中国农业大学学报 2002, 7(3): 47～51
Journal of China Agricultural University

① 基于关系型数据库带时间维GIS的一种数据模型

严泰来 吴 平*
(中国农业大学信息学院, 北京 100094)

摘 要 本研究提出一种依据关系型数据库管理下带有时间维数据结构的GIS模型。模型基于面向对象的思想, 探讨在国标地理信息系统数据交换格式基础上, 补充时间维数据结构, 实现GIS基于时间信息直接进行的历史追溯与断面复原的技术; 给出了时空数据模型的设计、历史数据的追溯与断面复原算法的分析, 以及算法实现中关键部分的程序实例。

关键词 地理信息系统(GIS); 时间维; 关系型数据库; 面向对象程序设计
中图分类号 TP391

Study on Model of Data Structure for GIS with Timing Dimension Based on Relationship Database

Yan Tailai Wu Ping
(College of Informatics, China Agricultural University, Beijing 100094, China)

Abstract This paper puts forward to a model of data structure for GIS with timing dimension based on the relationship database. This model based on the OOA and OOP technology, study on the methods of history information backward and recovering. This paper shows the design of time-space data model, the analysis of methods of history information backward and recovering, and some example of important programming.

Key words GIS; timing dimension; relationship database; object oriented programming

地理信息系统是一种管理地学信息数据、分析各种地学现象、实行地理资源环境数字化管理的强有力的工具。随着系统应用的深入, 在现有系统平台基础上开发与研制带有时间维的地理信息系统势在必行。时间与空间是相互关联的变量, 是描述与地学有关的各种事物发生、发展演化必不可少的相关的数据。自然科学中, 所谓动态研究就需有时间维数据; 在管理科学中, 研究国土资源、资产变更的历史沿革也必须将空间信息置于时间维的框架上。时间信息介入空间信息的描述之中已经是必不可少。

在实际工作中, 对于带有时间维的GIS的功能有以下要求: 1) 时间断面复原, 即要求系统能够对数字图件覆盖地区的全部或局部按任意指定时间段以图件形式准确复原出来; 2) 地块追溯、查询历史沿革, 即要求系统对数字图件中任意指定的某一地块, 追溯查询历年演变的情况, 查询结果将图形与属性数据显示出来。以上功能在一些场合下要求能够分别直接运作, 也有一些场合系统将其中之一与其他空间分析功能整合在一起完成某项工作, 比如房地产评估, 估价员不但要知道当前某宗地情况, 还应知道该宗地及其周边土地以前的利用情况, 从而综合分析当前宗地价位。

收稿日期: 2001-11-12
国家重点基础研究发展规划资助项目
* 吴平, 教授, 研究方向为计算机应用。联系作者。北京圆明园西路2号

带时间维的 GIS 实施困难之处在于数据的组织。实际工作中,表达一个时相的某个地区或某个行政单位的数字图件数据量相当大,如果将时间维数据加进来,而不加以周密的组织,容易造成数据量的极度膨胀,不但在存贮空间上不能被一般系统所接受,而且也给不同时间的同一地块的纵向空间分析带来困难。目前,对于这一问题基本上有 2 种处理方法:一种是将时间数据与空间 X Y 坐标数据一样对待,一同存入数据文件中,系统引进时间拓扑的概念,按照用户对时空数据或追溯历史沿革或复原历史数据分别进行时间拓扑分析,将所需空间数据逐一提取出来;另一种是所谓时间快照法,即如同摄像机一样,空间数据每变动一次,即使对全部数据只有 1% 的变动,也将全部数据录制一遍,"贴"以时间"标签"存贮起来。这 2 种处理方法实际代表着 2 条技术路线,前者是以复杂的时空拓扑分析换取存贮空间的减少与存贮方式的简单;后者是以大量存贮空间的冗余赢得数据结构的简单与系统算法的易行。现在问题是如何把它们有机地结合起来,寻求一种既要算法简单,又要尽量不对现有系统数据结构做大调整又要减少数据冗余、实现快速查询的数据结构及算法。本研究对这一问题进行了初步探索。

1 时空数据模型设计

面向对象的程序设计是当今计算机程序设计的一个重要思想,地理信息系统的程序基本上也是遵循这一思想设计的。任何一幅地理图件都有 3 个基本要素:点、线、面。通常,地理信息系统都是分别将点状地物、线状地物、面状地物作为对象,对每一对象分别赋予 ID 码,在属性数据库中,以此 ID 码作为主键,将其研究或管理所需用的属性数据与 ID 建立连接,生成属性关系数据库。当前各种地理信息系统对于空间数据的数据结构设计千差万别,但是以点、线、面作为数字图件的基本对象这一思想却是一致的。因而,在国标地理信息系统数据交换格式中,也是将点、线、面作为 3 类数据,分别规定其数据格式,并分别进行处理的。针对这一情况,我们对时空数据模型作以下设计:

1) 设立对象现状数据表,该数据表结构见表 1。对象现状数据表记录的是当前系统中的所有线状地物、面状地物的对象,数据表中每一记录对应数字图件中的一个图件对象。在数据表中,对象 ID 为系统对象的惟一标识;FromDate 为对象有效起始时间;X_{min} 和 Y_{min} 为对象所在包络矩形的左上角坐标;X_{max} 和 Y_{max} 为对象所在包络矩形的右下角坐标。包络矩形(Range)是指线状地物中轴线坐标链,面状地物边界线坐标链的坐标集合中 X、Y 的最大、最小值[1],系统中用这 4 个值生成的矩形将对应线状地物、面状地物的大致位置界定下来,以便于系统用 R^+ 树或其他方法快速检索。

在表 1 结构中,设置了有效起始时间(FromDate),这是指该对象从什么时间开始生效。对于从建表时间直到当前不曾发生变更的对象,则 FromDate 数据项中记录的是建表时间。由于上表仅记录对象现状数据,当然每一对象都是从起始时间到当前都是有效的。如果,对象发生任何变更,包括边界改动、删除或分裂,则都需将原有对象从此表中删除,并将此对象的信息保存到对象历史数据表(表 2)中,以便系统追溯或时段复原使用。变更后的对象添加在表 1 末尾,变更的时间填写在FromDate数据项内,表明该对象从这一时间起到现在都是有效的,对

表 1 对象现状数据表结构

对象 ID	FromDate	X_{min}	Y_{min}	X_{max}	Y_{max}
int	datetime	int	int	int	int

表 2 对象历史数据表结构

对象 ID	FromDate	ToDate	X_{min}	Y_{min}	X_{max}	Y_{max}
int	datetime	datetime	int	int	int	int

于变更过的对象的 ID 码要用新 ID 码, 以示变更前后的区别。

当然, 对于点状地物, 不存在其包络矩形数据, 因而点状地物对象现状数据表中不必设置包络矩形数据项, 而代之以点位 *X Y* 数据项。

2) 设立对象历史数据表(表 2), 该数据表与对象现状数据表结构基本相同, 但需要设置 2 个时间字段 FromDate 与 ToDate, 用来描述对象的生命期。

对象历史数据表记录系统中所有变更的对象。在数据表中 1 条记录对应系统中 1 个曾经变更的对象, 表中 ToDate 字段的逻辑意义为对象失效的时间, 其他字段的逻辑意义与表 1 中对应的字段相同。

需要说明的是表 2 中的对象 ID 与表 1 中的对象 ID 是统一的, 对于当前的或历史的对象要保持惟一性, 也就是说不能允许任何一个 ID 码在表 1 和表 2 中同时出现。如果出现这种现象, 就会破坏系统数据的一致性。在表 1 的说明中已经叙述过, 如果某个对象发生变更, 则应删除该对象在表 1 中的记录, 同时将该记录补充到历史数据表(表 2)中。这一变更时间即此对象在现状数据表中消失而成为历史数据的时间, 该时间将记入对象历史数据表中的 ToDate 数据项, 与从表 1 中继承来的 FromDate 数据项共同构成该对象的"生命期"。

对象现状数据表(表 1)与对象历史数据表(表 2)描述记录了全部时空数据。时间信息数据体现在 FromDate 和 ToDate 数据项中; 而空间信息数据则以对象 ID 与空间数据表中的坐标链 *X Y* 数据、弧段数据、结点数据相关联[2, 3]。

这样看来, 在表 1 与表 2 的背后, 以 ID 标识码为关键字的数据表存储着大量的 *X Y* 坐标数据以及它们的索引。如何组织这些巨额的 *X Y* 数据, 以达到数据冗余小、检索方便和保持关系型数据结构, 尚包含大量其他内容, 不在这里讨论。

2　历史追溯与时间断面的复原

历史追溯是对研究地区一个指定地块的纵向查询, 而时间断面复原可以认为是对时间序列的横向断面的查询。表 1 与表 2 展示的基于线、面对象的带时间维数据结构可以支持快速准确地完成这 2 项任务。

2.1　历史追溯

考虑到指定地块在多次历史变更中有可能面目全非, 不仅 ID 码发生变化, 地块形状也会发生变化, 该指定地块或被相邻地块合并, 或被割裂成多块, 亦或部分被吞并、部分被割裂。这里惟一可作为追溯痕迹的是当前该指定地块包络矩形。用包络矩形作线索, 在历史数据表中追溯与指定地块相关的所有历史地块数据。由于历史数据表中每个地块都有包络矩形数据, 凡是当前指定地块的包络矩形数据 X_{minp}, X_{maxp}, Y_{minp}, Y_{maxp} 与历史地块包络矩形数据 X_{minh}, X_{maxh}, Y_{minh}, Y_{maxh} 存在以下逻辑关系, 则可认为 2 包络矩形相关(相交叠或相包容):

$$(X_{minh} < X_{maxp}) \text{ and } (Y_{minh} < Y_{maxp}) \text{ and } (X_{maxh} > X_{minp}) \text{ and } (Y_{maxh} > Y_{minp}) \quad (1)$$

当然, 2 包络矩形相关只是对应的 2 个多边形相交叠或相包容的必要条件, 并不是充分条件。为了证明历史上的地块确与当前指定地块在地域上相关, 还必须再作叠加分析, 检验 2 者的弧段边界是否有交点, 如果交点数≥2, 则证实当前指定地块或地块的一部分确实属于历史上这个地块演变而来。

地块历史追溯除了式(1)的地块位置相关条件以外, 还需要有以下时间维数据条件:

$$T_{hf} < T_p \quad (2)$$

式(2)中, T_p 为指定追溯时间, 如果 T_p 时间就是现状数据表的"现在", 则不必用此时间维数据条件做判断, 因为所有历史数据表中的数据在时间数据上都是符合式(2)的。而如果 T_p 是历史上的某个时间, 要求从这个时间起向前追溯, 则式(2)就有意义, 此时, T_{hf} 为历史数据表中 FromDate 数据项的数据。凡符合式(2)逻辑条件的地块都是其"寿命"有效期覆盖指定时间或在指定时间之前的地块。

对象状态历史追溯的基本操作如下:

a. 在现状数据表中指定观察对象, 从而确定了对象的包络矩形坐标:

左上角: (X_1, Y_1), 右下角: (X_2, Y_2)

b. 建立时间临时数据表 temp. db, 数据结构与现状数据表相同。

c. 将现状数据表中的指定记录送入临时数据表 temp. db。

d. 搜索历史数据表中的记录:

```
select * from history. db
where (Xmin< X2 and Ymin< Y2) and (Xmax> X1 and Ymax> Y1)
order by FromDate DESC   (也可以按 ToDate 考虑)
```

e. 将搜索到的已经有序的记录送入 temp. db。

经过操作步骤 a～e, 在 temp 数据表中就包含了与指定对象状态可能相关的所有记录(通过 C++ Builder 程序测试)。

2.2 时间断面复原

对于任意指定的断面时间 T_c, 需要分别在现状数据表和历史数据表检索出以下符合时间逻辑关系的对象数据。在现状数据表中, 按式(3)进行检索:

$$T_f < T_c \tag{3}$$

而在历史数据表中, 按式(4)进行检索:

$$T_{hf} < T_c < T_{ht} \tag{4}$$

在式(3)和式(4)中, T_f 为现状数据表中的 FromDate 数据项的数据; T_{hf}、T_{ht} 分别为历史数据表中 FromDate 和 ToDate 数据项的数据。

利用对象现状数据表和历史数据表恢复指定时间的系统状态处理方法如下:

a. 设指定的断面时间为 Date。

b. 建立时间断面临时数据表 temp. db, 数据结构与现状数据表相同。

c. 从现状数据表中检索出所有有效期起始时间早于指定时间 Date 的记录:

```
select * from now
where FromDate< Date
```

d. 将检索到的记录全部送入临时数据表。

e. 从历史数据表中检索出所有有效期时间段包含指定时间 Date 的记录:

```
select * from history
where (FromDate< Date) and (ToDate> Date)
```

f. 将检索到的记录全部送入临时数据表。

经过 a～f 操作, 在 temp 数据表中形成了系统在指定时间有效的全部记录(通过 C++ Builder 程序测试)。

以上对于现状数据表和历史数据表检索数据的算法不存在对于同一地块发生重叠检索或

遗漏检索的问题。这是因为作为初始数字图件, 土地利用类型地块或土地权属地块都要覆盖图件的每个角落, 不存在"真空"区域。即使是水面, 也要作为土地利用类型(图 1), 并有其权属单位。任何图斑或线状地物, 一旦发生变更要将涉及变更的所有图斑都移植存入历史数据表中, 如一条弧段的改动要将弧段左右两侧的图斑都作为变更对象, 两者的 ID 识别码存入历史数据表, 一个结点的改动, 要将此结点涉及到的 3 个甚至更多的图斑或线状地物坐标链都检索出来, 将变更前的图斑或线状地物坐标链的 ID 识别码存入历史数据表。对于现状与历史数据表数据做这种增删、移植的操作, 可以保证对任何一个地块或地块一部分所在的时间段都是独立的, 不可能这个时间段又在同是该地块的另一个时间段内, 即对于 ID 标识码为 i 的某地块, 它的"寿命"期为 T_{fi}到 T_{ti}; 如若发生变更, 分割为 2 个地块, 则 ID 标识码分别为 k, j, 它的"寿命期"分别为 T_{fk}到 T_{tk}与 T_{fj}到 T_{tj}。显然, 式(5)与式(6)都不可能成立。

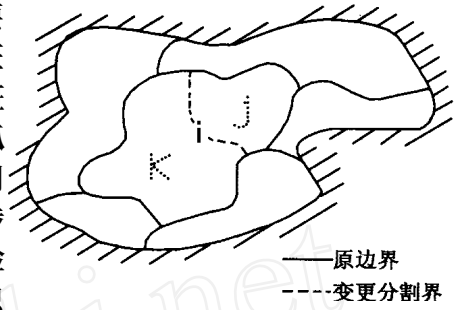

图 1　图斑分割变更

$$(T_{fi} < T_{fk} < T_{ti}) \text{ and } (T_{fi} < T_{fj} < T_{ti}) \tag{5}$$

$$(T_{fi} < T_{tk} < T_{ti}) \text{ and } (T_{fi} < T_{tj} < T_{ti}) \tag{6}$$

这就是说, 在任何一个时间断面, 同一地块或一地块同一部分不存在 2 个"寿命期", 只要在现状数据表与历史数据表分别将符合式(3)与式(4)要求的所有图斑检索出来, 就可以生成一幅完整的图件, 既没有遗漏留出的"空白", 也没有相互的重叠。

3　结论与讨论

经以上设计与分析, 可以得到以下结论:

1) 现状数据表与历史数据表的两元结构及其相应算法支持仅存贮变更的历史数据, 且这些数据以各自 ID 标识码为索引的存储结构, 从而数据结构简单, 冗余量小, 检索查询速度快。

2) 历史追溯与时间断面复原是时间维 GIS 的必备功能。现状数据表与历史数据表的并立设计方案可以支持这 2 个功能的实施且算法简单易行, 使用 SQL 数据库通用语言就可以编写程序, 这一程序结合 GIS 原有的一些空间分析功能就可以完成历史追溯与时间断面复原工作。

从 2 表的操作不断变更图形数据的过程可以看到: 在现状数据表中存贮的数据量与变更频繁、数据时间跨度大无关。而历史数据表存贮的数据却与变更频繁、数据时间跨度呈正相关。为避免历史数据表中容纳的数据过多, 检索不方便, 可以考虑定期重新开始生成 2 表。重新开始前, 将以前的 2 表合并成一个历史数据表, 存贮起来, 不再动用。重新开始以后时间变更的地块, 才作为新阶段的历史数据存贮在新历史数据表中。

对于大面积土地整理或旧城改造土地划拨的土地变更, 原则上也可以使用 2 表的设计方案记载变更过程, 只是注意在程序中做好逐个地块的删除与历史数据表中移植的操作, 以保证数据的准确无误, 维持数据的完整性与一致性。

参　考　文　献

1　严泰来, 朱德海　土地信息系统　北京: 科学文献出版社, 1993. 141～143
2　陈述彭, 鲁学军, 周成虎　地理信息系统导论　北京: 科学出版社, 2000. 39～43
3　朱德海, 严泰来, 杨永侠　土地管理信息系统　北京: 中国农业大学出版社, 2000. 52～55