支持栅格数据的GSQL扩展研究

刘 瑜 林 星 秦 适 张 毅 邬 伦
(北京大学遥感与地理信息系统研究所,北京 100871)

摘 要 空间数据库已经成为实现海量空间数据管理的高效手段,在空间数据库设计和实现中,定义GSQL是其关键。目前,针对于矢量数据访问已经有许多学者进行了研究,但对于地理栅格数据,其访问和操作接口却尚无相关规范,为此参考OpenGIS®,对该方面进行了研究,即首先定义了与栅格相关的抽象数据类型,其分别为Pixel(栅格点)、RasterRegion(栅格区域)、RasterCoverage(栅格覆盖),它们之间存在聚合关系;然后对每种数据类型进行了定义,包括其数据对象以及相关操作;进而,基于上述抽象数据类型,给出了支持栅格数据访问的GSQL-R描述以及相关实例。通过实例表明,GSQL-R能较好支持矢量/栅格一体化的数据访问和操作。

关键词 空间数据库 栅格数据 栅格空间结构化查询语言
中图法分类号: P208 TP311.12 **文献标识码**: A **文章编号**: 1006-8961(2005)01-0113-09

Research on GSQL Extension Supporting Raster Data

LIU Yu, LIN Xing, QIN Shi, ZHANG Yi, WU Lun
(*Institute of Remote Sensing and Geographic Information Systems, Peking University, Beijng* 100871)

Abstract Spatial database has become one of the dominating techniques to manage spatial data in geographical information systems, and geographic structured query language (GSQL) is the key issue of spatial database design and implementation. At present, GSQL for vector data has been widely studied. However, when considering the raster data, little research was made for their specifications on visiting and operational. Based on OpenGIS®, a GSQL extension named GSQL-R supporting raster data is discussed in this paper. At first, some abstract data types (ADTs), including Pixel, RasterRegion and RasterCoverage, are defined. The relationships between Pixel/RasterRegion and RasterRegion/RasterCoverage are part/whole. In these definitions, data objects and operations for each ADT are described in a formalized way. Then, GSQL-R syntax and some instances are presented. The GSQL-R description includes three parts, i. e. data definition language (DDL), data table schema and data manipulation language (DML). The example indicates that GSQL-R provides an efficient and integrated way for raster data access and operation.

Keywords spatial database, raster data, geographical structured query language for raster(GSQL-R)

1 引 言

目前,空间数据库(spatial database, SDB)已经成为实现海量空间数据管理的高效手段[1],国际上相应的软件产品有Oracle® Spatial、IBM® DB2 Spatial Extender等。在空间数据库实现中,其关键技术之一是定义一种支持空间数据访问和操作的查询语言。如今许多学者已对支持空间数据的查询语言进行了研究,文献[2]认为,一种查询语言需要达到以下两个方面目标:①查询语言是高层次的、强类型的、便于陈述的,以使得编程更为便捷和不容易出错;②查询语言基于一个特定构造的小的集合,很容易优化,并可使编程者不需要考虑执行的效率问题,而在文献[3]中,则指出实现空间数据查询语言的较好途径是通过扩展SQL (structured query

基金项目:国家自然科学基金项目(40071064)
收稿日期:2004-02-25;**改回日期**:2004-06-23

第一作者简介:刘瑜(1971~),男,北京大学遥感与地理信息系统研究所副教授,2003年获得北京大学计算机软件与理论专业博士学位。主要研究领域为地理信息系统原理、地理信息系统软件开发和应用、基于构件软件开发。E-mail:liuyu@urban.pku.edu.cn

language,结构化查询语言)来建立空间查询语言(geographical SQL, GSQL)。

迄今为止,对GSQL的研究主要针对矢量地物的访问和操作。如文献[4,5]中就研究了基于矢量地物的查询语言SQL/G及其优化;OGC(open GIS consortium)定义了针对简单地物要素(simple feature)的访问规范[6],其中包含一系列空间算子的定义,如空间关系检测、度量等,都是以扩展SQL支持空间对象的访问;而在Oracle spatial等空间数据库中,也主要提供对于点、线、多边形等矢量几何体的访问(在Oracle新发布的版本中,已经提供了对栅格数据的初步支持)。

然而在GIS应用中,栅格数据毋庸置疑占有重要位置,特别是随着遥感技术的发展及其同GIS的集成,都需要处理大量栅格格式的空间数据,但其主要管理途径目前还主要是文件方式,包括通用图像格式,如TIFF(以及GeoTiff)和特定格式的栅格文件,如Erdas的Img文件等。据研究,设计支持栅格数据的空间数据库,并定义相应的GSQL扩展,其意义包括以下3个方面:(1)基于成熟的商用数据库,实现栅格空间数据的高效管理;(2)定义栅格空间数据的统一访问接口,以实现数据的共享和互操作;(3)便于在空间数据库的基础上,进行矢量栅格数据的一体化查询和操作,如"基于格网DEM(digital elevation model)来计算所有行政单元内的平均坡度","基于土地利用栅格图来得到森林覆盖率最大的区域"等。目前,已经有栅格数据库(或影像数据库)的初步研究和实现,如在文献[2]中,就基于数组演算(array calculus)定义了面向多维数组的查询语言AQL(array query language),该语言虽然支持栅格空间数据查询,但是其抽象层次较高,并且语法与SQL存在较大差异;而在文献[7]描述的工作中,则针对影像数据库进行了研究,不仅实现了支持多维数组的数据库管理系统RasDaMan,并定义了相应查询语言,但由于该系统将栅格数据作为普通多维数组来处理,因此难以表达复杂的地理空间语义,如参照系统、基于栅格的空间分析等。

由此可见,要实现完整的支持矢量、栅格一体化操作的空间数据库,其一个关键问题就是在上述工作基础上,研究栅格数据的查询和操作规范的定义,并与相关已有规范(如OpenGIS®)兼容。本文探讨了支持栅格的GSQL扩展——GSQL-R,这是解决上述问题的核心。

2 栅格数据模型和抽象数据类型定义

考虑到抽象数据类型(abstract data type)是扩展关系数据库的关键,如同OpenGIS®定义的点(Point)、曲线(Curve)、表面(Surface)、几何体集合(GeometryCollection)等,为了给出支持栅格的结构化查询语言,首先需要根据栅格数据的基本操作来定义支持栅格空间分析和操作的数据类型。

在GIS中,栅格数据的操作可以分为点变换、邻域变换、区域变换以及几何变换等。其中,点变换包括栅格数据的算术运算以及多个栅格数据的叠加复合分析;邻域变换可用于计算坡度、坡向等,也可用于模拟元胞自动机(cellular automata)演化;区域变换主要指对一个区域内栅格数据的计算,如统计平均值、极值等;几何变换则包括了栅格数据的缩放、旋转以及基于多项式各种几何变换等。

基于栅格空间数据的特点以及对上述基本操作的支持,给出的描述栅格数据模型的类图如图1所示,其中Surface和SpatialReferenceSystem类型均采用OpenGIS®中的既有定义[5],而Pixel(像素)、RasterRegion(栅格区域)和RasterCoverage(栅格覆盖)形成的整体/部分关系,则表达了不同粒度的栅格数据抽象。Pixel是栅格数据的基本单元,而RasterRegion和RasterCoverage之间的关系则如同于OpenGIS®的Feature和FeatureCollection之间的关系。下面主要对这3种类型进行定义和进一步描述。

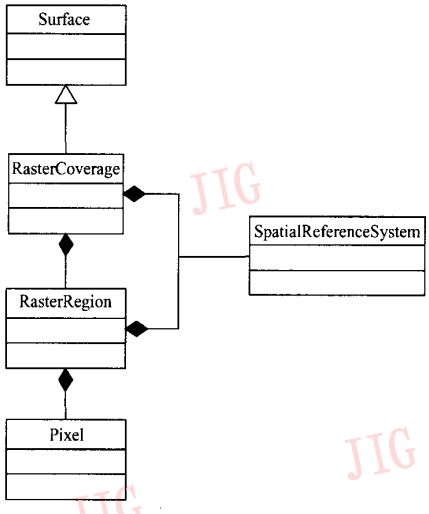


图1 支持栅格操作和分析的栅格对象模型

Fig. 1 Object-oriented model supporting raster data analysis

2.1　Pixel 类型

在 GSQL-R 中,一个栅格点(Pixel,像素)描述了二维空间 $\mathbf{R}^2$ 中一个规则正方形区域的信息,它形成了对研究区域的划分,是构成栅格数据的基本单元。若采用抽象数据类型(abstract data type, ADT)表示方式,则 Pixel 类型定义为(以下假设 p_i 为 Pixel 对象,$i \in \mathbf{N} \cup \{0\}$):

ADT Pixel

{

数据对象:

一个 Pixel 数据类型是复合类型,它定义了栅格点的数值、位置以及尺寸。

基本操作:

Value ( ): Variant

如果是单层栅格数据(如 DEM 数据或单波段遥感数据),则返回为简单类型;如果是 RGB 图像或者多波段遥感图像,那么返回值应该是一个由简单数值类型组成的数组。

Center( ): Point

用于得到栅格点的中心位置。

Size( ): Double

用于得到栅格点的尺寸,在本研究中,由于假定栅格点分布为二维空间中的正方形,因此可以用一个数值来表示栅格点的尺寸。

Geometry( ): Polygon

用于得到包围一个栅格点的最小多边形,事实上这个多边形是以栅格点的中心为中心,大小为栅格尺寸的一个正方形。

ValueType ( ): RasterValueType

栅格点存储数据的数值类型,RasterValueType 为枚举类型,定义如下:

```
enum RasterValueType{
    r_nominal,r_numerical,r_logical,r_colorvalue
};
```

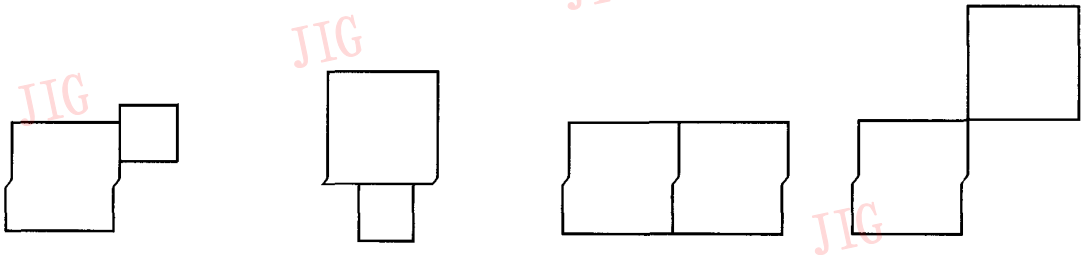
其中 r_nominal 表示栅格数据为命名量,只能进行相等判断,不能进行其他算术运算;r_numerical 为数值量,数据可以进行大小判断以及其他算术运算;r_logical 为逻辑类型,栅格数据可以进行相等判断及算术乘法运算;r_colorvalue 表示栅格数据为 rgb 类型,不能进行任何运算,但可用于数据显示。

Touches (anotherPixel: Pixel): Integer (QueenTouches)

如果两个栅格点 p_1, p_2 在空间上相邻,则返回为真值(true),栅格点相邻的定义如下:

$p_1.\text{Touches}(p_2) \Leftrightarrow (\text{fabs}(p_1.\text{Center}().X() - p_2.\text{Center}().X()) = (p_1.\text{Size}() + p_2.\text{Size}())/2 \wedge \text{fabs}(p_1.\text{Center}().Y() - p_2.\text{Center}().Y()) <= (p_1.\text{Size}() + p_2.\text{Size}())/2 \vee (\text{fabs}(p_1.\text{Center}().Y() - p_2.\text{Center}().Y()) = (p_1.\text{Size}() + p_2.\text{Size}())/2 \wedge \text{fabs}(p_1.\text{Center}().Y() - p_2.\text{Center}().X()) <= (p_1.\text{Size}() + p_2.\text{Size}())/2)$

其中 fabs 为取绝对值操作,下同。图 2(a)和 2(b)表示了栅格点相邻的情形。



(a) 相邻(边交叠情形)　(b) 相邻(边包含情形)　(c) 正相邻　(d) 全相邻

图 2　栅格点的相邻

Fig. 2　Touch Relation between two pixels

OrthoTouches(anotherPixel: Pixel): Integer (RookTouches)

如果两个栅格点 p_1, p_2 在空间上正相邻,则返回为真值(true),栅格点正相邻的定义如下:

$p_1.\text{OrthoTouches}(p_2) \Leftrightarrow p_1.\text{Size}() = p_2.\text{Size}() \wedge ((\text{fabs}(p_1.\text{Center}().X() - p_2.\text{Center}().X() = (p_1.\text{Size}() + p_2.\text{Size}())/2 \wedge p_1.\text{Center}().Y() = p_2.\text{Center}().Y()) \vee (\text{fabs}(p_1.\text{Center}().Y() - p_2.\text{Center}().Y()) = (p_1.\text{Size}() + p_2.\text{Size}())/2 \wedge p_1.\text{Center}().X() = p_2.\text{Center}().X()))$

栅格点正相邻是相邻的一种特例,它是针

对于分辨率相等的栅格点，即在一个 RasterRegion 中，基于四邻域模型的所有相邻栅格点都是正相邻，如图 2(c)所示。

FullTouches (anotherPixel: Pixel): Integer (BishopTouches)

如果两个栅格点 p_1,p_2 在空间上全相邻，则返回为真值(true)，栅格点全相邻的定义如下：

p_1. FullTouches (p_2) $\Leftrightarrow p_1$. OrthoTouches (p_2) $\vee$ (fabs (p_1. Center ( ). X ( ) $-$ p_2. Center ( ). X( )) = p_1. Size ( ) $\wedge$ fabs (p_1. Center ( ). Y ( ) $-$ p_2. Center ( ). Y ( )) = p_1. Size ( ))

栅格点全相邻是相邻的一种特例，在一个 RasterRegion 中，基于八邻域模型的所有相邻的栅格点都是全相邻，如图 2(d)所示。

Overlaps(anotherPixel: Pixel): Integer

如果两个栅格点 p_1,p_2 在空间上交叠，则返回为真值(true)，栅格点交叠的定义如下：

p_1. Overlaps (p_2) $\Leftrightarrow$ fabs (p_1. Center ( ). X ( )) $-$ (p_2. Center ( ). X ( )) < (p_1. Size ( ) + p_2. Size ( ))/2 $\wedge$ fabs (p_1. Center ( ). Y ( ) $-$ p_2. Center ( ). Y ( )) < (p_1. Size ( ) + p_2. Size ( ))/2)

Disjoint(anotherPixel: Pixel): Integer

如果两个栅格点 p_1,p_2 在空间上相离，则返回为真值(true)，栅格点相离的定义如下：

p_1. Disjoint (p_2) $\Leftrightarrow$ fabs (p_1. Center ( ). X ( ) $-$ p_2. Center ( ). X ( )) > (p_1. Size ( ) + p_2. Size ( ))/2 $\vee$ fabs (p_1. Center ( ). Y ( ) $-$ p_2. Center ( ). Y ( )) > (p_1. Size ( ) + p_2. Size ( ))/2)

Contains(anotherPixel:Pixel):Integer

如果一个栅格点 p_1 包含另外一个栅格点 p_2，则返回为真值(true)，栅格点包含关系的定义如下：

p_1. Contains (p_2) $\Leftrightarrow p_1$. Size ( ) $\geqslant p_2$. Size ( ) $\wedge$ (fabs (p_1. Center ( ). X ( ) $-$ p_2. Center ( ). X ( )) $\leqslant$ (p_1. Size ( ) $-$ p_2. Size ( ))/2 $\wedge$ fabs (p_1. Center ( ). Y ( ) $-$ p_2. Center ( ). Y ( )) $\leqslant$ (p_1. Size ( ) $-$ p_2. Size ( ))/2)

Within(anotherPixel:Pixel):Integer

如果一个栅格点 p_1 被另外一个栅格点 p_2 包含，则返回为真值(true)，栅格点被包含关系的定义如下：

p_1. Within (p_2) $\Leftrightarrow p_2$. Contains (p_1)

Equals(anotherPixel:Pixel):Integer

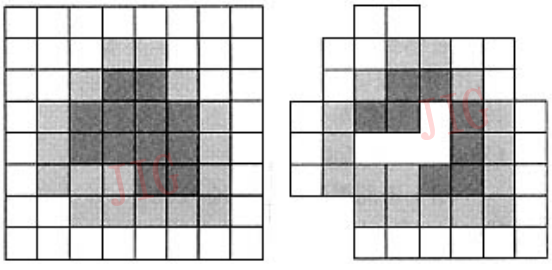
如果一个两个栅格点 p_1,p_2 在地理位置上重合，则返回真值(true)，栅格点重合的定义如下：

p_1. Equals (p_2) $\Leftrightarrow$ p_1. Center ( ). Equals (p_2. Center ( )) $\wedge$ p_1. Size ( ) = p_2. Size ( )

} **ADT Pixel**

2.2 RasterRegion 类型

由于栅格数据描述了在某一特定地理空间上连续分布的现象，如高程、温度、气压等，因此地理栅格数据数据可以使用一个连续的区域来表达，该区域不仅记录了栅格数据本身，同时还记录了栅格数据分布的范围等信息。在研究中，称该数据类型为 RasterRegion(图 3)。



(a) 规则栅格区域 (b) 不规则栅格区域

图 3 RasterRegion 数据对象示意

Fig. 3 Object for RasterRegion data type

RasterRegion 类型的定义如下：

ADT RasterRegion

{

数据对象：

RasterRegion 类型对象是 Pixel 对象的集合，它至少包含一个像素。设 p 为 Pixel 类型对象，$\boldsymbol{R}^{\text{region}}$ 为 RasterRegion 类型对象，则 RasterRegion 类型对象满足以下约束条件：

(1)连通约束，定义如下：

$\forall p_i, p_j \in \boldsymbol{R}^{\text{region}}$, $\exists$ Path (p_i,p_j)

Path (p_i, p_j) $\Leftrightarrow$ { p_i, p_{i+1}, p_{i+2}, $\cdots$, p_j | p_k. OrthoTouch (p_{k+1}) = 1, $i <= k < j$}

(2)分辨率约束，定义如下：

$\forall p \in \boldsymbol{R}^{\text{region}}$, p. Size ( ) = $\boldsymbol{R}^{\text{region}}$. Resolution ( )

(3)数据类型约束，定义如下：

$\forall p \in \boldsymbol{R}^{\text{region}}$, p. ValueType ( ) = $\boldsymbol{R}^{\text{region}}$ ValueTypeDefinition ( )

基本操作：

Resolution( ): double

用于得到栅格区域的分辨率。

ValueTypeDefinition( ): RasterValueType

用于得到栅格区域栅格点的值类型。

Geometry(anotherPixel: Pixel): Polygon

用于得到一个 RasterRegion 对象在二维空间中的范围(图 4),由于 RasterRegion 类型存在连通约束,因此其范围可以用一个 Polygon 对象表达,对于规则 RasterRegion 数据(图 3(a)),其范围为一个矩形;而对于非规则的栅格区域,其范围为可以带一个或多个"岛"的多边形。

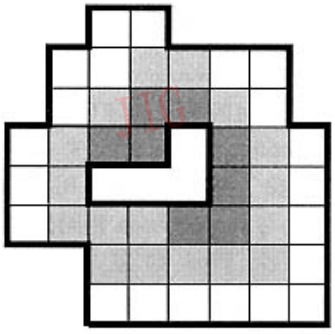


图 4　栅格区域的范围示意(不同的灰度表示栅格值不同)

Fig. 4　Geometry operator for RasterRegion object (Different gray scales represent different values)

假定栅格区域范围为 S_l,则有

$S_l = \min\{g \mid \forall p \in \boldsymbol{R}^{\text{region}}, g.\ \text{Contains}(p.\ \text{Geometry}())\}$

此处 Contains 操作直接采用 OpenGIS® 的定义。

Interpolation(x: double, y: double): Variant

由于栅格数据是场模型的一种表达,因此可以根据栅格数据通过内插来得到空间中任意一点的数值。如果 Point(x,y)在 RasterRegion 的边界内,则使用特定方法来插值;如果 Point(x,y)在 RasterRegion 的边界外,则返回无效值。通常可以选择的内插方法有最邻近点法、双线性内插法、双三次褶积法等,参照文献[8],其定义如下:

```
enum InterpolationMode{
        rim_ nearestneighbor, rim_ linear, rim_ bilinear, rim_ bicubic, rim_ lostarea, rim_ barycentric, rim_piecewiseconstant, rim_none
};
```

栅格数据类型限定了可以采取的内插方式,如对于命名型栅格数据只能采用最邻近点法进行内插。

Statistic (statMode: RasterDataStatMode): Double

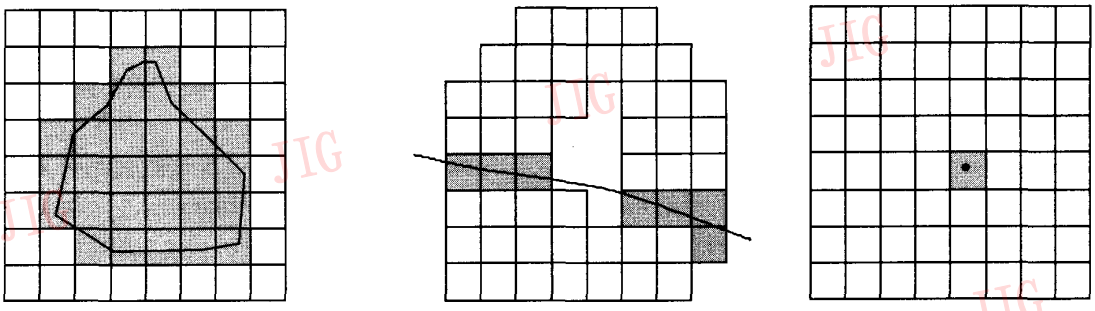
针对给定的栅格区域对象,返回某个特定的统计项。

```
enum RasterDataStatMode{
        rds_ max, rds_ min, rds_ mean, rds_ mode, rds_median
};
```

其中,rds_max 表示求最大值;rds_min 表示求最小值;rds_mean 表示平均值;rds_mode 表示众数;rds_median 表示中位数;同样,栅格数值类型也限定了可以进行的统计项,例如对命名型数据,平均值、极值分析是无意义的。

Subset (subRegion: Geometry): RasterCoverage

通过一个几何体 subRegion 在 RasterRegion 中得到子栅格区域(图 5),该子区域可以是一个连通的 RasterRegion,也可以是一个或多个 RasterRegion 的集合,即 RasterCoverage 对象。



(a) 面的 Subest 操作　　(b) 线的 Subset 操作　　(b) 点的 Subset 操作

图 5　RasterRegion 对象的 Subset 操作

Fig. 5　Subset operator of RasterRegion object

假定栅格区域为 R，几何体为 g，Subset 运算结果为 $\boldsymbol{R}_{\text{subset}}$，有：

$$\boldsymbol{R}_{\text{subset}} = \min\{\boldsymbol{x} \mid \forall p \in \boldsymbol{x}, p \in \boldsymbol{R}^{\text{region}} \wedge \boldsymbol{x}.\text{Geometry}().\text{Contains}(g')\}$$

其中，$\boldsymbol{x}$ 为 RasterCoverage 类型，而 $g' = \boldsymbol{R}^{\text{region}}.\text{Geometry}().\text{Imtersection}(g)$

Union（anotherRasterRegion：RasterRegion，oper：OperatorFlag）：RasterRegion

根据特定的规则，合并两个栅格区域，两个栅格区域能够进行合并操作的前提是：

$$\boldsymbol{R}_1^{\text{region}}.\text{Resolution}() = \boldsymbol{R}_2^{\text{region}}.\text{Resolution}() \wedge \exists p_1 \in \boldsymbol{R}_1^{\text{region}}, p_2 \in \boldsymbol{R}_2^{\text{region}}, p_1.\text{Center}().\text{Equals}(p_2.\text{Center}()) \wedge \boldsymbol{R}_1^{\text{region}}.\text{ValueTypeDefinition}() = \boldsymbol{R}_2^{\text{region}}\text{ValueTypeDefinition}()$$

即其分辨率和栅格点数据类型必须一致。在该操作中，oper 参数为运算符标识，其可用 OperatorFlag 枚举类型定义，它包含了常用的算术、关系以及逻辑等双目运算符。如前所述，栅格数据的类型同样约束了可以进行的操作类型。值得指出的是，Union 操作中，两个栅格区域可以不一致，而存在差异的区域将被视为无效值。

Evolve（rule：EvolveRule，Times：Integer）：RasterRegion

Evolve 操作用于实现栅格数据的邻域变换，由于 4 邻域是 8 邻域的子集，因此可以基于 8 邻域来定义操作，其中，rule 为针对单点的变换规则，它可被定义为一个过程，其接口为

void EvolveRule（[in] VARIANT * pData，[out] VARIANT resultValue）；

其中，pData 为中心栅格点以及 8 邻域栅格点的数值，resultValue 为根据变换规则计算得到的结果，而 Evolve 操作中的 Times 参数为进行变换的次数。

GeoTransform（xTrans：GeoTransRule，yTrans：GeoTransRule）：RasterRegion

GeoTransform 用于实现栅格区域的几何变换，其中二维栅格数据几何变换可以表示为

$$g(x,y) = f(x',y') = f[a(x,y), b(x,y)]$$

其中，$g(x,y)$ 为输出，而 $f(x,y)$ 为输入，因此，在 GeoTransform 操作中，xTrans 和 yTrans 分别为两个方向的几何变换规则，为接口定义如下的 GeoTransRule 类型：

void GeoTransRule（[in] double x，[in] double y，[out] double result）；

为了便于栅格数据的计算，通常 xTrans 和 yTrans 采用反变换规则。

} **ADT RasterRegion**

2.3 RasterCoverage 类型

在现实世界中，栅格数据所表达的现象并非是完全无缝拼接在一起的。由于天然的屏障、人为行政边界以及研究对象实际分布的形态等因素，致使只在局部区域内保持连通的栅格数据，才可以用一个 RasterRegion 对象表示；而由多个这样栅格区域组成的一个集合才能更好地表达实际现象的空间分布，并可采用 RasterCoverage 表达。RasterRegion 与 RasterCoverage 之间的关系问题可以使用组合设计模式（composite pattern）[9] 来解决，容器 RasterCoverage 除了提供遍历元素 RasterRegion 的操作外，还要提供与 RasterRegion 相同的接口。因此，设 p_i 为第 i 个 Pixel 对象，$\boldsymbol{R}_i^{\text{region}}$ 为第 i 个 RasterRegion 对象，$\boldsymbol{R}_i^{\text{coverage}}$ 为第 i 个 RasterCoverage 对象，g_i 表示第 i 个几何体对象，$i,j,k \in \mathbf{N} \cup \{0\}$，则 RasterCoverage 主要有以下操作：

（1）基本数据属性与操作

RasterCoverage 是 RasterRegion 的集合类，即一个 RasterCoverage 对象包含了至少两个互不重叠或相邻的 RasterRegion 对象，并且与 RasterRegion 类似，同样需要满足类型约束、分辨率约束，而对于所包含 RasterRegion 对象的互不重叠和相邻约束的定义如下：

$$\forall \boldsymbol{R}_1^{\text{region}}, \boldsymbol{R}_2^{\text{region}} \subset \boldsymbol{R}^{\text{coverage}}, \boldsymbol{R}_1^{\text{region}} \cap \boldsymbol{R}_2^{\text{region}} = \varnothing \wedge \forall p_1 \in \boldsymbol{R}_1^{\text{region}}, p_2 \in \boldsymbol{R}_2^{\text{region}}, p_1.\text{OrthoTouches}(p_2) = \text{false}$$

由于所有 RasterRegion 的操作都适用于 RasterCoverage，RasterCoverage 的操作可以认为是其包含的每个 RasterRegion 对象的对应操作的并，因此不再详细列出 RasterCoverage 的同类操作。

（2）集合元素遍历操作

RasterCoverage 对集合元素的遍历操作主要有：

GetElementAt（nIndex：Integer）：RasterRegion

该操作通过索引返回一个子 RasterRegion 对象，还可以提供更复杂的搜索与遍历接口。

RemoveElementAt（nIndex：Integer）

该操作按照索引删除一个子 RasterRegion 对象。

(3) 表现操作

数据表现操作是指将地理栅格数据转换为特定图像格式,常见的栅格数据表现方式有调色板方式与假彩色合成方式。RasterCoverage 提供的数据表现操作为:

ExportImage (tagFormat : ImageFileFormat, renderer: Renderer) : IStream

该操作通过指定的渲染方式(rendererer)以及指定的输出图像格式(tagFormat)将栅格数据输出为二进制数据流。其中,系统支持的图像格式在枚举类型 ImageFileFormat 中的具体定义如下:

```
enum ImageFileFormat{
    BMP, JPEG, GIF, TIFF,GeoTIFF
};
```

在网格覆盖类型实现中,Renderer 是所有渲染方法的基类,可以派生出相应的基类,如表示调色板渲染方法的 PaletteRenderer 以及表示假彩色合成的 PesudocolorRenderer 等。在文献[2]中,指出空间查询语言最好能够支持数据表达,但考虑到空间数据访问和分析操作是重点,对于栅格数据只给出了 ExportImage 一个操作接口。

3　GSQL-R 描述

在 SQl99/SQL3 中支持自定义数据类型及针对自定义数据类型的操作。根据上述的 3 种基本数据类型描述,下面给出支持栅格数据 GSQL 扩展所需的 3 个方面的内容,即数据定义语言(data definition language, DDL)、数据表结构定义和数据操作语言(data manipulation language, DML)。

3.1　数据定义语言

(1) 像素(Pixel)

在 SQL99/SQL3 中,一般都是采用脚本方式来表示自定义数据类型,这里可以使用类似 OpenGIS® 的 WKT(well-known text)格式表示一个像素点。像素的 DDL 表达如下:

```
create type sdo_raster_pixel_type as object (
    sdo_raster_center sdo_point_type,
    sdo_raster_size number,
    sdo_raster_type varchar2(32),
    sdo_ratser_value varchar2(32));
```

例如:p_1 = sdo_raster_pixel_type(sdo_point_type(100,100), 10, 'r_numerical', '1.2')表示一个中心在(100,100)处,尺寸为 10 的一个像素点,值的类型为数值类型,值大小为 1.2。

(2) 栅格区域(RasterRegion)和栅格覆盖(RasterCoverage)

对于 RasterRegion 与 RasterCoverage 数据而言,除了定义 Pixel 的数值类型、分辨率、参照系统等内容外,还需要表达其中每个栅格点的取值。为了便于管理以及减少数据冗余,并顾及栅格区域的不规则特征,在其 DDL 中,采用分行存储每个栅格点的方式。因此,栅格区域可以采用如下数据定义:

```
create type sdo_raster_region_type as object (
    sdo_raster_lt sdo_point_type,
    sdo_raster_size number,
    sdo_raster_type varchar2(32),
    sdo_srid number,
    sdo_raster_rowdata sdo_ratser_rowdata_array);
```

其中,sdo_raster_lt 为栅格数据中(0,0)点所对应的空间位置,因为栅格区域的分辨率约束,所以可以只采用两个整数值(即行列号)来表达每个 Pixel 的位置;sdo_raster_size 为每个栅格点的大小;sdo_srid 为空间参照系统的标识;sdo_ratser_rowdata 存储了栅格点的数值,sdo_raster_rowdata_array 为 sdo_ratser_rowdata_type 的数组,而后者定义为

```
create type sdo_ratser_rowdata_type as object (
    sdo_raster_startlocationx number,
    sdo_raster_startlocationy number,
    sdo_raster_rlcdata sdo_raster_rlcdata_array);
```

其中,sdo_raster_rlcdata 表示采用行程长度编码(run length coding, RLC)来表达栅格数据,它是 sdo_raster_rlcdata_type 类型的数组,而 sdo_raster_rlcdatatype 定义为

```
create type sdo_raster_rlcdata_type as object(
    sdo_raster_value varchar2(32),
    sdo_raster_runlength number);
```

而栅格覆盖是栅格区域的数组,如图 6 所示的一个地区栅格图像,在 GSQL-R 中的表示方法为(其中黑线表示栅格图像范围,空白区域为无效值(在 GSQL-R 中用'nul'表示),整个 RasterCoverage 由两个 RasterRegion 对象组成):

```
rc1 = sdo_raster_coverage_type(
        2,
    sdo_raster_region_type(
        sdo_point(0,0),10,8307,
    sdo_ratser_rowdata_array(
```

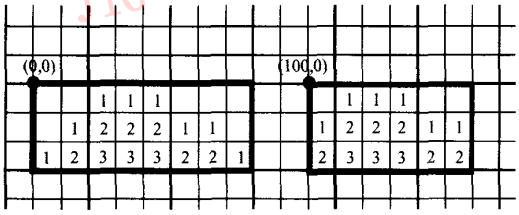


图 6 RasterCoverage 在 GSQL-R 中的表示示例

Fig. 6 GSQL-R representation of RasterRegion object

```
        sdo_ratser_rowdata_type(0,0,sdo_raster_
        rlcdata_array('nul',2,'1',3,'nul',3),
        sdo_ratser_rowdata_type(1,0,sdo_raster_
        rlcdata_array('nul',1,'1',1,'2',3,'1',2,
        'nul',1),
        sdo_ratser_rowdata_type(2,0,sdo_raster_
        rlcdata_array('1',1,'2',1,'3',2,'2',2,
        '1',1))
    ),
    sdo_raster_region_type(
      sdo_point(100,0),10,8307,
    sdo_ratser_rowdata_array(
        sdo_ratser_rowdata_type(0,0,sdo_raster_
        rlcdata_array('nul',1,'1',3,'nul',2),
        sdo_ratser_rowdata_type(1,0,sdo_raster_
        rlcdata_array('1',1,'2',3,'1',2),
        sdo_ratser_rowdata_type(2,0,sdo_raster_
        rlcdata_array('2',1,'3',3,'2',2))
    )
);
```

3.2 数据表结构定义

在数据库存储中，一般都是使用 WKB(well-known binary)来存储非结构化的信息，类似 OGC 规定的矢量地物存储方法，而且针对栅格数据同样可以定义其 WKB 结构。与矢量数据不同，由于栅格数据具有“位置隐含，属性明显”的特点，因此在数据存储中，一个栅格覆盖中的全部栅格点已经描述了栅格数据的主要信息，而在矢量数据存储中则通常需要一个几何体类型列和一组属性列来表述一个矢量图层[10]。在支持地理栅格数据的空间数据库中，一个 RasterCoverage 对象对应于一个关系数据库表的一条记录。在数据表中，必须包含一个 sdo_raster_coverage_type 类型的数据列，而其他列则可以存储栅格数据的元数据信息。在一个数据表中，还可以用多条记录存储 RasterCoverage 对象，每条记录对应于栅格覆盖的一个子集。图 7 给出了符合上

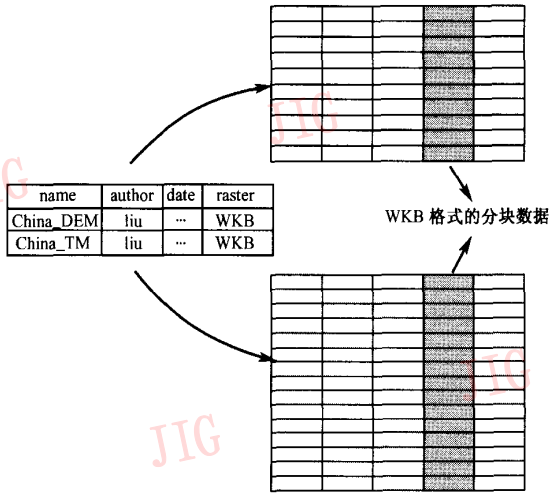


图 7 栅格数据表的结构

Fig. 7 Table's schema with raster data column

述描述的一个栅格数据表的结构，它包括 name、author、date 和 raster 4 个列，其中 Raster 列用于存储栅格数据。

创建图 7 所示栅格数据表的 SQL 语句如下：

```
create table rastertable(
    name varchar(32),
    author varchar(32),
    date datetime,
    raster sdo_raster_coverage_type);
```

为了向上述数据表中添加如图 6 所示的一条栅格数据记录，其语句如下：

```
insert into rastertable values('China_DEM','liu',
'2004-1-1',@rc1)
```

3.3 栅格数据操作及实例

由于在已有数据类型和数据表定义的基础上，实现栅格数据的操作相对较为简单，因此下面主要是通过列举实践中的一些常见操作，并使用GSQL-R来描述这些操作，以说明 GSQL-R 的完备性与应用性。由于 select-from-where 子句在查询语言中的重要性，所以以下实例均为该子句形式。

(1) 将某个地区的数字高程模型输出为图像，图像格式为 TIFF

```
select exportimage(raster,'tiff',paletterenderer)
    from rastertable
    where name='China_DEM'
```

(2) 统计出某个地区数字高程模型数据的最大值，最小值与平均值

```
select statistics(raster,'rds_max'),statistics(raster,
```

'rds_min'), statistics(raster, 'rds_meam')
from rastertable
where name = 'China_DEM'

(3) 通过已知的边界(福建省)对 RasterCoverage 进行剪裁

```
select subset(a.raster, b.geometry)
    from rastertable a, admintable b
    where a. name = 'China_DEM' and b. name
    = 'Fujian'
```

(4) 在某地数字高程模型数据中,求取给定点(有地理坐标或者行列号标识)的高程值

```
select interpolate(raster, sdo_point_type(100,100))
    from rastertable
    where name = 'China_DEM'
```

(5) 求取某条河流(长江)流经区域的高程数据

```
select subset(a.raster, b.geometry)
    from rastertable a, river b
    where a. name = 'China_DEM' and b. name
    = 'Changjiang'
```

(6) 求取海拔大于 2 000m 的中国省会名称

```
select b.name
    from rastertable a, captital b
    where value(interpolate(a. raster, b. geometry))
    >2 000 and a. name = 'China_DEM'
```

(7) 对某地数字高程模型数据进行坡度分析,并生成坡度数据

```
select evolve(raster, slope, 1)
    from rastertable
    where name = 'China_DEM'
```

通过上述的实例表述可以看出,GSQL-R 能够很好地支持针对栅格数据以及矢量/栅格交互的大部分查询分析操作,从而为不同格式的空间数据访问提供了一致的访问接口。

4　结论与展望

在空间数据库已经成为 GIS 软件开发中基本数据管理手段的今天,需要研究基于空间数据库的空间数据访问规范,以便为数据共享和互操作提供便利。目前许多学者的研究主要是针对矢量数据,而本文参照 OpenGIS®,通过定义支持栅格数据的 SQL 扩展——GSQL-R,以便为进行地理栅格数据的访问和操作提供一致的访问接口。本文进行的工作主要包括以下几个方面:

(1) 给出了支持地理栅格数据操作描述的 3 种基本数据类型,即包括 Pixel、RasterRegion 和 RasterCoverage;

(2) 定义了上述数据类型的数据对象和基本操作;

(3) 基于 3 种基本数据类型,描述了地理栅格数据的数据定义和数据操作 SQL 扩展;

(4) 考虑到实际 GIS 应用中的需求,给出了一些常见栅格空间分析的 GSQL-R 表达。

本文的工作有助于地理栅格数据的共享和互操作,并且为矢量以及栅格数据提供一致的访问规范奠定了良好的基础。在下一步工作中,需要根据 GSQL-R,设计并实现一个支持地理栅格数据的空间数据库原型系统。

参考文献(Reference)

1 Scholl M, Woisard A. Advances in spatial databases[M]. Berlin: Springer-Verlag, 1997.

2 Libkin L, Machlin R, Wong L. A query language for multidimensional arrays: design, implementation, and optimization techniques[J]. SIGMOD(Special interest group on management of data) Record, 1996,**25**(2):228 ~ 39.

3 Egenhofer M J. Spatial SQL: A query and presentation language[J]. IEEE Transactions on Knowledge and Data Engineering, 1994,**6**(1): 86 ~ 95.

4 Fang Y, Chu F, Chen B. Spatial structural query language-G/SQL [J]. Journal of Image and Graphics, 1999,**4**(11):901 ~ 910. [方裕,楚放,陈斌. 空间结构化查询语言——G/SQL[J]. 中国图象图形学报, 1999,**4**(11):901 ~ 910.]

5 Fang Y, Chu F. Spatial query optimization[J]. Journal of Image and Graphics, 2001,**6**(4):307 ~ 314. [方裕, 楚放. 空间查询优化[J]. 中国图象图形学报,2001,**6**(4):307 ~ 314.]

6 Open GIS Consortium Inc. OpenGIS simple features specification for SQL 1. 1[S/OL], http://www. opengis. org/ docs/99-049. pdf, 1999.

7 Baumann P. Web-enabled raster GIS service for large image and map database[A]. In: Proceedings of the IEEE Computer Society 12th International Workshop on Database and Expert Systems Applications [C], Munich, Germany, 2001: 870 ~ 874.

8 Open GIS Consortium Inc. Web coverage service (WCS) 1. 0. 0 [S/OL]. http://www. opengis. org/ docs/03-065r6. pdf, 2003.

9 Gamma E, Helm R, Johnson R, *et al.* Design patterns: elements of reusable object-oriented software[M]. Boston, MA, USA: Addison-Wesley Publishing Company, 1995.

10 Oracle Corporation. Oracle spatial user's guide and reference release 9. 0. 1 [EB/OL]. http://download-west. oracle. com/docs/pdf/A96630_01. pdf, 2001, 6.