Construction of the Voronoi Diagram for over 10^5 Generators in Single-Precision Arithmetic

Kokichi Sugihara and Masao Iri

Department of Mathematical Engineering and Information Physics, Faculty of Engineering, University of Tokyo, 7–3–1 Hongo, Bunkyo-ku, Tokyo 113, Japan

Abstract

A number of "efficient" algorithms for constructing Voronoi diagrams have been proposed. However, they are usually designed on the assumption that no numerical error takes place in the course of computation. In real computation, on the other hand, numerical errors cannot be avoided completely, so that it is difficult to judge always correctly geometric relations among points, lines, etc. Misjudgement on geometric relations often results in topological inconsistency, and causes a "theoretically correct algorithm" to fail in practice. Thus, there is an unsurmountable gap between "theoretically correct" algorithms and "practically valid" computer programs.

To fill this gap this paper presents an algorithm that can construct Voronoi diagrams stably in finite-precision arithmetic. The algorithm is based on an incremental-type method, but differs from conventional ones in that the highest priority is placed on the preservation of topological properties that the Voronoi diagrams should possess. From a topological point of view (P1) *Voronoi diagram is a planar graph which partitions the plane into as many cells as the generators,* and (P2) *two cells share at most one common edge.* Each time a generator is added, the algorithm modifies the diagram in such a way that the properties (P1) and (P2) may not be violated. Numerical judgements are made only to select the most probable structure of the diagram as far as (P1) and (P2) are preserved. Hence, however poor the precision in computation may be, the algorithm carries out its task and gives an output that is topologically consistent in the above sense, and the output converges to the true Voronoi diagram as the precision becomes higher.

This algorithm was implemented in a FORTRAN program, and many computational experiments have been done. For any set of generators given in the experiments, the program carried out its task and gave an output that was consistent in the sense that it satisfied the properties (P1) and (P2). The program did not fail even if hundreds of points were given on the periphery of a circle as the generators (which is a typical degenerate case for which a conventional algorithm usually does not work), or even if all the results of floating-point computation in the program were replaced by random numbers. Moreover, we carefully selected the FORTRAN codes for floating-point computation in order to make the program as stable as possible also in the quantitative sense. Actually, although all the floating-point numbers were represented in single precision, the program could construct a diagram that is correct quantitatively as well as qualitatively for 10^5 generators distributed uniformly over a unit square.