地理信息系统 时空信息数据库 层次存储 (14)

计算机科学2000Vol.27№2.7

47-49,91

时空信息的层次存储和管理

Hierarchical Storage and Management for Spatiotemporal Data

柴晓路 曹 晶 施伯乐

(复旦大学计算机科学系数据库研究中心 上海200433)

P9 TP392

Abstract To get high-performance data accessing in GIS, a hierarchical model for spatiotemporal data presentation and storage, which is composed of four levels: general level, global level, theme level and view level, is provided. The general level, global level and theme level are for logical presentation and physical storage, and the view level is for logical presentation only. The general level provides uniform presentation for spatial objects by time measurement. The global level, as the mediate level in data presentation, simplifies the decomposition and recomposition for spatial data, which can lead to fast data access and easy level-to-level data transformation. The theme level classifies the spatial object according the objects' original type, which is a semantic level. Lastly, the view level provides the data access interface for end-user.

Keywords GIS, Spatiotemporal database, Hierarchical storage

1 引言

目前地理信息系统(GIS)已经逐步应用于各类机构和团体中。网络通讯和分布式计算是GIS进入企业化水平的核心技术支柱,它们允许公用的数据和应用模型在网络上供多用户使用。但对于非专业用户来说,GIS还处于一个相当难使用的状况,如何为用户所需的基本功(定位、识别、比较、关联等)能提供一个相当易使用的界面是GIS研究的一个重要方面。另一面就是GIS内部的实现,GIS的核心是空间信息数据库,在很多情况下,它还有可能被扩充为时空信息数据库。在空间信息数据库中,区域分类存贮已越来越多地受到重视,较有代表性的有R-树、K-D-B树、BANG树等,起初的空间信息数据库并未采用流行的关系数据库管理,而是使用自设计结构的文件来实现,其结果是实用性受到了很大的限制。目前,运用关系数据库作为空间信息数据库的基本载体,并辅以特殊结构的空间信息数据文件已经成为空间信息数据库实现的主流。在关系数据库中,我们可以获得原本缺乏的对空间对象表现力,结合特殊结构的空间信息数据文件则可以获得优秀的数据操纵能力,通过两者的结合,可以为用户提供更好的GIS服务。

但是针对时空信息数据库的管理,尚未有很好的模型,主要原因是时空数据库中的数据复杂度再一次上升,如何兼顾时间域和空间域而获得较好的数据访问能力是时空数据库的一个研究方向。在这个背景下,本文提出了一个时空信息存贮管理的层次模型,并作了一些探讨。

2 存贮模式

地理信息系统中信息的核心是地理分布信息,属性信息是依托地理分布信息而存在的。地理分布信息的变更相对一般信息系统而言其变更频率往往是不频繁的。在绝大多数情况下,地理分布信息一旦持久化就不会被改变,直到现实世界中其对应的地理对象发生了变更,对于一般的地理信息系统来说,查询是系统的主要操作,不仅待查询的地理空间信息的数据量很大,而且查询结果所包含的数据量也很大。那么,如何调整系统的结构,使之尽量对查询优化从而获得较佳的系统性能,是一个地理信息系统实现的时候尤其需要考虑的问题。当然,对信息的添加、修改和删除操作也应当是完备的。

目前地理信息的组织一般通过空间对象这一逻辑单元来实现。事实上,空间对象数据库的设计在地理信息系统的设计中占了很大比重,因为空间对象完全要依靠数据库来存贮和管理。

我们考察了一些地理信息系统,认为在地理信息系统的信息存贮模式的设计中应当考虑到它的这几个特性:1)面向主题的地理信息查询与分析;2)查询对象具有海量信息;3)用户需要较高的查询响应;4)空间对象信息更改的可能性相对较小。根据这些特性,结合具体项目的具体需要,我们总结出一个层次模型来存贮表示地理信息系统内的所有地理信息(包括空间对象的空间信息和属性信息)。主要思想是在信息存贮模式上,兼顾输入与查询,但主要是对查询进行优化,按空间域和时间域对空间对象进行划分,全局数据和局部数据并存,存在同一信息的数个副本,通过用户视图和

索引信息组织面向主题的信息表示。

该模型由如下四层组成：广义模式、全局模式、主题模式、视图模式。模式的示意图如图1所示

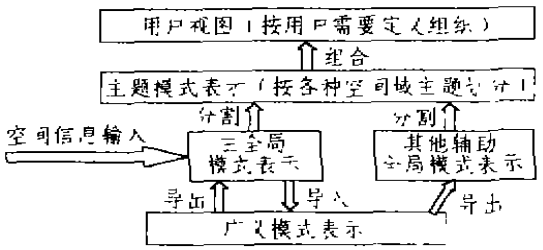

图1　层次模型的体系结构示意

广义模式 G

在本地理信息系统的空间信息存贮模式中，一个广义模式表示是系统中所有有效存在或曾经有效存在的全体空间对象信息的表示。每一个对象除其原固有的属性信息外，还有其存在的有效时间的属性，表明该对象存在的时间域。一个地理信息系统中广义模式表示一般认为只有一个。

之所以设置广义模式表示，是因为地理信息系统与一般数据库应用系统不尽相同，当一个空间对象不复存在时，其信息还是需要保留的。地理信息是历史性的，我们不但关心现在的对象，同时也关心过去曾经存在的对象。一个简单的例子就是我们常常要比较旧城区和新城区的建筑分布。从这一点来看，广义模式表示同数据仓库有一定的相通之处，广义模式下的空间对象表示是四维的，除了其空间位置外，还有时间位置，即它具有时空位置。另外值得注意的是，在本模型内空间的三维并不是通常意义的三维，因为目前的地理信息系统的下层支撑系统一般对二维地图具有很强的表现能力，而对三维空间的对象表现能力不足，本模型也是针对二维地图而言的。本文中的空间的第三维事实上是逻辑上对空间对象的分层，我们将每一层赋予第三维坐标，形成空间对象的空间三维表示。

全局模式 P

从应用的角度出发，一个具体的应用总是针对一个或多个时间域或时间点在一定区域内的所有或部分空间对象的，那么就有必要按时间和空间的量度分别划分空间对象，并给予一个相对局部的表示模式，由于访问多个时间域的需要远比访问多个空间域的需要小得多。因此，我们首先按时间域来划分广义模式表示下的空间对象，即每一个时间域内有效的空间对象全体构成一个全局模式表示，这里的全局指的是空间上的全局。

一般来看，应该，至少应该有一个主全局模式表示，即当前有效的空间对象构成的全局模式表示，该模式表示应当是整个系统空间信息表示的核心。所有新输入的空间对象都将首先成为主全局模式下的空间对象表示，然后再导入广义模式表示中去。主题模式表示的生成也是从该层模式表示出发，进行分割和重新组织形成各自的主题模式表示。简而言之，主全局模式是本空间信息表示模型的核心，它是各模式联系的纽带。此外，我们当然需要其他时间域的全局模式表示的补充，在临时使用或特殊应用环境下，我们需要构造特定时间域的全局模式表示，此时很有可能我们并不需要纯粹意义上的全局模式表示，而只需要一个局部的全局模式表示，所谓局部，是指其中包含的对象只是该时间域中所有空间对象全体的一个子集，这一设计主要是考虑到这部分是临时生成和临时保留的，应将其规模定义得足够小。

主题模式 T

对于一个全局模式表示而言，其中包含的空间对象的类型是非常丰富的，且它们针对的应用及包含的空间信息和属性信息的结构都不尽相同，而一个用户的具体应用总是针对若干种类的空间对象的，因此根据属性信息存贮和空间检索的方便性和一致性的要求，我们需要将一个全局模式表示，按空间对象类型，或者也可称按空间对象所体现的主题，划分为多个相对全局模式表示来说局部的表示模式，我们称其为主题模式，并且这种局部性包含了时间域的局部性和空间域的局部性，例如：城市的道路分布是一个主题模式表示，城市的绿化分布也可以是一个主题模式表示。

在主题模式这一级，我们还可以将其定义为两个子层：全域主题模式和区域主题模式。

全域主题模式：一个全域主题模式表示是包含了全空间域内该主题下的所有空间对象信息。

区域主题模式：一个区域主题模式表示一般是相对于一个全域主题模式表示的，通常将一个全域主题模式划分为若干个区域，或将一个全域主题模式中划出一定区域，作为区域主题模式。设置区域主题模式的好处是加快检索速度，减轻视图生成的负担。它并不是必须的。

视图模式 V

对用户而言，一个应用视图一般是这样定义的：在一定区域，包含若干主题信息的地理信息的全体，那么我们就需要将若干主题模式表示作为若干层，并界定区域，以便定义出用户需要的视图模式。一般情况下，视图模式表示是临时生成或准临时生成（可能被保留一段时间），并不会被永久保存，数据库中保存的仅是视图定义的信息。视图定义信息可以被保存在检索信息数据库中，因为说到底所谓用户视图信息也就是对

空间对象信息的检索信息，此外检索信息数据库中还应定义很多用于快速检索的索引信息，主要目的是加快查询操作。这一部分对于查询的性能来说也是非常重要的，鉴于本文的重点不在于此，就不再详究。

3 模式转换

广义模式⇔全局模式

设广义模式表示为 S_g，广义模式下空间对象表示为三元组 $O_g(s,p,t)$；全局模式表示为 S_f，全局模式下空间对象表示为二元组 $O_p(s,p)$，其中 s 为空间对象的空间信息，p 为空间对象的属性信息，t 为空间对象的有效时间域。显然，$O_p(s,p)$ 为 $O_g(s,p,t)$ 的投影。时间域是一个二元组 (t_s,t_e)，t_s 为起始时间，t_e 为终止时间。事实上可能有两种状态：

1）封闭：(t_s,t_e) 该空间对象已经成为历史，其有效时间从 t_s 起至 t_e 终。

2）右开：$(t_s,null)$ 该空间对象当前正存在，其有效时间从 t_s 起。

广义模式到全局模式的转换可以描述如下：（选定时间为 t'）

```
select Op(s,p)
From Sg
Where t' in t
```

对于全局模式到广义模式的转换，则有两种方式，一种是批量的，即将全局模式下的所有对象一次成批地转换为广义模式表示。其过程可以描述如下：

```
for (all Og(s,p,t) in Sg) do
 if (t 为右开状态 and 无对应 Of(s,p)存在)then(所谓
   对应使用 Object ID 来联系)
  t.te=current_time;
end for
for (all Op(s,p) in Sf) do
  if(Sg 中存在对应的 Og(s,p,t))then
    Og.s=Of.s;Og.p=Op.p;
 else
  New Og(s,p,t);Og.s=Op.s;Og.p=Of.p;
  t.ts=current_time;t.te=NULL;
 end if
end for
```

批量方式的好处在于可以集中处理，可以在系统非工作时间进行，减轻系统负担，但该方式的缺点在于可能丢失有用信息，例如空间对象的变更和信息修正的区别，而且包含了很多无用的操作，例如需要遍历 S_g。这些问题在实时方式下，都被很好地解决了，当然在实时方式下，会增加工作时间的系统负担，这就要靠系统设计者的权衡了。所谓实时方式描述如下：

定义全局模式下的几个对空间对象的基本操作：

1. insert　插入新的 $O_p(s,p)$

1 delete　删除指定 $O_p(s,p)$

3. remove　删除错误输入的 $O_p(s,p)$

4 update　更新 $O_p(s,p)$ 信息

5. correct　更正 $O_f(s,p)$ 信息

其中 remove 和 correct 都是用于纠正错误的输入，其余三个是空间对象的基本操作，它们触发的广义模式的相应操作如下：

1 insert　插入对应的 $O_g(s,p,t)$；$t.t_s=current\_time$；$t.t_e=NULL$；

2. delete　修改对应的 $O_g(s,p,t)$ 的时间域：$t.t_e=current\_time$；

3. remove　删除对应的 $O_g(s,p,t)$

4 update　修改对应的 $O_g(s,p,t)$ 的时间域：$t.t_e=current\_time$；

赋于 $O_p(s,p)$ 新的 ObjectID，在 S_g 中新增相对应的 $O_g(s,p,t)$；

$t.t_s=current\_time$；$t.t_e=NULL$；

5 correct　修正对应的 $O_g(s,p,t)$ 的 s 和 p。

实时方式有效地解决了从全局模式到广义模式的转换，并对信息更正提供了支持，较好地保证了广义模式的完整性和一致性。

对于从全局模式到主题模式的转换，则与实际应用关系比较紧密，需要结合系统针对的领域、用户关心的主题具体分割划分，在这里也就不详细阐述了。

另外要指出的是，在实际应用中，下三层模式的划分并不是严格分离的，可以根据需要按耦合度的不同相互交融。例如在我们实现的温岭城建 GIS 中，全局模式中包含了主题模式的概念，整个全局模式是按照主题来组织的，而且一般来看，这也是空间信息存贮的常用方式。另外由于使用 GIS 支撑软件 MapInfo 作为系统的基本平台，因此主题模式就对应于 MapInfo 的空间对象文件，而未使用关系数据库表现，在这个系统中，各个模式的关系如下：

$$\pi_c(G)=P \quad T_i\subset P \quad i=1,2,\cdots,n$$

$$V_j=\bigcup_{i=1}^{n}T_i \quad j=1,2,\cdots,l$$

结论　本文提出的这个层次模型将空间数据库在时间和空间的不同层次上进行局部化，使空间信息按照耦合度的大小分类存贮，从而令空间对象操纵基本上总能在预先划定的较局部的视图中进行，保持较高的效率。通过广义模式的这一表示层次，使同一空间中不同时间段的空间对象得以统一管理，并可以通过广义模式到全局模式的模式转换实现重现任一时刻的空间对象分布信息。通过在广义模式和主题模式之间设置全局模式，降低了空间对象抽取的复杂度。

但是由于该模型忽略了很多细节，通过简化，获得了描述的清晰性，不可否认将其应用到实际系统尚有

（下转第91页）

(1)使用抽象逻辑结构作图工具进行逻辑算法设计。抽象逻辑结构图不涉及语言实现细节，可使程序员将注意力集中于解决问题的算法上，使算法的设计更加有效。抽象逻辑结构图提供的程序块 BLK 结点和未定义 UND 结点可方便地支持自顶向下、自底向上、以及自顶向下和自底向上相结合的程序设计。设计中将逻辑 ALSD 中的未定义结点确定化，对未分解复合逻辑结点进行细化，每进行一步就得到一个新的进化逻辑 ALSD。当逻辑 ALSD 或进化 ALSD 满足形式化定义中的逻辑结点分解条件并且不含任何未定义结点时，我们就得到确定的逻辑程序，即确定逻辑 ALSD。

(2)将确定逻辑 ALSD 中带数据流的逻辑结点的数据流表示为 Java 程序语言形式，遵循 Java 语言的语法规则写出每个带数据流的逻辑结点的 Java 表达式或基本操作语句，构造并得到操作表达式表。操作表达式表具体给出程序与 Java 语言密切相关的实现细节。

(3)建立确定逻辑 ALSD 和操作表达式表的联系，在确定逻辑 ALSD 中每个带数据流的逻辑结点上标记操作表达式表中与其对应物理表示的表项号，得到具有实现形式映射关系的确定 ALSD。至此，我们已建立了一个完整的 Java 过程蓝图程序。

(4)生成程序源代码和实现 ALSD。从 Java 过程蓝图生成 Java 源程序是为了进一步编译并最终运行程序，而生成实现 ALSD 则是 Java 过程蓝图产生的一种附加文档，实现 ALSD 是将确定逻辑 ALSD 上的每个带数据流的逻辑结点语义替换为操作表达式表中对应 Java 语言实现表示后得到的树图。在 Java 过程蓝图基础上产生 Java 程序源代码和实现 ALSD 是非常机械的，使用代码生成工具和文档生成工具是完成此项工作的最好选择。

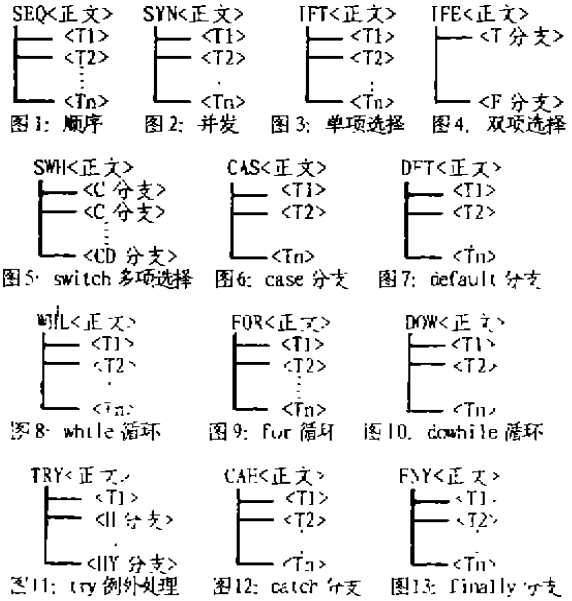

图1: 顺序　图2: 并发　图3: 单项选择　图4. 双项选择

图5: switch 多项选择　图6: case 分支　图7: default 分支

图8: while 循环　图9: for 循环　图10. double 循环

图11: try 例外处理　图12: catch 分支　图13: Finally 分支

参 考 文 献

1 刘建宾 一种工程化的程序表现技术 过程蓝图. 计算机工程与应用，1996，32(1)：31～34

2 Gosling J. et al. 著，蒋国新，等译. Java 语言规范. 北京 北京大学出版社，1997 12

3 Arnold K，Gosling J. 著，杨承高，等译 Java 程序设计语言. 北京 北京大学出版社，1998. 1

4 李明，李厚安 Java 程序设计教程. 北京 科学出版社，1997 8

5 Tripp L. L. A Survey of Graphical Notations for Program Design-An Update ACM SIGSOFT Software Engineering Notes，1988，13(1)

6 Chu Yaohan. Software Blueprint and Examples. Lexington D C Heath and Company，1982

(上接第49页)

一段距离，需要将很多细节落实到系统的设计中去。此外，在该模型中存在相当的数据冗余，在对模型的进一步探讨中可以考虑将重复的部分局限在控制信息和索引信息，而非空间对象信息的全体。另外考虑针对各种不同的应用环境，将该模型加以细化，使之更能适应各种具体系统的具体要求。

参 考 文 献

1 Newell D. Perspectives in GIS Database Architecture[C] SSD'1997

2 Aggarwal C C，et al The S-Tree，An Efficient Index for Multidimensional Objects[C] SSD'1997

3 Belussi A，et al. Manipulating Spatial Data in Constraint Databases[C] SSD'1997

4 詹奇波，张其善 电子地图数据库存贮文件的设计[J]. 计算机科学，1996，23(3)

5 Traynor C，Williams M G. Why Are Geographic Information Systems Hard to Use?[C]. CHI'95

6 Moore L，et al Network Access to Geographic Names Defense Mapping Agency Prototype to the Information Super Highway[C]. ACSM/ASPRS，1995