1

How to Draw a Delaunay Diagram with a Given Topology

Kokichi Sugihara
Department of Mathematical Engineering and Information Physics
Graduate School of Engineering, University of Tokyo

Tetsuya Hiroshima
Japanese Ministry of Defense

Abstract

This paper presents a polynomial-time method for judging whether a given planar embedded graph is topologically equivalent to a Delaunay triangulation, and an iterative method for finding a Delaunay triangulation with a given graph structure.

1. Introduction

The recognition of the Delaunay and Voronoi diagrams can be divided into two problems, i.e., the geometric problem and the combinatorial problem.

The geometric problem is to judge whether a given drawing is a Delaunay or a Voronoi diagram. For the Delaunay diagram, this problem is almost trivial because the empty circle property characterizes the Delaunay diagram completely [9]. On the other hand, the problem is not so trivial for the Voronoi diagram, because we have to find the locations of the generating points. However, this problem has already been solved [1, 2, 6].

The combinatorial problem is to judge whether a given embedded graph is isomorphic to some Delaunay or Voronoi diagram. This problem is practically important because it is closely related to the design of a topologically consistent algorithm for constructing the Delaunay/Voronoi diagram in finite precision [10]. Until recently, only some necessary conditions [5] and sufficient conditions [4] were known.

Recently, a necessary and sufficient condition for a three-connected planar graph to be the vertex-edge graph of a convex polyhedron whose vertices are all on a common sphere [7]. Since there is a one-to-one correspondence between such a convex polyhedron and a pair of the Delaunay diagram and the farthest-point Delaunay diagram on the plane [3], the above condition gives a solution to the combinatorial recognition problem. However, the proof is based on hyperbolic geometry, and is not easy for nonspecialists to understand.

In this paper, on the other hand, we give a necessary and sufficient condition in terms of primitive mathematics, i.e., in terms of linear equations and inequalities, which can be checked in polynomial time by the linear programming technique. The proof of the condition is not constructive; we can just guarantee that the Delaunay diagram exists if the condition is fulfilled. However, we can also construct an iterative method for finding a way to draw a Delaunay diagram with the given graph structure.

2. Delaunay Graph

Suppose that G is a simple, connected, and planar graph embedded in the plane. We assume that every vertex in G is incident to two or more edges. Let $V = \{v_1, v_2, \cdots, v_n\}$ be the set of vertices of G, and let E be the set of edges of G. The edges of G divide the plane into connected regions. These regions are called *cells* generated by the embedded graph G. The outermost cell is unbounded while the other cells are bounded. The bounded cells are called *inner cells*. We call the embedded graph G a *triangulation graph* if all the inner cells are bounded by exactly three edges.

Let G be a triangulation graph. Let C be the set of all the cycles of G that bound the inner cells counterclockwise. We represent this triangulation graph by $G = (V, E, C)$. When we write $G = (V, E, C)$, we concentrate our attention on the topological structure of the embedded graph; we do not care about the actual positions at which the vertices are placed.

For a triangulation graph $G = (V, E, C)$, let ψ be a mapping from V to $\mathbf{R}^2$ such that $\psi(v_i) \neq \psi(v_j)$ whenever $i \neq j$. Placing the vertices v_i of G at $\psi(v_i)$ and representing the edges of G by the straight line segments connecting the two terminal vertices, we get

*This work is partly supported by the Grant-in-Aid for Scientific Research of the Ministry of Education, Science and Culture of Japan.

a diagram. If this diagram realizes the triangulation graph G (i.e., both if no edges intersect except at the end points, and if the set of all the cycles that bound the inner cells coincides with C), then this diagram is called a *triangulation* and is denoted by $H = (G, \psi)$ or $H = (V, E, C, \psi)$.

Triangulation $H = (V, E, C, \psi)$ is called a *Delaunay triangulation* if

(D1) the boundary of the outermost cell forms a convex polygon, and

(D2) the circumcircle of each inner cell (i.e., a triangle) does not contain any vertex in its interior.

Triangulation graph $G = (V, E, C)$ is called a *Delaunay graph* if there is a mapping ψ such that $H = (G, \psi)$ is a Delaunay triangulation.

Graph G is said to be 2-connected if deletion of any one vertex from G does not make the resultant graph disconnected. The Delaunay triangulation gives a 2-connected graph unless all the vertices align on a common line. Hence, we hereafter assume that G is 2-connected. We consider the method for judging whether a given 2-connected graph is a (nondegenerate) Delaunay graph.

3. Main Theorem

Let $G = (V, E, C)$ be a triangulation graph, and let the elements of C be $c_1, c_2, \cdots, c_{|C|}$, where $|C|$ represents the number of elements of the set C. Each cycle c_i has three vertices; we assign three variables x_{3i-1}, x_{3i-2} and x_{3i}, one to each vertex of c_i. In the sequel we want to interpret these variables as angles at the corresponding corner of the triangle. So let us call these variables *angle variables*. In total, we have $3|C|$ angle variables, each corresponding to a corner of a triangle.

A vertex of a triangulation graph $G = (V, E, C)$ is called an *outer vertex* if it is on the boundary of the outermost cell, and an *inner vertex* otherwise. Similarly, an edge of G is called an *outer edge* if it is on the boundary of the outermost cell, and an *inner edge* otherwise.

For a triangulation graph $G = (V, E, C)$, we generate the following five types of constraints; these are the conditions that should be satisfied if G is realized as a Delaunay triangulation.

(C1) For each cycle in C, the sum of the associated three angle variables is equal to 180.

(C2) For each inner vertex, the sum of all the associated angle variables is equal to 360.

(C3) For each outer vertex, the sum of all the associated angle variables is equal to or less than 180.

(C4) For each inner edge, the sum of the associated pair of the facing angle variables (i.e., the angle variables corresponding to the vertices that are on the same cycle as, but are not incident to, the inner edge) is equal to or less than 180.

(C5) Each angle variable is positive.

Let us denote by $S(G)$ the system consisting of all the constraints obtained by (C1)$\sim$(C5).

For any Delaunay triangulation $H = (G, \psi)$, the realized values of the angle variables (measured in degrees) satisfy the constraints. Indeed (C1), (C2) and (C5) are satisfied by any triangulation, whereas (C3) follows from (D1), and (C4) follows from (D2).

However, the converse is not true. That is, the values of the angle variables that satisfy all the constraints in $S(G)$ do not necessarily correspond to a Delaunay triangulation. In order to avoid such inconsistency we need still other conditions.

Let x_i, x_j, x_k be three angle variables corresponding to the three vertices $v_\alpha, v_\beta, v_\gamma$, counterclockwise in this order, of a cell. Then, we say that x_j is *cc-facing* (meaning "facing counterclockwise") around v_α and x_k is *c-facing* (meaning "facing clockwise") around v_α.

Suppose that v_i is an inner vertex of G. Let $d(i)$ denote the number of edges incident to v_i, let $\phi_1^{(i)}, \phi_2^{(i)}, \cdots, \phi_{d(i)}^{(i)}$ be the cc-facing angle variables around v_i, and let $\theta_1^{(i)}, \theta_2^{(i)}, \cdots, \theta_{d(i)}^{(i)}$ be the c-facing angle variables around v_i. Furthermore, we define

$$F(v_i) = \frac{\sin\phi_1^{(i)} \sin\phi_2^{(i)} \cdots \sin\phi_{d(i)}^{(i)}}{\sin\theta_1^{(i)} \sin\theta_2^{(i)} \cdots \sin\theta_{d(i)}^{(i)}}. \qquad (1)$$

We consider only positive angle values, and hence we get $0 < F(v_i) < \infty$.

Note that we have defined three different notations for representing the angle variables. The symbol x_i is used to represent an angle variable in general, while the symbols $\phi_j^{(i)}$ and $\theta_j^{(i)}$ are used to refer to the angle variables cc-facing and c-facing around v_i.

We obtain the following lemma and theorem, though we omit their proofs.

Lemma 1. Any triangulation $H = (V, E, C, \psi)$ satisfies the next condition.

(C6) $F(v_i) = 1$ for any inner vertex v_i.

Theorem 1. 2-connected triangulation graph $G = (V, E, C)$ is a Delaunay graph if and only if the set of conditions (C1)$\sim$(C6) is satisfiable.

Thus, we get the necessary and sufficient condition for a triangulation graph to be a Delaunay graph. However, the condition stated in Theorem 1 is not useful for the recognition of a Delaunay graph, because we do

not know a finite-step algorithm for judging the satisfiability of the condition.

If we happen to be able to delete the condition (C6), we can judge the satisfiability in finite steps because the conditions (C1) through (C5) are linear in the variables and the method for checking their satisfiability has been established in linear programming. Indeed we can delete the condition (C6). The next theorem is the main result of this paper.

Theorem 2. 2-connected triangulation graph $G = (V, E, C)$ is a Delaunay graph if and only if the set of conditions (C1)$\sim$(C5) is satisfiable.

As we have already seen, the angle values satisfying (C1)$\sim$(C5) cannot be used directly for the construction of a Delaunay triangulation. However, we can show that once we obtain the angle values that satisfy (C1)$\sim$(C5), then we can change them so as to satisfy (C6) while (C1)$\sim$(C5) are kept unviolated. This is the main story of the proof of Theorem 2.

Note that the feasibility of a system consisting of linear equations and inequalities can be checked in polynomial time [8]. Hence, we can judge whether given embedded graph $G = (V, E, C)$ is realizable as a Delaunay diagram in polynomial time.

4. Iterative Method for Drawing

Let $G = (V, E, C)$ be a triangulation graph. Suppose that we have an assignment Q of angle values that satisfy $S(G)$. Suppose that v_i is an inner vertex of G. We denote by M_i^α the modification of the angle values in such a way that all the cc-facing angles around v_i are increased by α and all the c-facing angles around v_i are decreased by α. Suppose that, applying M_i^α to the assignment Q, we get the resulting assignment Q'. Then, we write $Q' = M_i^\alpha Q$.

Let m be the number of the inner vertices of triangulation graph $G = (V, E, C)$. We number the inner vertices as $v_1, v_2, \cdots, v_m$ and the outer vertices as $v_{m+1}, v_{m+2}, \cdots, v_{|V|}$. Moreover we assume that any inner vertex v_i is adjacent to a vertex v_j with the larger index j ($> i$). Note that the numbering with this property is always possible.

We denote by $F(v_i)|_Q$ the value of equation (1) for the assignment Q of the angle values. We consider the next procedure.

procedure *angle-adjust*(Q, i):
if $i = 1$ **then** find α by binary search such that
$F(v_1)|_{M_i^\alpha Q} = 1$
and **return** $M_1^\alpha Q$
else find α by binary search such that
$F(v_i)|_{angle\text{-}adjust(M_i^\alpha Q, i-1)} = 1$
and **return** *angle-adjust*$(M_i^\alpha Q, i-1)$

Let Q_0 be an initial assignment of angle values that satisfies $S(G)$. Then, by

$$Q \leftarrow angle\text{-}adjust(Q_0, m)$$

we obtain the assignment Q for which we can draw a Delaunay diagram consistently. However, the procedure *angle-adjust* contains binary search and consequently requires infinite steps. Hence, we obtain only an approximation of the angle values that realize the Delaunay diagram.

References

[1] Ash, P. F., and Volker, E. D.: Recognizing Dirichlet tessellations, *Geometriae Dedicata*, **19** (1985), 175–206.

[2] Aurenhammer, F.: Recognizing polytopical cell complexes and constructing projection polyhedra, *Journal of Symbolic Computation*, **3** (1987), 249–255.

[3] Brown, K. Q.: Voronoi diagrams from convex hulls, *Information Processing Letters*, **9** (1979), 223–228.

[4] Dillencourt, M. B.: Toughness and Delaunay triangulations, *Discrete and Computational Geometry*, **5** (1990), 575–601.

[5] Dillencourt, M. B.: Realizability of Delaunay triangulations, *Information Processing Letters*, **33** (1990), 283–287.

[6] Hartvigsen, D.: Recognizing Voronoi diagrams with linear programming, *ORSA Journal on Computing*, **4** (1992), 369–374.

[7] Hodgson, C. D., Rivin, I., and Smith, W. D.: A characterization of convex hyperbolic polyhedra and of convex polyhedra inscribed in the sphere. *Bulletin of the American Mathematical Society*, **27** (1992), 246–251.

[8] Khachian, L. G.: Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, **20** (1980), 53–72.

[9] Okabe, A., Boots, B., and Sugihara, K.: *Spatial Tessellations—Concepts and Applications of Voronoi Diagrams*, John Wiley and Sons, London, 1992.

[10] Sugihara, K., and Iri, M., A robust topology-oriented incremental algorithm for Voronoi diagrams. *International Journal of Computational Geometry and Applications*, **4** (1994), 179–228.