文章编号：1007-4619 (2000) 01-0032-04

一种生成 Delaunay 三角网的合成算法

武晓波，王世新，肖春生
(中国科学院 遥感应用研究所 北京 100101)

摘 要： 经过 20 多年的研究，自动生成 Delaunay 三角网的算法已趋于成熟。它们基本上可分为分治算法、逐点插入法、三角网生长法等 3 类。其中前两类较第 3 类在应用上更加广泛。但即使这两类算法也分别存在着时间和空间效率上的缺陷，使它们的应用受到了一定的限制。提出了一个融以上两类算法优点于一体，兼顾空间与时间性能的合成算法。经测试，它的运算效率大大高于逐点插入法，在大多数情况下，也高于分治算法，在分割阈值约为总数据量的十分之一时，效率最高。

关键词： Delaunay 三角网；合成算法；分治算法；逐点插入法
中图分类号： TP79/TP393 **文献标识码**： A

1 引 言

在地学领域中存在着大量基于点的数据，如高程数据、气象观测数据、钻井资料、物化探资料等。充分利用这些空间信息是许多地学研究的基础。1908 年，俄国学者 G·Voronoi 完成了一项奠基性研究，从数学上定义了每个观测点所能代表的空间范围，并用 Voronoi 图(V-图)在二维平面上表示了它的几何意义。1911 年，A·H·Thiessen 应用 V-图理论进行了区域降水量的研究[1]。1934 年，B·Delaunay 根据 V-图推演出了更易于使用的 Delaunay 三角网(D-三角网)。此后，V-图和 D-三角网在地学、图形图像等相关领域得到了极为广泛的应用。

D-三角网是由连接 V-图中具有公共边相邻多边形的中心形成的网络。它具有两个显著的特征，即：

(1) 空外接圆性质。任何一个三角形的外接圆均不包含其它数据点。

(2) 最大的最小内角性质。在所有可能形成的三角网中，D-三角网中三角形的最小内角是最大的。

这两个性质有效地保证了 D-三角网是最接近等角或等边的三角网，同时它也是自动建立 D-三角网的算法依据。

分治算法和逐点插入法由于易于实现，是当前应用较广的两类算法。这两类算法所采用的实现方法决定了它们存在着明显的局限性，分治算法需要大量的内存，逐点插入法运行速度极慢。当数据量较大或计算机性能较差时，它们的使用都将遇到困难。

本文提出了一个成功地解决了上述问题的合成算法。该算法将逐点插入法嵌入到分治算法中，使它们优势互补，弥补了各自的缺陷。经过一个有 2533 个点数据测试，表明合成算法的运算效率大大高于逐点插入法，在大多数情况下也高于分治算法。

2 已有算法介绍

由于建立 D-三角网需要进行大量的计算，所以电子计算机一经出现，人们很自然地就把繁重的计算任务交给了它。到目前为止，学者们提出的各种自动建立的算法基本可归为 3 类：分治算法、逐点插入法和三角网生长法。Tsai 对此曾作了一个非常好的回顾[2]，现将其作一简要介绍。

2.1 分治算法

Shamos 和 Hoey 首次提出了一个用分治算法的思想实现的生成 V-图的算法[3]。它后来被 Lewis 和 Robinson 加以改进并应用于生成 D-三角网[4]。这一算法是不断地将数据分割为两个近似相等的子集，

收稿日期：1998-09-09；修订日期：1998-12-28
基金项目："九五"国家攻关，"95-759 国土资源环境和区域经济信息系统及空间信息基础设施关键技术研究"项目支持。
作者简介：武晓波 (1965—)，男，1986 年毕业于长春地质学院，1989 年在中国科学院遥感应用研究所获地图学与遥感专业硕士学位，主要从事遥感、地理信息系统开发及其应用研究。

直至子集中的点数不大于 4 而生成子三角网，然后逐级合并生成最终的三角网。

分治算法是通过递归地执行同一源代码而实现的。因此，当数据量很大时，会产生许多中间层次而占用大量计算机内存，即要求栈空间非常大。如果计算机没有足够的内存，这一方法就无法使用。

2.2　逐点插入法

逐点插入方法是由 Lawson 1977 年提出的[5]。它首先建立一个大的三角形或多边形，把所有数据包围进来。然后逐点插入到这个超级三角形或多边形中。同时用 Lawson 设计的局部优化过程 LOP 进行优化，以保证生成 D-三角网。这一算法后来被许多人作了完善，如 Lee 和 Schachter，Bowyer，Sloan，Macedonio 和 Pareschi 等[6−9]。

逐点插入法虽然容易实现，空间要求不大，但它最大的不足是时间效率极低。

2.3　三角网生长法

三角网生长法的操作过程是任选一点，找到与它距离最近的点相连成为一条 Delaunay 边。按 Delaunay 条件寻找与此边构成 Delaunay 三角形的第 3 个顶点。重复进行这一过程直到所有数据都被连接进三角网中。因搜索第 3 个顶点的方法不同，这一算法有多种实现，如 Brassel 和 Reif，McCullagh 和 Ross 等[10,11]。

表 1 列出了上述 3 类算法的时间复杂度，即程序中最大的语句执行频度，它是数据量的函数。

表 1　几种 Delaunay 三角网生成算法的时间复杂度[2]

Table 1　Run-time complexities for several Delaunay triangulation algorithms

	算法	一般情况	最坏情况
分治算法	Lewis 和 Robinson (1978)	O(NlogN)	$O(N^2)$
	Lee 和 Schachlter (1980)	O(NlogN)	O(NlogN)
	Dwyer (1987)	O(NloglogN)	O(NlogN)
	Chew (1989)	O(NlogN)	O(NlogN)
逐点插入法	Lawson (1977)	$O(N^{4/3})$	$O(N^2)$
	Lee 和 Schachlter (1980)	$O(N^{3/2})$	$O(N^2)$
	Bowyer (1981)	$O(N^{3/2})$	$O(N^2)$
	Watson (1981)	$O(N^{3/2})$	$O(N^2)$
	Sloan (1987)	$O(N^{5/4})$	$O(N^2)$
三角网生长法	Green 和 Sibson (1978)	$O(N^{3/2})$	$O(N^2)$
	Brassel 和 Reif (1979)	$O(N^{3/2})$	$O(N^2)$
	MaCullagh 和 Ross (1980)	$O(N^{3/2})$	$O(N^2)$
	Mirante 和 Weigarten (1982)	$O(N^{3/2})$	$O(N^2)$

注：表中 N 表示数据量

3　合成算法

由以上介绍不难看出，目前采用较多的前两类算法各具优势又有局限，同时，它们又具有明显的互补性。分治算法时间性能好，空间性能差；逐点插入法空间性能好，时间性能差。我们评价一个算法的优劣是看它对时间和空间的消耗，即时空性能的综合表现。因此，就产生了一个非常自然的想法，为何不把它们结合起来，取长补短，从而提高算法的性能呢？

把分治算法与逐点插入法结合起来的具体做法是，以分治算法为主体，当递归分割数据集的过程进行到子集中的数据量小于一个预定值——分割阈值时终止，然后用逐点插入法在子集中生成子三角网。我们把这一新的算法称为合成算法。它的流程图见图 1。其中 V 表示数据集；N_V 是 V 的数据量；N_d 是分割阈值；N_L，N_R 分别表示两个子集的数据量；T_L，T_R 分别表示在子集中建立的两个子三角网。

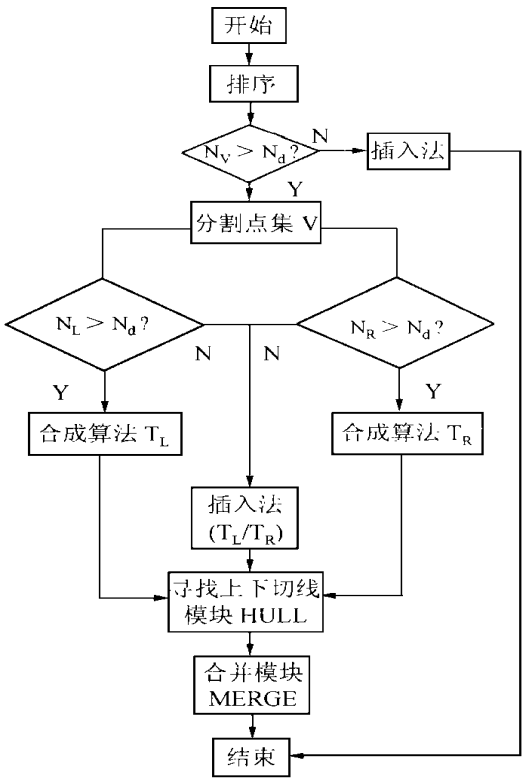

图 1　合成算法主程序流程图

Fig. 1　The flow chart of data processing in hybridized method

图 1 表明，为便于分割，首先按升序以横坐标 x 为主，纵坐标 y 为辅把数据排序。然后比较 N_V 与

N_d，如 $N_V > N_d$ 则分割数据集，否则用逐点插入法生成三角网。合成算法包含 3 个模块：逐点插入法模块、寻找上下切线模块以及合并模块。

3.1 逐点插入法模块

由上述知，逐点插入法有多种实现方法，这里选用 Macedonio 和 Pareschi 提出的方法。在这个方法中，把数据中的凸壳作为超级多边形。所谓凸壳即由数据中的凸集顺序相连形成的多边形。逐点插入法模块由以下 4 个步骤完成。

(1) 找出凸壳。这一步用 Larkin 1991 年提出的算法来做[12]。为提高搜索效率，先将数据集划分为矩形栅格。设每个栅格单元内包含的数据为 4，则栅格的行列数为 $N_V/4$。因具有 $x, y, x+y, x-y$ 的最大值及最小值的点是凸集中的点[13]，把它们按逆时针方向顺序相连作为初始凸壳。然后用递归函数 convex(I,J)完成整个凸壳。I,J 是凸壳上的相邻点。convex(I,J)找出凸壳上 I,J 间的另一点 K。如果不存在，则返回空。K 是与 IJ 相交的栅格中位于 IJ 右侧且与 IJ 距离最大的点，或当 IJ 右侧无点，但 IJ 上有点，且位于 I,J 之间的点。

(2) 建立初始三角网。以凸壳上 x 值最小的点为出发点，按序与其余点相连。

(3) 插入点。先找出包含要插入的点 P 的三角形 t，然后连接 P 与 t 的 3 个顶点。

(4) 优化。用 D-三角网的空心圆性质考查新生成的三角形，如不满足，则调换与相邻三角形所组成的四边形中相连的对角线。向相邻三角形扩展此过程直到满足条件为止。

后两步循环执行直到插入全部数据。

3.2 寻找上下切线模块

这个模块找出连接左右两个子三角网的凸壳 H_L 和 H_R 的上切线和下切线。设 X 是 H_L 中 x 值最大的点；Y 是 H_R 中 x 值最小的点；Z' 是 H_R 上 Y 逆时针方向的邻点；Z'' 是 H_L 上 X 顺时针方向的邻点。寻找 H_L 和 H_R 的下切线过程由两个循不完成。(1) 如 Z' 位于 XY 右侧，则 Z' 成为新的 Y，Y 逆时针方向的邻点成为新的 Z'，否则退出。(2) 如 Z'' 位于 XY 右侧，则 Z'' 成为新的 X，X 逆时针方向的邻点成为新的 Z''，否则退出。用类似的方法可以生成 H_L 和 H_R 的上切线。

3.3 合并模块

子三角网 T_L 和 T_R 的合并由一个循环完成。以 H_L 和 H_R 的下切线为起始基线，分别在 T_L 和 T_R 中找出与它构成 Delaunay 三角形的第 3 个顶点 L_1 和 R_1，在这两个三角形中选择外接圆半径小的连接到三角网中。新生成的边作为新的基线继续这个过程，当基线成为 H_L 和 H_R 的上切线时终止。

在寻找上下切线模块和合并模块中，要频繁用到两个函数 $\mathrm{pred}(V_i, V_j)$ 和 $\mathrm{succ}(V_i, V_j)$。$\mathrm{pred}(V_i, V_j)$是在包含公共端点 V_i 的一簇线段中，得到与边 V_iV_j 顺时针方向邻边的另一个端点。$\mathrm{succ}(V_i, V_j)$相反，是得到与边 V_iV_j 逆时针方向邻边的另一个端点。

4 实例测试与结论

我们用 Microsoft C 实现了合成算法，并用一个有 2533 个点的数据进行了测试。所用平台是 586/90 微机，内存 16 MB。为与分治算法及逐点插入法比较，由小到大取多个分割阈值进行测试。分割阈值越小，算法愈接近于分治算法，这里以 10 代之。因为当取更小的值时，子集中的点有共线现象。

表 2 合成算法在不同分割阈下运行所需时间

Table 2 Hybridized Algorithm's performance under various split threshold

分割阈	运行时间/s
10	35(分治算法)
50	25
100	23
200	22
500	26
1000	28
2000	33
3000	50(逐点插入法)

测试结果如表 2。合成算法的时间效率远优于逐点插入法，在大多数情况下，它也好于分治算法。当分割阈值约为数据量的十分之一时，它的效率最高。

参考文献(References)

[1] A. H. Thiessen. Precipitation averages for large areas[J]. *Monthly*

Weather Review, 1911, **39**: 1082—1084.

[2] V. J. D. Tsai. Delaunay triangulations in TIN creation: an overview and a linear-time algorithm [J]. *Int. J. of GIS*, 1993, **7**(6): 501—524.

[3] M. I. Shamos, D. Hoey. Closest-point problems [C]. Proceedings of the 16th Annual Symposium on the Foundations of Computer Science. 1975, 151—162.

[4] B. A. Lewis, J. S. Robinson. Triangulation of planar regions with applications [J]. *The Computer Journal*, 1978, **21**(4): 324—332.

[5] C. L. Lawson. Software of C' surface interpolation, Mathematical Softeware III. J. Rice [M]. Academic Press, New York, 1977.

[6] D. T. Lee, B. J. Schachter. Two algorithms for constructing a Delaunay triangulation [J]. *Int. J. of Computer and Information Sciences*, 1980, **9**(3).

[7] A. Bowyer. Computing Dirichlet tesselations [J]. *Computer Journal*, 1981, **24**: 162—166.

[8] S. W. Sloan. A fast algorithm for Constructing Delaunay triangulations in the plane [J]. *Andvanced Engineering Software*, 1987, **9**: 34—55.

[9] G. Macedonio, M. T. Pareschi. An Algorithm for the triangulation of arbitrarily distributed points: applications to volume estimate and terrain fitting [J]. *Computers & Geosciences*, 1991, **17**: 859—874.

[10] K. E. Brassel, D. Reif. Procedure to generate Thiessen polygons [J]. *Geophysical Analysis*, 1979, **11**: 289—303.

[11] M. T. McCaullagh, C. G. Ross. Delaunay triangulation of a random data set for irarithmic mapping [J]. *The Cartographic Journal*, 1980, **17**: 93—99.

[12] B. J. Larkin. An ANSI C program to determine in expected linear time the vertices of the convex hull of a set of planar points [J]. *Computers & Geosciences*, 1991, **17**(3): 431—443.

[13] Sedgewick, R. Algorithms [M]. Addison-Wesley, New York, 1988.

致谢

阎守邕研究员全面指导了这项研究，对本文的段落安排及文字运用等方面提出了许多珍贵建议并作了修改。在此谨向他表示诚挚的敬意和感谢。赵健、崔景年、田青、周艺、王涛等同仁在研究过程中给予了大力帮助，在此也向他们表示衷心感谢。

A Hybridized Method for Building Delaunay Triangulation

WU Xiao-bo, WANG Shi-xin, XIAO Chun-sheng

(*Institute of Remote Sensing Applications, Chinese Academy of Sciences, Beijing* 100101, *China*)

Abstract: A wide variety of algorithms have been proposed to construct triangulation. They fall into three broad categories: divide-and-conquer, incremental insertion and triangulation growth. The first two groups of the methods have been extensively applied to many disciplines because of their easiness in implementation. They are, however, constrained either by their computational inefficiency or by their stringent demand on computer memory. In this paper a hybridized method is proposed to take advantage of both algorithms' strengths so that these limitations could be overcome. In a test of 2533 points, the computation efficiency of the hybridized method is much higher than that of incremental insertion method in all cases, and is also higher than that of divide-and-conquer method in most cases. The best efficiency is achieved when the data points are partitioned into one-tenth of the original size.

Key words: hybridized method; delaunay triangulation; divide-and-conquer; incremental insertion