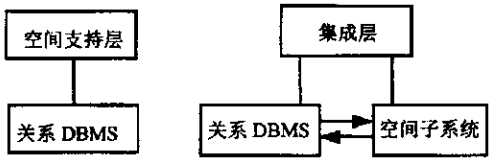
SADBS 的空间数据库设计与索引组织*

李　萍
(盐城工学院 计算机工程系,江苏 盐城　224003)

摘　要:空间分析系统是比较耗时的一种系统,而存储介质是制约系统速度的一个重要原因,在主存中组织数据库并将空间数据索引也建在主存中可以极大地改善系统性能。详细介绍了我们设计的空间分析数据库系统 SADBS 的空间数据库存储结构及其索引组织。

关键词:SADBS;空间数据库;空间索引;R 树;Realms
中图分类号:TP311.13　**文献标识码**:A　**文章编号**:1671 - 5322(2004)01 - 0041 - 05

空间数据库是一个存贮空间和非空间数据的数据库系统,早期的空间数据库是建立在关系数据库上的,有对偶结构(Dual)和分层结构(Layered)[1],如图 1 所示。



(a) 分层结构　　(b) 对偶结构
图 1　早期的空间数据库结构

Fig.1　The structure of Early Spatial Database

其中分层结构是直接在关系 DBMS 上加一个处理空间数据的模块——空间支持层,系统在空间支持层中处理空间数据,其主要缺点是在关系 DBMS 层和空间支持层之间数据必须作频繁的格式转换,致使系统效率降低;对偶结构用一个独立的空间子系统来存储空间数据,而用 DBMS 存储、处理非空间数据,然后在其上进行集成,空间数据和非空间数据通过逻辑指针相联系,避免了大量数据格式的转换,很多商业化的 GIS 系统(ARC/INFO,SICAD)和一些原型系统研究中采用了这种方法,其主要缺点是查询时对象必须分解成两部分,这样势必影响系统的查询效率,也难以进行全局优化。

目前有很多数据库管理系统(如 AMOS,POSTGRES)支持扩充系统的数据类型和数据操作的技术,利用这种现有条件,可以有效地实现空间子系统和可扩充数据库技术的结合。我们正在研制的空间分析数据库管理系统正是采用了这种集成结构(Integrated),将空间子系统通过扩充接口集成到特定可扩充数据库管理系统(POSTGRES),作为可扩充 DBMS 中的空间数据类型的操作集合提供给建立和运行 GIS 系统的上层用户。

1　SADBS 系统中空间数据库的结构设计和实现[2]

SADBS 系统是基于 Realms[3,4] 和主存数据库技术的具有强大空间分析功能与管理功能的空间子系统,它实际完成空间数据的组织和约束、空间分析算法的实现、存储管理、数据的编辑和使用,空间索引的建立等功能。SADBS 系统引入了 Realms 概念,以此为基础进行数据组织和约束,从而降低了空间分析实现的复杂程度,提高了系统的分析效率。

1.1　基于 Realms 的 SADBS 系统的空间对象

SADBS 存储子系统中的空间数据建立在由 R.H.Güting 教授等在 1993 年提出的 Realms 概念之上,它是一种用于描述空间平面的数学模型,它具有如下性质:

(1)点集中的每一个点和每条线段的端点都是 Realms 中的一个网格点;

* 收稿日期:2003 - 12 - 08

作者简介:李萍(1970-),女,江苏建湖县人,硕士,盐城工学院讲师,主要研究方向:空间数据库及其应用。

(2)没有一个 Realms 点位于任一条 Realms 线段的中间;

(3)除了端点外,任何两条 Realms 线段都不相交。

在实际应用中,一个空间对象的所有空间线段不可能不相交,此时,必须将交点的坐标调整到网格点上,并将其插入 Realms 的点集中,同时依据交点将线段进行分裂,使整个点集合和线段集合仍符合 Realms 的规范。

SADBS 系统中空间对象可分为点、线和区域三种类型,分别表示为 Points、Lines 和 Regions,每种空间对象的元素集合由各自的基本空间元素组成:Points 类型空间对象的元素集合由 Point 类型空间元素构成;Lines 类型和 Regions 类型的空间对象的元素集合由 Segment 类型空间元素构成。对应现实世界中的实例分别有:导游地图中某地区的景点分布可用点类型空间对象 Points 表示;地形分布图中的河流可用线类型空间对象 Lines 表示;一个地区的行政区域可用 Regions 表示。

每个空间对象都由基本空间元素构成。空间对象的存储内容包含:属于对象本身的空间元素的集合和描述空间对象内部拓扑结构特征的信息。结构特征信息通过对空间对象的元素集合进行分析得到,包含了两类内容:一部分用于描述空间对象的整体特征,如空间对象覆盖区域的直径 diameter,包含空间对象的所有空间元素的最小边界矩形 BBox,区域类型空间对象的周长 perimeter,即构成区域类型对象的所有线段的累计长度等;另一部分用于描述空间对象内部的更详细的拓扑结构特征,如线类型对象中包含的分块 block,区域类型对象中包含的圈 cycle 和面 face 等。

1.2 SADBS 系统的空间数据库组织

1.2.1 主存和磁盘的访问特性

计算机的主存和磁盘具有不同的访问特性,这些特性将影响着数据库系统的设计和性能。与磁盘相比,主存有下列优点:

(1)主存的存储速度远快于磁盘的存取速度,他们的存取时间上有数量级的差别,这是主存的最重要的优势;

(2)磁盘是块存储设备,以块为单位进行编址,而主存是随机存储设备,以字或字节编址,磁盘的存取必须通过主存缓冲;

(3)磁盘访问时有一个固定代价——寻址时间,因此数据在磁盘上的组织很重要,如果能将相关的数据放入相邻的磁盘块中,则可减少寻址时间;而主存由于允许随机存取,访问时间与数据的组织基本无关。因此数据在磁盘上的存储结构对系统性能的影响远比在主存中数据结构对系统的性能影响要大。

在其他访问特性上,主存比磁盘要脆弱一些:

(4)主存通常是易逝的,磁盘是永久性存储;

(5)主存通常可以由处理器直接存取,而磁盘则不然,因而主存数据比磁盘数据更易受到由于软件错误所导致的数据破坏。

在主存数据库中,主存中的数据作为主拷贝。这也就意味着在对数据的组织和存取上,和外存中的有着根本的不同,在主存中组织数据库将会享有很多优势,并且面临新的技术问题。

1.2.2 在主存中组织空间数据的存储

空间数据库可以看作是空间对象的集合,为了提高对系统的核心数据的检索效率,SADBS 系统引入了主存技术,将空间数据库的主备份放在主存中,并对其采取符合主存特点的组织和管理方式。在创建或装载空间数据库的时候,在内存开辟一块符合原数据库大小的连续的内存区域,用来连续存放数据库头和数据库体两部分,如图 2 所示。

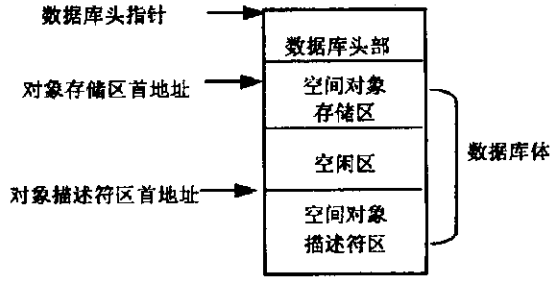


图 2 空间数据库的结构示意图

Fig.2 The Structure of Spatial Database

数据库体中的内容可以分为三部分:空间对象存储区,空间对象描述符区和空闲区。空间对象存储区用来存储各种空间对象的数据结构,由于各个空间对象的大小是不相等的,为了管理方便,引入了长度固定的空间对象引用标识,即空间对象描述符,它们存储在空间对象描述符区域,该区域可以看作是定长元组组成的数组,而空间对象描述符元组在对象描述符数组中的下标称为空间对象序列号,它可以唯一标识空间对象。从图 2 中可以看出空间对象存储区和空间对象描述区是相向增长的。数据库头部用来对数据库的总体进行

描述和控制，其中存储了空间对象存储区域和空间对象描述符区域以及空闲区的起始地址。

在实际操作中，空间对象序列号是按如下方法获得的：当空间对象建立后，便可利用 createSpObject()函数将其插入到数据库中，首先会根据对象实际大小及系统规定的预留空间检查库中是否尚有足够空间，若有，则将对象插入库中，一旦对象正确插入，则此函数将空间对象实际占用的存储空间大小及存储空间的偏移量作为新对象的描述符元组的相关内容记录下来，并返回新元组号，该元组号即空间对象序列号。然后以此序列号为参数调用 getSpObject()函数，就可以获得实际存储空间对象的指针，这样就可以找到真正的空间对象并得到该对象最小边界矩形 MBR 信息(建立索引时需要)。

数据库头部和存储空间的管理相关的主要数据结构如下：

```
struct mainMemDB {
    ……
    int memSize;/* 数据库体占用的内存大小 */
    int usedEnd;/* 空闲区起始地址，初值 0 */
    int itNum;/* 所有对象描述符的数目，包括被标记为删除的描述符，初值 0 */
    int deleteList;/* 删除链表，空间对象标识为删除但内存不释放，初值 0 */
    int deleteEnd;/* 删除链表尾指针，初值 0 */
    int deleteCount;/* 删除链表中的元素数目，初值 0 */
    int availableList;/* 可用链表，空间对象标识为删除且内存已释放，初值 0 */
    int availableEnd;/* 可用链表尾指针，初值 0 */
    int availableCount;/* 可用链表中的元素数目，初值 0 */
    char mem[MaxDBSize];/* 数据库体指针 */
};
```

空间对象存储区首址就是 mem[1]单元的地址，mem[memSize]所指即空间对象描述符区头部的位置，这两个区域是相向增长的。空间对象描述符是长度相等的一系列元组，每个对象描述符主要包含下面内容：

```
struct objIterator {
    ……
    int size;/* 空间对象的存储空间大小 */
    int offset;/* 空间对象相对于空间对象存储区起始地址的偏移量 */
    objType type;/* 空间对象的类型，点、线和区域类型中的一种 */
    boolean avail;/* 标识该空间描述符对应的空间对象是否被删除，TRUE 表示未删除 */
    int next;/* 用于将描述符组织成链表 */
};
```

1.3　SADBS 系统空间数据库的操作

空间数据库作为对象集合，有许多相关信息需要管理，包括对象的增加、删除和修改。SADBS 系统在空间对象的插入、删除和修改等过程中完成空间的分配和使用等工作。下面简要介绍数据库常用操作的实现方法(限于篇幅，源程序略)。

1.3.1　空间对象的增加 CreateObject

插入空间对象前，先要在系统的工作变量区建立空间对象，在此过程中，空间对象的所有元素和整个数据库的相关空间元素都必须按 Realms 的要求调整完毕。

向数据库插入空间对象，实际上要在数据库中建立空间对象描述符和它的存储空间并将空间对象从工作变量区拷贝到这个存储空间中。考虑到在实际编辑中经常会对已存在的空间对象进行修改，因此在初次建立空间对象的时候，为其预留一部分空间(称为预留百分率，取决于空间对象的大小)。插入空间对象的过程可以描述如下：

(1)若数据库的空闲空间小于空间对象实际需要的大小，将引起数据库的增长，如果数据库无法增长则报告错误并结束此次插入过程；

(2)根据空间数据库的内容产生这个对象的空间对象序列号；

(3)在数据库的空闲区分配该对象实际需要占用的存储空间，将存储空间的偏移记录到描述符元组中，调用空间对象拷贝过程将工作变量区的空间对象拷贝到该存储空间中；

(4)返回新空间对象的空间对象序列号；

1.3.2　空间对象的删除 DeleteObject

删除一个空间对象时，给定该对象的空间对象序列号，直接定位该对象的描述符，具体删除过程如下：

(1)将该对象描述符的有效位置为无效(avail = FALSE)；

(2)将该对象描述符连接到 deleteList 链表中；

(3)将数据库头部的数据库整理标识设置为未整理。

1.3.3　空间对象的更新 ModifyObject

更新一个空间对象时，实际上是在内存先建立更新后的对象，然后用这个对象替换数据库中的对象，主要算法思想如下：

(1)若原来对象的空间足够容纳新对象，则调用拷贝过程将新对象复制到数据库中并返回，否则执行下面动作：

(2)为更新后的对象创建新的描述符和空间，

将新对象的内容拷贝到这个空间中；

(3)将新描述符和原来的描述符的内容交换，删除新对象描述符所指对象(即删除原对象)。

2 SADBS 系统索引结构设计和存储管理

空间索引是依据空间对象的位置和状态按一定顺序排列的一种数据结构，包含空间对象的概要信息如对象的标识、边界矩形及指向空间对象实体的指针，空间索引技术对于提高空间数据库中空间数据存取效率至关重要，它能使空间操作迅速访问操作对象，减少搜索范围，提高效率。

R 树类索引是目前空间索引技术的研究热点，索引结构类似于 B 树，是一维 B 树在空间数据上的扩展。R 树和 B 树在插入算法上很相近，只在删除算法上有些差别：B 树在删除时若发生“下溢”，要进行‘合并’结点的操作，而 R 树则将发生‘下溢’的结点及因此而导致的发生‘下溢’的父结点全部删除，重新按插入算法将结点插入到树中。最先出现的 R 树类索引结构是 1984 年由 Guttman 提出的 R 树[5]，此后人们对它的算法和结构进行了一些改进，不过大多数都保持了与原 R 树相同或类似的结构特点。比如 R + 树就是为了避免 R 树中间结点外接矩形的交叠(overlap)而提出的，它的主要改进思想是 如果允许在动态划分结点空间时，劈开中间结点的矩形空间，那么就可以在中间结点的子结点间去掉交叠，这样可以减少无效查找，从而提高效率，但同时却又带来了叶结点的冗余，并且随着数据库内容的增加，冗余有递增的趋势，难以控制。为了充分发挥主存数据库技术的优越性，提高系统性能，在 SADBS 中采用了主存 R 树索引方法对空间对象集合进行组织，实现了 R 树索引的创建、插入、删除、更新及查询等操作。

2.1 空间对象的索引组织

以 R 树索引方法对空间对象集合进行组织时，每个空间对象都是 R 树叶结点中的一个数据项，R 树按照空间对象的最小边界矩形 MBR 来组织空间索引，不同类型对象的最小边界矩形分别通过相应的函数 Points — GetBBox()，Lines — GetBBox()和 Regions — GetBBox()来获得。

2.2 主存 R 树索引的存储结构

和数据库文件类似，索引文件也在内存中连续的存储空间内存放，分为索引文件头部和索引文件体两部分。索引文件头部用来对索引文件的总体进行描述和控制，索引文件体则用来按 R 树的方式组织空间对象。如图 3 所示：

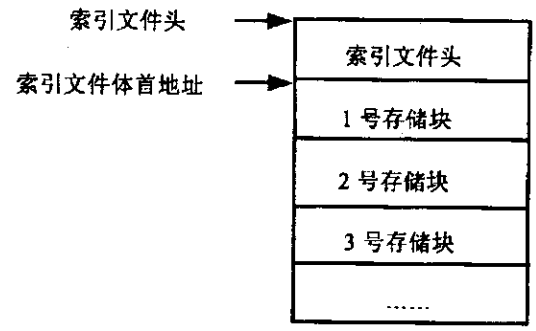


图 3 R 树索引文件的存储结构示意图

Fig.3 The Storage Structure of R tree Index File

索引文件体部分划分为长度相等的若干个存储块，并对这些存储块从 1 开始进行编号，每个存储块用于存储一个 R 树结点信息。R 树叶结点中的数据项包含了空间对象序列号的定义，通过该序列号，可以得到对象偏移量 offset，然后直接定位到对象所在的位置。当数据库中增加或删除空间对象时，修改相应的 R 树结点信息，并根据需要分配或回收相应的存储块。

索引文件头部的主要数据结构如下：

```
typedef structure mainMemIDX {
    int rootPage；/ * 存放 R 树根结点所在存储块号，初值为 1 * /
    unsigned char modified；/ * 修改标志，若索引文件被修改则置为 1，初值 0 * /
    unsigned maxIdxItems；/ * 索引项数，即 R 树中结点个数，初值 0 * /
    int DeleListHeader；/ * 删除列表头指针，存放表中第一个被删除结点所在的存储块号，初值为 - 1，后续表项由前一个被删结点的子结点指针指向，从而形成一删除列表 * /
    unsigned int DeleNum；/ * 已删除结点个数，初值 0 * /
    ……
} mainMemIDX；
```

R 树结点的结构定义如下：

```
typedef struct R — treeNode {
    int RecNo；/ * 结点所在存储块号 * /
    int DeleFlag；/ * 结点删除标记，初值为 0，若结点已被删除，则赋值为 1 * /
    int level；/ * 结点所在层号，叶结点为 0，其上每层结点依次增 1 * /
    int numEntries；/ * 结点中所包含的元素个数，对中间结点，表示包含的子结点数，对叶结点则表示包含的空间对象数 * /
    Entry e[MAX — NUM — OF — ENTRIES]；
} R — treeNode；
```

Entry 结构的具体定义如下：

```
typedef struct {
```

```
    Rect r;/* 空间对象的最小边界矩形 */
    union {
        unsigned int child;/* 非叶结点时,指向一子结点 */
    Object o;/* 叶结点时,定义的是空间对象序列号 */
    }ref;
}Entry;
```

2.3　索引空间的存储管理

2.3.1　索引项空间的分配

当向数据库中插入一个新的空间对象时,如果对象所应插入的 R 树叶结点中已没有足够的空间存放下此空间对象时,该叶结点要发生分裂,此时即需要申请一个新的存储块以建立相应的索引项,若父结点也因此而发生分裂,则该申请过程继续进行直至根结点。

索引项空间分配的过程描述如下:

(1) 首先检查删除列表是否为空,若为空,则在索引文件尾申请一个存储块,并调整相关信息;若已经没有可申请的存储块,则需在内存重新申请一片更大区域的连续存储空间来存放该索引文件,并在新的空闲区域中申请存储块;

(2) 若删除列表不空,说明曾经有过结点的删除,此时将删除列表头所指向的存储块分配给索引项,同时调整删除列表头及相关信息。

2.3.2　索引项空间的回收

当从数据库中删除一个或若干个空间对象时,应相应地修改索引文件,即从相应的 R 树叶结点中删去该对象,若因此引起叶结点的下溢,则应将该叶结点所在存储块进行回收。

索引项空间回收的过程描述如下:

(1)首先检查删除列表是否为空,若删除列表为空,则将被删除的索引项所在存储块设置为删除列表头,修改删除列表项数,并调整相关信息;

(2) 若删除列表不为空,则找到当前删除列表头,并将被删除索引项所在存储块插入到当前删除列表头部的前面,同时修改相应信息。

3　结束语

SADBS 系统是一个基于 Realms、主存数据库和可扩充数据库的空间分析 DBMS,通过 Realms 概念对空间元素作出完整性约束及排序规则,使得基于 Realms 的空间数据组织能有效地支持平面扫描算法。目前我们已设计并实现了其空间子系统,具有存储、管理空间数据功能和近 50 个空间拓扑分析操作;实现了主存 R 树索引的创建、插入、删除、更新等动态操作,有关 R 树索引的具体实现作者已另文叙述[6];基于 R 树索引还可以进行各种空间连接查询。今后将在系统性能改进方面作进一步研究。

参考文献:

[1] Güting R H. An Introduction to spatial database systems[J]. VLDB Journal, 1994,3(4):357 – 400.

[2] Garcia – Molina, H.(美)等著, 杨冬青等译. 数据库系统实现[M]. 北京:机械工业出版社, 2001.

[3] 秦小麟.空间分析数据库的研究方法及技术[J].中国图象图形学报, 2000,5(9):711 – 715.

[4] Güting R H, Schneider M. Realms:A foundation for Spatial Data Types in Database Systems[C]. Proc. 3rd Intl. Symposium on Large Spatial Databases, Singapore, 1993:14 – 35.

[5] Guttman A. R – tree:A dynamic index structure for spatial searching[C]. Proc. Of the ACM SIGMOD Intl. Conf. on Management of Data. 1984:47 – 54.

[6] 李萍. 基于 Realms 的主存 R 树索引的实现[J]. 计算机应用,2003,23(5):94 – 97.

The Spatial Database Design and Index Organization of SADBS

LI Ping

(Computer Engineering Department, Yancheng Institute of Technology, Yancheng Jiangsu　224003, China)

Abstract: Spatial analysis system is time – consuming and the storage media is one of the important reason that restrict the system speed. It will greatly improve the performance of the system if we store the database and the spatial index in main memory. In this paper, we introduce the storage structure of the spatial database and the organization of the spatial index of SADBS in detail.

Keywords: SADBS; spatial database; spatial index; Rtree; Realms