance function, and evaluated the performance of MOR-tree.

For our future work, other map themes with complex relations among levels would be investigated.

References

[1] B. Becker, H.-W. Six, and P. Widmayer. Spatial priority search: an access technique for scaleless maps. *Proc. of ACM SIGMOD'91*, pages 128–137, 1991.

[2] E. Chan and K. Chow. On multi-scale display of geometric objects. *Data and Knowledge Engineering*, 40(1):91–119, 2002.

[3] J. Feng and T. Watanabe. Effective representation of road network on concept of object orientation. *The Transactions of IEEJ*, 122-C(12):2100–2108, 2002.

[4] A. Guttman. R-trees: A dynamic index structure for spatial searching. *Proc. of ACM SIGMOD'84*, pages 47–57, 1984.

[5] Y. Leung, K. Leung, and J. He. A generic concept-based object-oriented geographical information system. *Int'l J. of Geographical Information Science*, 13(5):475–498, 1999.

[6] S. T. Leutenegger, J. M. Edgington, and M. A. Lopez. Str: A simple and efficient algorithm for r-tree packing. *Proc. of ICDE'97*, pages 497–506, 1997.

[7] P.V.Oosterom. The reactive-tree: a storage structure for a seamless geographic database. *Proc. of Auto-Carto*, 10:393–407, 1991.

[8] S. Timpf. Hierarchical structures in map series. *http://www.geoinfo.tuwien.ac.at /publications /former-Personnel /timpf /diss /table-of-contents.htm*, 1998.

[9] T. Yamanashi and T. Watanabe. Multi-layers/multi-phases model adaptable to integrated gis. *Proc. of VSMM'00*, pages 668–676, 2000.