第 37 卷 第 2 期
Vol.37 No.2

计 算 机 工 程
Computer Engineering

2011 年 1 月
January 2011

• 软件技术与数据库 •　　文章编号：1000—3428(2011)02—0027—03　　文献标识码：A　　中图分类号：N945

一种用于网络地图的栅格索引技术

赵艳伟[1,2]，程振林[1]，董 慧[1,2]，方金云[1]

(1. 中国科学院计算技术研究所，北京 100190；2. 中国科学院研究生院，北京 100049)

摘 要：分析目前常用的空间检索技术在网络地图服务器中的局限性，提出一种利用 AGG 渲染引擎生成栅格索引的技术，即两素(要素-像素)对应方法，并分析该方法的设计与实现过程，给出其与 R-树效率对比的实验评估。实验结果证明，两素对应栅格索引查询效率的平均性能优于 R-树约 22%。

关键词：地理信息系统；空间索引；R-树；网络地图

Grid Index Technology Applied in Web Map

ZHAO Yan-wei[1,2], CHENG Zhen-lin[1], DONG Hui[1,2], FANG Jin-yun[1]

(1. Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China;
2. Graduate University of Chinese Academy of Sciences, Beijing 100049, China)

【Abstract】The R-tree method, which is a popular method used in spatial database, has limitations for Web map service. This paper proposes a grid index method utilizing Anti-Grain Geometry(AGG) as the engine for the map between geographic feature and image pixel. After illustrating the design and implementation of this method, it conducts experiments to compare the performance of R-tree and the proposed method. The result shows the significant performance advantage about 22% of the new method.

【Key words】 GIS; spatial index; R-tree; Web map

DOI: 10.3969/j.issn.1000-3428.2011.02.010

1 概述

随着计算机技术及空间技术的飞速发展，地理信息系统(Geographic Information System, GIS)[1]技术在生产与生活中发挥着越来越重要的作用。电子地图作为空间信息可视化产品之一，其主要任务之一是有效地在后台服务器检索空间数据及快速响应前台用户的在线查询需求。

R-树是一种经典的空间索引结构，它采用最小约束矩形(Minimum Bounding Rectangles, MBR)法来近似表示空间对象。R-树具有自动平衡、空间利用率高等优点。然而由于中间节点目录矩形允许且可能重叠，因此在查询的过程中往往存在多条路径，并且其中的某些路径不包含查找结果，这就影响了查找性能[2]。尤其是面向并发用户的网络地图服务，一般采用预发布栅格瓦片的方式来生成地图。面向桌面 GIS 的空间索引技术在网络地图服务应用环境中已经不适用，建立一种高效的空间索引技术显的尤为重要。

本文提出一种基于 AGG[3]绘图引擎的两素(要素-像素)对应栅格索引方法。两素对应就是矢量要素与图像像素对应。该方法能够避免搜索存在重叠区域这种情况。

2 索引栅格渲染过程

索引栅格渲染过程如下：

步骤 1 为了使矢量要素点坐标精确到亚像素精度，采用如下转换：

[浮点数据整数部分]＝(浮点数据<<8)>>8

[浮点数据小数部分]＝(浮点数据<<8)& 0xFF

该做法能够将浮点坐标的小数部分转换到 0~255 区间，这种坐标转换的优点是考虑了小数部分对像素的栅格单元权值(cover)的影响，便于利用亚像素精度进行反走样。

步骤 2 轮廓扫描运用亚像素精度(subpixel accuracy)的 Bresenham 生成直线算法。与传统 Bresenham[4]不同的是，该算法利用误差判别选择像素的过程是基于亚像素的，即将单位栅格平均分成了 256×256 个子像素。与横纵栅格线交点的计算非仅仅依赖于斜率将原坐标(x_i, y_i)进行增量(delta)运算，而是需要转换成亚像素精度。描述如下：

$$delta = (256 - fy) \times \frac{\Delta x}{\Delta y} \tag{1}$$

其中，fy 为实数坐标分量 Y 部分经过坐标转换通道所得的小数部分；$\triangle x = x2 - x1$, $\triangle y = y2 - y1$，如图 1 所示。

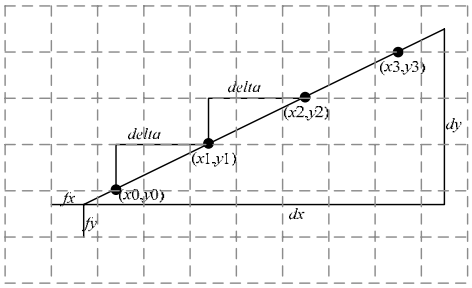


图 1 增量计算

因此，计算受该直线所影响的整数栅格点的 *cover* 和 *area* 的描述如下：

$$cover = fy2 - fy1 \tag{2}$$

$$area = (fx2 + fx1) \times cover \tag{3}$$

基金项目：国家“863”计划基金资助项目(2009AA12Z220)

作者简介：赵艳伟(1985－)，女，硕士研究生，主研方向：地理信息系统，网络地图服务器；程振林，博士；董 慧，博士研究生；方金云，副研究员、博士

收稿日期：2010-06-30　**E-mail：**zhaoyanwei@ict.ac.cn

轮廓扫描后，渲染控制器开始计算。具体步骤为：

(1)将生成轮廓的单元格(cell)排序；

(2)按照行扫描顺序从小到大逐行扫描；

(3)利用图形对每个扫描栅格点的覆盖面积 *area* 判断当前单元是否填充。最后，便可把当前要素的栅格信息存入渲染缓存中。

3 两素对应的索引解析过程

两素对应就是矢量要素与图像像素对应，具体指在渲染图形要素的同时，生成一张与原图匹配的栅格底图，此图像上的每个像素点都对应 2 个部分内容：(1)该真实图像的颜色(原图)；(2)该要素索引号 ID 转换为 RGB 的颜色。

3.1 索引生成

索引结构的生成运算与地图数据的渲染过程是同步的。首先读取 shp 格式地图数据，进而分解出单独要素信息及该要素索引信息，从而生成某个地图数据的索引底图。伪代码描述如下：

```
Getfeature()
{
利用 GDAL 库获取数据源 OGRDataSource
*poDS
//获取该数据源的图层 poLayer 及该图的外包 Env
poLayer=poDS->GetLayer(0)
poLayer->GetExtent(&Env,true)
根据外包计算世界坐标向屏幕坐标转
公式
//循环读取该图层的要素 OGRFeature*poFeature
While(poFeature =
poLayer->GetNextFeature()!=NULL)
{
定义 OGR 几何体 OGRGeometry *geom
获取要素索引信息 FeatureId
 将 FeatureId 解析为 R、G、B 分量
 根据要素的点序列及要素 RGB 填充要素
}
}
```

上述做法的核心是：要素 ID 序号转换为 RGB 颜色分量。颜色格式采用 RGB24，即使用 24 位来表示一个像素，*R*、*G*、*B* 每个分量都用 8 位表示，取值范围为 0~255。解析过程采用的做法是将十进制 ID 转换为二进制，先从 B 分量的低 8 位开始存，共能存 256 个 ID，超过部分存入 *G* 分量，再超过部分存入 *R* 分量。因此，在利用渲染控制器进行多边形填充的过程中，根据如上计算的 ID 颜色，将每个像素的 RGB 分量存入渲染缓存中。最后保存为图像文件。

3.2 索引搜索运算

对于点查询，只需根据(*X*,*Y*)点获得此点颜色值并解析出 ID。对于框查询，只需根据框在缓存中的位置，利用行扫描循环取出相交栅格的颜色进行解析即可。解析的代码如下：

```
//获取渲染好的画布
unsigned char * b =(unsigned char *)this->bufId;
//获取 R、G、B 分量
  unsigned int red = b[(height – point_y) * width * 3 + point_x * 3];
  unsigned int green = b[(height - point_y) * width * 3 + point_x * 3+1];
  unsigned int blue = b[(height - point_y) * width * 3 + point_x * 3+2];
```

当分别取出 *R*、*G*、*B* 分量后再采用和生成颜色相反的方式，组合得到要素的 ID。

3.3 索引效率分析

由于 RGB24 是用 24 位来表示颜色，因此将要素 ID 转换为颜色值进行存储，能表示 16 777 216 个要素，对于一般的地图数据已经足够。基于此索引进行点/框查询的时间复杂度是不变的，即 $O(n)$。但是对于 R-树，随着索引数据量的增加，树的高度及 R-树中间结点目录矩形的重叠均会增加，查找性能将会下降。经过以上分析，采用两素对应的栅格索引机制比 R-树更具有优势。

4 性能对比

为了对比 R-树和两素对应的索引行为，本文设计 3 个实验针对以下指标来测试它们的性能：2 种索引的过滤精度，建立索引所花费的时间及内存，2 种索引查询的效率。实验的测试环境：操作系统为 WindowsXP(Service Pack 2)，CPU 为 Intel Pentium M 1.73 GHz，内存为 1 GB。测试数据如表 1 所示。

表 1 测试数据集

编号	名称	几何类型	大小/MB	要素数目
D1	BOUND_poly.shp	MultiPolygon	17.30	3 407
D2	CS_99_03c.shp	MultiPolygon	1.48	123

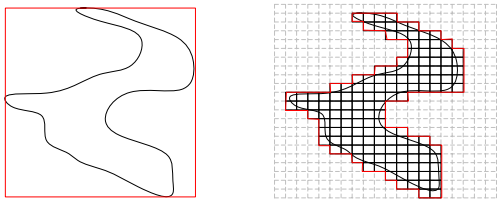
4.1 过滤精度对比

该实验采用的方法是判断给定随机点落在要素多边形内的概率。为了进行比照还将选一种基于矢量的计算几何的方法来作为参照。这种计算几何方法采用射线算法[5]能够精确判断所查询的随机点的索引号，即是否属于某一要素。实验采用的数据为表 1 测试数据集中的 D2。实验结果如表 2 所示。

表 2 过滤精确率对比

随机数据	计算几何	R-树	两素对应	两素对应筛选率/(%)	R-树筛选率/(%)
10 000	667	721	1 026	92.5	65.0
20 000	1 355	1 419	2 043	95.5	66.3
40 000	2 609	2 704	3 938	96.5	66.3
60 000	3 931	4 075	5 912	96.5	66.5
80 000	5 211	5 411	7 843	96.3	66.4
100 000	6 512	6 739	9 799	96.6	66.5
平均筛选率/(%)				95.6	66.2

从实验结果可以看出：两素对应所检索出的结果中的 95%是最终精确结果，比 R-树的过滤集精度更高一些。产生这种差异的原因是，R-树空间目标采用 MBR 来加以近似表达，如图 2(a)所示，而且允许兄弟结点之间相互重叠，因此对于精确匹配查询，R-树不能保证唯一的检索路径和唯一的检索结果。而本文提出的两素对应索引机制由于采用的是栅格方法，它的误差只产生在轮廓线的边沿像素，如图 2(b)所示。



(a)R-树过滤边界　　(b)两素对应过滤边界

图 2 过滤边界比较

4.2 索引建立性能对比

将 R-树与两素对应分别对建立索引的时间与内存占用做对比。实验数据为表 1 测试数据集中的 D1。实验结果如图 3 所示。可以看出，R-树建立索引占用的内存会随着数据量的增大而增加，而两素对应由于是在内存中生成 *Bitmap*，只要长度与宽度指定，它建立索引占用的内存不会随着数据量的增大而增加，由此看来，两素对应更适合大数据量的索引建立。从在建立索引占用的时间来看，R-树所用时间要少一些，但

在实际电子地图服务器的应用中，索引是预生成的。因此，该方法更适用于在电子地图服务器应用。

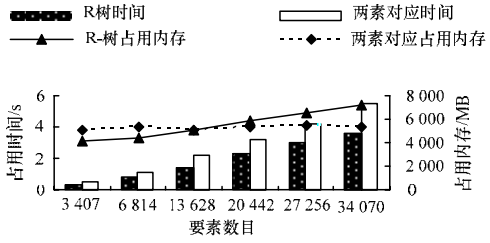


图 3　建立索引对比

4.3　查询性能对比

为了实验结果的客观性，将分别对密集数据和稀疏数据做测试。密集数据采用的是表 1 测试数据集中的 D1，该面图层的重叠区域较多。而稀疏数据采用的是表 1 测试数据集中的 D2，该面图层的重叠部分较少。点查询的实验对比如表 3 所示。结果表明，无论是对密集数据还是稀疏数据，两素对应的点查询效率更高一些，因为这种索引机制是将每一点的索引保存在 *bitmap*，点查询时，只需根据点坐标计算该点在 *bitmap* 中的行列地址，便可根据颜色信息解析出该点索引，时间复杂度是一维线性的 $O(n)$，而对于 R-树，无论是密集数据还是稀疏数据都要从根结点向下搜索，当索引数据量增加时，这种索引效率就会降低。

表 3　查询对比

数据(点查：10^6；框查：10^5)			1	2	4	6	8	10
点查询/s	稀疏	R-树	8.54	17.01	34.53	51.45	68.25	85.17
		两素	8.01	15.99	32.11	47.86	64.23	79.91
	密集	R-树	8.69	17.12	33.96	51.41	68.12	85.71
		两素	7.92	15.94	31.71	47.69	64.02	81.10
框查询/s	稀疏	R-树	0.93	1.86	3.69	5.58	7.36	9.29
		两素	0.94	1.91	3.78	5.92	7.60	9.64
	密集	R-树	4.10	9.18	16.62	24.65	33.09	42.34
		两素	0.94	1.89	3.82	6.22	7.62	10.33

框查询的实验结果如表 3 所示。查询框的大小为 200×200。结果表明，针对密集数据的框查询，两素对应的效率明显高于 R-树索引效率，因为数据越密集，*MBR* 的重叠越多，这样在框查询时，R-树搜索的路径不唯一，导致效率下降。针对稀疏数据的框查询，两素对应的效率却没有 R-树高，原因是：由于数据稀疏，MBR 的重叠少，R-树的效率将有所提升，而两素对应法以行扫描方式检索底图，循环时间将影响两素对应索引的整体效率。由于网络地图应用场景中，尤其是城市电子地图应用中，绝大部分情况要素都较为密集，因此此种情形不会影响两素对应算法在实际应用中的性能表现。

5　结束语

电子地图是 GIS 应用的一个热门领域，其中，快速响应用户在线查询效率，提高空间索引的查询效率是关键。本文介绍被广泛应用的 R-树空间索引方法的特点，分析目前常用的空间检索方法在网络地图服务器中的局限性，提出一种可用于网络地图的高效的栅格索引技术，即两素(要素-像素)对应方法，着重分析其栅格渲染方法及索引解析方法，并针对不同情况数据进行了实验对比。实验结果表明，两素对应索引机制适用于大数据量的空间信息，并且在过滤精度及密集数据的查询效率上整体优于 R-树。两素对应索引机制相对于精确的计算几何方法，是存在一定误差的，误差产生在要素轮廓的边沿像素，栅格越小，误差也将越小，但栅格细化的过程却会使扫描效率降低，如何平衡这两者间的关系将是本文的后续工作。同时，如何将本文提出的两素对应索引方法高效地应用到网络地图服务器也是今后的研究方向。

参考文献

[1] 郎　兵, 方金云, 韩承德. VegaGIS 可视化系统的设计和实现[J]. 计算机工程, 2008, 34(23): 237-239.

[2] 郭　薇, 郭　菁, 胡志勇. 空间数据库索引技术[M]. 上海: 上海交通大学出版社, 2006.

[3] Maxim S. Anti-Grain Geometry[EB/OL]. (2007-07-08). http://www.antigrain.com/doc/index.html

[4] Bresenham J E. Algorithms for Computer Control of a Digital Plotter[J]. IBM Systems Journal, 1965, 4(1): 25-30.

[5] 金文华, 何涛唐, 卫　清, 等. 简单多边形可见点问题的快速求解算法[J]. 计算机学报, 1999, 22(3): 276-277.

编辑　陈　文

(上接第 26 页)

4:T0->T4->T12->T16

5:T0->T1->T2->T1->T3

6:T0->T4->T5->T6->T5->T14

7:T0->T4->T7->T8->T9->T5->T14

8:T0->T4->T7->T10->T11->T8->T15

9:T0->T4->T12->T13->T5->T14

从以上测试路径可得知，最后生成的转移路径保证每条路径中至少有一条转移是其他路径未使用的(即每一条路径引入一条新的转移)。根据循环复杂度，计算状态图中转移路径条数。环路复杂度的公式为：$V=E-N+2$，其中，E 表示转移的边数；N 表示状态数。图 2 中，$V=17-10+2=9$，与实际寻找的路径数一致。

移路径条数也就是需要设计的测试用例数。可以从每一条测试路径得到一个测试用例，该测试用例用来检查目标程序是否存在这一条路径。

4　结束语

本文提出一种算法从状态图中获得测试路径，根据循环复杂度对测试路径数进行优化，然后由测试路径进一步设计测试用例，由此可生成较少而有效的测试用例，提高测试效率。

参考文献

[1] 许秀林, 胡克瑾. 基于约束满足搜索算法的测试用例生成策略[J]. 计算机工程, 2008, 34(18): 75-78, 84.

[2] Zhu Hong, Hall P A V, May J H R. Software Unit Test Coverage and Adequacy[J]. ACM Computing Surveys, 1997, 29(4): 366-427.

[3] Haworth B, Kirsopp C, Roper M, et al. Towards the Development of Adequacy Criteria for Object-oriented Systems[C]//Proc. of the 5th European Conference on Software Testing Analysis and Review. Edinburgh, UK: [s. n.], 1997: 417-427.

[4] Haworth B. Adequacy Criteria for Object Testing[C]//Proc. of ISQWE'98. [S. l.]: IEEE Press, 1998: 47-56.

[5] Offutt A J. Criteria for Generating Specification-based Tests[C]//Proc. of the 5th IEEE International Conference on Engineering of Complex Computer Systems. Las Vegas, USA: [s. n.], 1999.

[6] 占学德, 繆淮扣. 基于 UML 状态图测试的充分性准则[J]. 计算机科学, 2005, 32(5): 25-49.

[7] 繆淮扣, 费立志. UML 状态图测试充分性准则的功利化评估[J]. 上海大学学报: 自然科学版, 2007, 13(5): 489-496.

编辑　陈　文