# 多核环境下 Hilbert 曲线划分简单要素多边形合并算法*

陈占龙[1]， 吴 亮[1]， 刘焕焕[1,2]

（1. 中国地质大学 信息工程学院，武汉 430074；2. 教育部地理信息系统软件及应用工程中心，武汉 430074）

**摘 要**：为了解决大规模复杂多边形数据合并运算效率问题，减少在空间数据库中检索多边形时的磁盘读取次数，针对多核环境下简单要素类多边形合并的特点，利用 Hilbert 曲线划分方法对简单要素的多边形进行空间数据划分，利用多核处理器并发执行，充分利用了多核计算环境中 CPU 的计算能力，保证了合理的任务分配与充分利用。介绍了该算法中用到的简单要素类多边形合并算子，利用对重合边的判断来进行多边形的合并；最后对提出的算法进行了实验分析。实验证明，本算法在进行大数据的多边形集合合并时效率较高，基于本算法开发的功能用于实际问题中可较好地解决大规模复杂多边形数据层合并运算的效率问题。

**关键词**：多边形合并；Hilbert 曲线；多核计算；简单要素模型

**中图分类号**：TP391 **文献标志码**：A **文章编号**：1001-3695(2012)07-2747-04

doi:10.3969/j.issn.1001-3695.2012.07.095

# Algorithm for simple data model polygon merging based on Hilbert-curve on multi-core CPU

CHEN Zhan-long[1], WU Liang[1], LIU Huan-huan[1,2]

(1. *Faculty of Information Engineering, China University of Geosciences, Wuhan* 430074, *China*; 2. *China GIS Software Research & Application Engineering Center of Ministry of Education, Wuhan* 430074, *China*)

**Abstract**: For solving efficiency problem of large scale complex polygon data merge and decreasing the number of disk-reading while retrieving polygons in spatial database, this paper considered the charactistics of simple element polygon merging in mult-core environment, divided spatial data with Hilbert curve method, made well useful of computing capability of multi-core environment, priserved reasonable task assigments and used of CPU. And it introduced simply elements polygons merge operator, made use of judgment to coincide to the edge. At last, it carried the experiment analysis. Experiments show that this algorithm has high efficiency while merging large data polygon collection. Functions based on this algorithm can well solve the efficiency of operations, bring by large-scale complex polygon data.

**Key words**: polygon merging; Hilbert curve; multi-core programming; simple element mode

矢量图形合并算法是 GIS 空间分析中比较重要的功能。本文中的简单要素类模型是未存储空间拓扑关系模型，其避免了构建拓扑关系的复杂操作，空间对象合并速度比复杂要素模型更快。

多边形形态复杂，是地学应用中的普遍数据[1]。由于 GIS 处理的数据都是有实际地理意义的，进行空间操作和空间分析也必须保持完整的地物特征，地理的位置性造成对矢量空间数据的并行化较其他分析操作更加复杂。如何在多核环境下进行多边形的合并，是近几年研究的热点和难点。首先要进行数据的聚集性划分。现有的空间划分有如下几类方法：传统一维模式的数据划分方法（轮转法（round-robin）、散列划分（hash partitioning）、范围划分（hybrid-range））、基于空间位置的范围划分方法、根据 ID 的直接划分方法等。目前已有一些对于多边形空间分析算法，陈占龙等人[2]采用单调链算法对输入的多边形进行单调处理，以减少多边形比较的次数，并引入平面图的概念和 STR 树空间索引，提高了结果多边形的构建速度。本文给出多核环境下 Hilbert 曲线数据划分方法，较好地保证了空间数据的邻近与聚集性[3,4]；给出简单数据模型基于公共边查找的合并算法，可处理带洞多边形并有较高的效率，尤其在海量复杂多边形数据层合并运算时有较高的效率。

这里先提出对空间实体进行 Hilbert 曲线编码，再进行数据划分，以充分保证多边形的聚集性，再利用本文给出的简单要素多边形合并算法进行合并处理。本文提出的多核环境合并模型保证了合理的任务分配与 CPU 的充分利用，并且给出了多核环境下多边形并行实现方案。实践结果表明，该算法快速、有效，有很好的实践价值。

## 1 多核计算中并行算法的多边形合并模型

多核计算中多边形合并的并行算法在模型设计上应充分利用已有多核处理器的优势，最大限度地利用 CPU 的处理能力。本文提出基于数据划分的多边形合并模型。一个数据划分将具有相近空间位置的多边形尽可能地划分在一起，每个数据划分只提供整个服务的一个数据子集[5,6]，能独立地接收并处理与该数据相关的请求。每个数据划分只能存在于一个物

**收稿日期**：2011-11-25；**修回日期**：2011-12-30 **基金项目**：中央高校基本科研业务费专项资金资助项目（CUGL090251）；国家教育部地理信息系统软件及其应用工程研究中心开放课题（20111109）

**作者简介**：陈占龙（1980-），男，河北无极人，博士，主要研究方向为地理信息系统研究与应用（chenzhanlong2005@126.com）；吴亮（1976-），男，副教授，博士，主要研究方向为空间分析；刘焕焕（1987-），女，硕士研究生，主要研究方向为空间分析、数据挖掘.

理节点上,不同数据划分之间的数据互不重叠,所有的数据划分构成了整个多边形合并的服务数据。图 1 显示了基于数据划分的多边形合并模型。

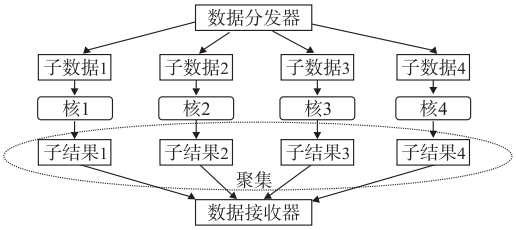


图1 多核计算中的并行算法模型

在图 1 中,由数据分发器按照一定的规则将数据进行划分,得到各个子数据;把各子数据随机地分配给各个核,每个核对各自的数据进行处理,将得到的各子结果返给数据接收器;待各个核把子结果都传递给数据接收器后,数据接收器再对子结果进行处理。

在该模型中需要一个子数据的阈值 $D_{max} = \min(D_1, D_2)$。其中:$D_1$ 为一个核可以负载的最大值,$D_2$ 为每个核应该承载的数据量。在数据分配器中需要一个计数器,初始值设为 $N$($N$ 为子数据的个数),当某个核处于空闲状态时通知数据分配器,如果仍有子数据没有被分配,则数据分配器把一个子数据分配给该核,计数器减少 1;当计数器减少到 0 时,不再对空闲核分配子数据。在数据接收器对子数据结果进行处理之前,也需要一个计数器来对收到的子结果数据进行计数,该计数器初始化为 0。当一个子结果到达数据接收器时,计数器增加 1;当计数器增加到与数据分发器发出的子数据个数一致时,数据接收器对子结果进行处理。该模型对数据进行动态分配,有较好的负载均衡能力。

## 2 基于 Hilbert 空间填充曲线的数据划分算法

### 2.1 Hilbert 曲线分析

Hilbert 曲线是由德国数学家 David Hilbert 发现并构造的[7],编码定义是通过对正方形区域逐次分割得到的标号次序给出的。

将一个正方形等分成四个小正方形,依次按照 dot0→dot1→dot2→dot3 的次序遍历每个四等分正方形格网的中心,这是一次迭代。如果对四个小正方形继续上述过程,往下划分,反复进行,最终就得到一条可以填满整个正方形的曲线,这就是 Hilbert 曲线,其大致过程如图 2 所示。本文主要研究二维 Hilbert 曲线的性质,通过上述说明可以发现,Hilbert 曲线拥有良好的递归和复制性质,下一级 Hilbert 曲线总是通过对上一级部分曲线的复制,运用旋转操作完成整体结构。

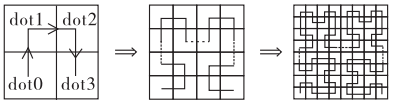


图2 Hilbert曲线生成过程

### 2.2 Hilbert 编码

对于 Hilbert 编码,已有许多方法[8~10],本文通过对曲线走势规律的掌握,计算空间对象前的空间点数目,从而得到正确的编码。

为了尽可能地逼近空间对象的真实位置,减少空间对象 Hilbert 编码的重复,Hilbert 曲线阶数 $N$ 与空间对象个数 num 之间需要满足条件 num≤22$N$,本方法中 $N$ 取满足条件的最小整数值。理想情况下,每个网格只包含一个空间对象标志,这样就建立了空间对象标志与 Hilbert 编码之间的一一对应关系。当然,允许存在同一编码对应多个空间对象的情况。

#### 2.2.1 空间区域栅格化

该方法是基于层次分解的,通过递归计算得到编码值。基本思路是将空间区域栅格化为 $2N \times 2N$ 的网格阵列,使用空间对象的最小外包矩形的中心点作为空间对象的近似,每个网格可以包含多个空间对象。

各点原始坐标以屏幕坐标为准,屏幕左上角为原点,分别向正东和正南方向延伸,如图 3 所示。

根据栅格化结果,分别计算出空间对象的栅格编码的行、列二进制编码,编码数与划分阶数等值,为 $N$。通过不同位上的数据类型采用字符数组的形式,字符数组长度二进制编码组合,得到当前空间对象在对应的划分层次所处的象限。例如对于一阶划分,现有四个空间点,坐标分别为 $a_0$(100,300)、$a_1$(100,100)、$a_2$(300,100)、$a_3$(300,300),其栅格编码的二进制编码的行列组合为(0,0)、(0,1)、(1,1)、(1,0),行列走向以左下角为原点,分别向正东、正北向延伸,如图 4 所示。

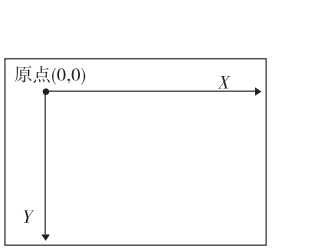


图3 屏幕坐标系

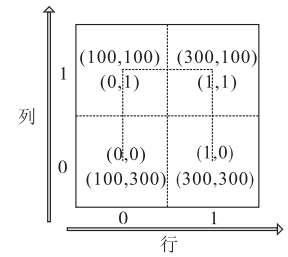


图4 对象栅格化对应的二进制编码

#### 2.2.2 父型和子型

Hilbert 曲线本身最大的特性是它的复制性。Hilbert 曲线的基本图形如图 5 所示。以此为基本图形进行不断地旋转与复制,得到曲线的四个基本图形,开口方向分别为南、西、北、东,走势固定,如图 5 中箭头所示,曲线通过对该组图形的不断复制得到。进一步观察,高阶 Hilbert 曲线是由下列四种走势不随阶数变化的子型不断复制得到的,如图 6、7 所示。

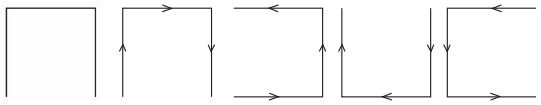


图5 四个组成曲线的基本图形

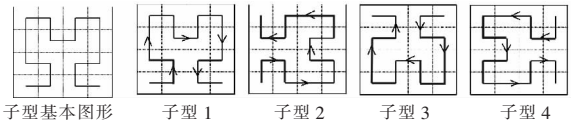


图 6 子型类型

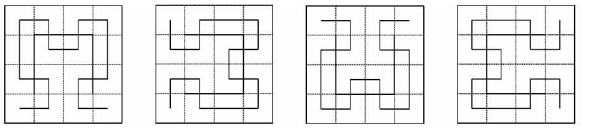


图 7 基本图形-子型的派生

进一步观察发现,在曲线大于等于 3 阶时,是由开口方向分别为南、西、北、东的基本子型 1、2、3、4 复制得到的。由此引出父型与子型的概念。父型是在上一级划分时得到的基本图形(南 1、西 2、北 3、东 4)或者子型(子型 1、2、3、4)。以父型几

何中心点为原点划分出四个子象限,根据空间对象所处不同象限得到的基本图形(或子型),子型可以作为某一父型继续派生出新的子型,父型也可以是根据上级划分得到的父型派生得到的子型。

这里规定象限的划分是将区域四等分后,从左上角起顺时针依次为 1、2、3、4 象限,并且其派生出不同子型,子型与父型是相对而言的,对应的二进制编码分别为(0,1)、(1,1)、(1,0)、(0,0)。

表 1 和 2 为父型与象限组合派生出的子型类型。当父型为基本图形时,子型如表 1 所示;当父型为子类型时,子型如表 2 所示。

表 1　基本图形派生的子型

| 父型 | 子型 |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |

表 2　父型与象限组合派生的子型

| 父型与子象限类型 | 子型 |
|---|---|
| 11,12,24,43 | 1 |
| 14,22,23,31 | 2 |
| 21,33,34,42 | 3 |
| 13,32,41,44 | 4 |

根据派生出的子型类型,可以得到曲线走势,根据这个走势,判断出在当前象限前沿着曲线方向的网格单元数。

#### 2.2.3　编码算法步骤

a)获取空间对象属性信息,取得整个区域空间对象数目 num,计算出划分中止阶数 $N$,栅格化行列数 $M$。当空间对象为点时,直接取其坐标;当空间对象为线、面时,取其最小外包矩形(MBR)的中心点,对区域进行栅格化并得到其栅格编码字符数组 $a[N]$,$b[N]$。

b)当 $N=1$ 时,此时只需进行一次划分,曲线为基本图形 1,计算点所在象限,直接得到编码。

c)当 $N=2$ 时,此时需要进行两次划分,计算两级划分时点分别所在象限,得到编码。

d)当 $N\geqslant 3$ 时,需要进行 $N$ 次划分,设 progression 为当前划分阶数,初值为 0。

(a)若 progression = 0,进行一级划分,计算此时的 key 值,progression 自增;

(b)若 progression = 1,进行二级划分,计算此时的 key 值并与上级 key 值累加,progression 自增;

(c)若 progression ≥ 2,计算由上级划分得到的父型与本级划分得到的象限共同派生出的子型类型,再求得本级象限划分后的子象限值,由此可计算出当前划分阶数的 key 值,与上级 key 累加,progression 自增;

(d)若 progression ≤ $N$,则返回步骤(a);否则编码结束,返回该空间对象的 Hilbert 编码值。

### 2.3　多边形数据划分算法步骤

数据划分的思想是:把数据按照 Hilbert 曲线进行编码,根据核的数量将数据进行 $N$ 个划分,如图 8 所示。

a)获取多边形对象属性信息(包括 ID、坐标值、数据量),进行 Hilbert 编码并按编码值进行升序排序。

b)计算多核中每个核的平均空间数据量。

c)按编码升序,取当前空间对象的数据量大小,累加至当前核中的数据量。

d)当当前核中的数据量小于或等于平均数据量时,将该对象分配给当前核;否则从该核起,开始向下一核存储,转步骤 c)。如果当前处理的是最后一个节点,则将剩余空间数据全部存入,划分结束;否则,按聚类块的下标次序,依次取出当前聚类中的空间对象,获取当前空间对象的数据量大小,累加至当前节点中的数据量;过程结束。

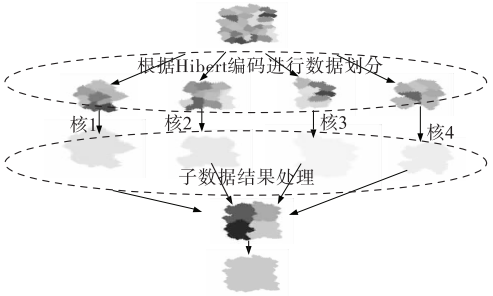


图8　多边形合并示意图

## 3　简单要素类多边形合并算子

简单要素类中没有显式地存储拓扑信息,且该数据类型多边形数据有以下的特点:多边形可以是多圈的(即可以有洞);多边形每圈都以首末点重合的点坐标串描述,外圈顺时针存储,内圈逆时针存储,即沿坐标点的方向行进,多边形的内部始终在行进方向的右边,多边形的外部始终在行进方向的左边;相邻多边形间的共享边被存储两次。据此,对于有严格一致的共享重合边的多边形数据,其算法思路是:将多边形间的公共边找出,记录公共边的起止点(端点);根据连接起止点的弧段情况,建立端点与弧段间的 link 链表;根据 link 链表去掉重合边;最后重组 link 表中的边得到合并结果。因此,公共边的准确、快速判别既是以后成功合并多边形的前提,又是本文研究的重点问题。

本文算法对重合边的判断是基于线段依次判别,这里的线段是指由两坐标点组成的线段。通过比较线段的外包矩形与另一多边形的外包矩形是否相交,初次排除大量的非重合点;如果线段与另一多边形的外包矩形相交,则线段与另一多边形可能有重合点,再进行线段与另一多边形每段线段的外包矩形的相交判断;如果相交,则对两段线段是否有公共点进行判断,找出重合点,此时只需比较四个点是否重合。由于公共边在两个多边形中的坐标点序列肯定是连续的,找到一个重合点后,只需对两多边形同步向两端找到各端的第一个非重合点,即可找到公共边的端点,公共边也即可被求出;下一个公共边的求取,只需以一个多边形为基准从上一个公共边的末端点的后一个点开始计算求取。其算法为:

```
typedef struct tag_RepeateEdge_INFO
{
long a_start;//公共边在多边形 A 上的起始点在 A 坐标数组中下标
long a_end; //公共边在多边形 A 上的终止点在 A 坐标数组中下标
long b_start;//公共边在多边形 B 上的起始点在 B 坐标数组中下标
long b_end; //公共边在多边形 B 上的终止点在 B 坐标数组中下标
long ilen;   //公共的点数
}RepeateEdge_INFO;
```

对于图 9 这种有公共边的两多边形,箭头方向为每个多边形中坐标点的存储顺序方向,图中有一个公共边,此公共边在 RepeateEdge_INFO 结构体中对应的信息为{a_start = 2,a_end = 5,b_start = 1,b_end = 4,long ilen = 4}。

对于有洞的多边形,多边形间的公共边也是在外圈处有重合,只需对外圈进行考虑。首先对多边形 A(其上有 $n$ 个点,包括首尾闭合的相同点)和 B 去掉与首点重复的尾点;然后对多边形 A 进行一次从 0 ~ ($n$ - 1)的遍历,取 A 中每段线段计算其外包矩形,再与 B 的外包矩形进行是否相交比较,如果相

交,对 B 中所有线段依次求外包矩形,并进行相交比较;再对 A 中下一段线段进行类似的操作,那么对 A 最多进行[(n-1)/2]+1 次循环比较。

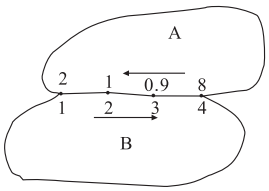


图9 多边形公共边信息

基本过程是对 A 从下标为 0 的点开始找公共边,如果从 0 ~k 都没有重合点,此种情况下标记 mark = n - 1,此时遇到第一个重合点必定是一个公共边的端点(命名为首端点);再向后找,遇到第一个不重合点,那么这个点的前一个点必是公共边的另一个端点(命名为末端点)。找下一个公共边时,从上一个公共边的末端点所对应 A 上的点的后一线段找起,直到第 n-2 个点和第 n-1 个点组成的线段终止。如果第 0 个点与 B 中点有重合的点(如图 9),那么这个点不一定是此线段的端点,要向 A 两端找重合点,找到的公共边的首末端点在 A 中对应的分别是 8 和 2,把末端点的前一个点记录下来(mark = 8 -1);再从末端点后的一个点即 3 开始找另一个公共边,此时不是到(n-1)-1 终止,而是到 mark-1 终止。这两种情况可以统一起来。算法步骤如下:

a)定义的公共边的结构体进行公共边查找。

b)端点演变节点,归并连续线段为弧段,置公共边弧段为删除。

c)弧段重组以合并多边形。

按由点找弧、再由弧找点遍历节点表和弧段表,以重组弧段生成合并结果。

## 4 算法实现及实验分析

本文提到的算法已在 MapGISK9 平台上采用 C++ 编程语言进行了开发实现。采用无拓扑的简单要素类数据,能够很好地解决有公共边的多边形合并,并能正确处理带洞的多边形的合并操作,对大数据量的多边形合并速度也较快。图 10 为某地区的多边形简单要素矢量区数据,共有 8 050 个区,对其进行多边形合并,其结果如图 11 所示,共耗时 6 min 58 s。

图10 合并前

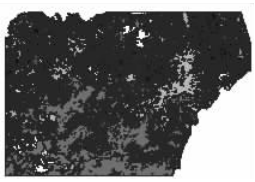

图11 合并后

该测试分析利用了网格环境中的双核 CPU 进行测试。图 12 为合并多边形区分析效率。可以看出,图 12(a)是对多边形的空间实体为 20 000 时,单核点和双核的运算时间都差不多,因为此时的数据量较小,并且点做合并多边形区算法相对比较简单;当规模为 700 000 时,单核做完此合并区分析需要用时 230 s,双核需要用时 135 s;合并对象是线时,如图 13(b)所示,当数据量是 7 000 时,单核和多核的差别仍然较小,但当数据量增大如 50 000 时,单核和双核差距很大,单核用时大约 109 s,双核用时大约 65 s;而对于图 12(c)所示的多边形合并用时,单核和双核在数据量 300 时,相比于点和线合并时用时的差距明显变大,这是由于多边形的合并要复杂得多,对于 3 000的多边形合并用时差距相差一半左右,单核用时 165 s,双核用时 87 s。总之,当数据量相对较小时,单节点和双节点的运算时间相差不大;但当数据达到一定规模时,双核并行计算可以提高将近一倍的运算效率。随着 CPU 核的增加,空间合并区运算的效率也呈线性增长的趋势。

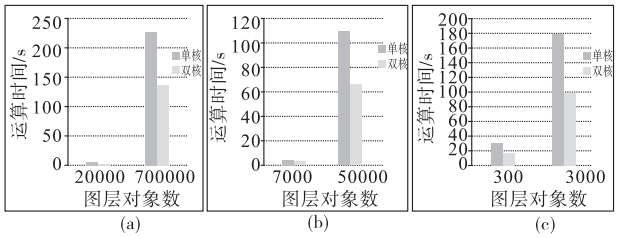


图12 合并多边形区分析效率图

## 5 结束语

本文提出的算法引入 Hilbert 曲线编码,对数据进行划分,充分保证了多边形的聚集性,进一步提高了运算速度。实践结果表明,该算法快速、有效,具有较大的应用价值。

**参考文献:**

[1] 何宇兵. 地学制图综合中多边形对象的合并算法研究与应用[D]. 杭州:浙江大学,2007.

[2] 陈占龙,吴信才,吴亮. 基于单调链和 STR 树的简单要素模型多边形叠置分析算法[J]. 测绘学报,2010,39(2):102-108.

[3] CASTRO J,GEORGIOPULOS M,DEMARA R,*et al*. Data-partitioning using the Hilbert space filling curves:effect on the speed of convergence of fuzzy ARTMAP for large database problem[J]. Neural Networks,2005,18(7):967-984.

[4] MOON B, JAGADISH H V,FALOUTSOS C,*et al*. Analysis of the clustering properties of the Hilbert space-filling curve[J]. IEEE Trans on Konwledge and Data Engineering,2001,13(1):124-141.

[5] 杨小虎,王新宇,毛明. 基于数据划分的分布式模型及其负载均衡算法[J]. 浙江大学学报:工学版,2008,42(4):602-607.

[6] MATTSON T G,SANDERS B A,MASSINGILL B L. 并行编程模式[M]. 敖富江,译. 北京:清华大学出版社,2005.

[7] 齐东旭. 分形及计算机生成[M]. 北京:科学出版社,1997.

[8] 陆锋,周成虎. 一种基于空间层次分解的 Hilbert 码生成算法[J]. 中国图象图形学报,2001,6(5):465-467.

[9] 陈宁涛,王能超,陈莹. Hilbert 曲线的快速生成算法设计与实现[J]. 小型微型计算机系统,2005,26(10):1754-1757.

[10] 曹忠升,张杨,李晨阳. 一种基于分划思想的 Hilbert 曲线快速编码算法[J]. 计算机工程与科学,2006,28(11):63-65.

[11] BUTZ A R. Alternative algorithm for Hilbert space-filling curve[J]. IEEE Trans on Computers,1971,20(4):424-426.

[12] GRIFFITHS J G. An algorithm for displaying a class of space-filling curves[J]. Software-Practice and Experience,1986,16(5):403-411.

[13] BARTHOLDILLL J J,GOLDSMAN P. Vertex-labeling algorithms for the Hilbert space filling curve[J]. Software-Practice and Experience,2001,31(5):395-408.

[14] CHEN Ning-tao,WANG Neng-chao,SHI Bao-chang. A new algorithm for encoding and decoding the Hilbert order[J]. Software-Practice and Experience,2007,37(7):897-908.