Implicit convex polygons

Francisco Gómez[1] Ferran Hurtado[2] Suneeta Ramaswami[3]
Vera Sacristán[2] Godfried Toussaint[4]

Given a polygon P in the plane, there are many elementary problems, such as deciding whether or not a given point p belongs to P, computing the supporting lines of P from an external point p, or finding the minimum circle that encloses P, that, in spite of their apparent simplicity, are fundamental pieces of many different applications, from clicking with a mouse on a computer icone, to visualizing objects in computer graphics or detecting collisions in robotics. As a consequence, these problems have been largely studied, and the obtained results are well known (see [7] or any other textbook). They are not all of the same complexity, but most of them can be solved very efficiently, specially when the involved polygons are known to be convex.

Nevertheless, most of the solution algorithms strongly rely on the order of the list of vertices of the polygons, and not much has been said for other situations, such as the case in which the polygons under consideration are given as the convex hull of a set of not ordered and possibly redundant points (this case is considered in a companion work). In this paper we study the case, that appears to be very natural, in which the polygons are given by a set of linear restrictions, i.e. by a possibly redundant intersection of halfplanes. There is not need of much justification of the claim that these problems are very natural, for they modelize a great number of real problems in which a set of linear restrictions appear as constraining the freedom of the solution.

Given a set $H = \{H_1, \ldots, H_n\}$ of halfplanes, we consider the polygon $P(H) = \cap_{i=1}^{n} H_i$. We want to solve decision, computing and optimization problems, related to one or several such polygons in the plane. Obviously, we could always "construct" the polygons, that is, obtain the ordered lists

[1]Dep. de Matemática Aplicada, E.U. Informática, Universidad Politécnica de Madrid, Ctra. de Valencia Km 7, 28031 Madrid, Spain. E-mail: fmartin@sobrino.eui.upm.es.

[2]Dep. Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Pau Gargallo 5, 08028 Barcelona, Spain. E-mail: hurtado@ma2.upc.es, vera@ma2.upc.es. Partially supported by Projecto DGES-SEUID PB96-0005-C02-02.

[3]Dep. Computer Science, 322 Business and Science Building, Rutgers University, Camden, NJ 08102, USA. E-mail: rsuneeta@crab.rutgers.edu

[4]School of Computer Science, McGill University, 3480 University, Montréal, Québec, Canada H3A 2A7. E-mail: godfried@cs.mcgill.ca.

of their vertices from the sets of halfplanes that define the polygons, and then apply the well known algorithms to solve the problems for the "ordered" polygons. But the complexity of such a procedure would always be $\Omega(n \log n)$, for the reconstruction of the polygons cannot be done more efficiently. The question is then to find out which problems can be solved in optimal $\Theta(n)$ time, and which have complexity $\Omega(n \log n)$, hence subsuming that of the polygon construction.

Examples of problems that can be solved in optimal $\Theta(n)$ time, involving one single polygon, are:
– deciding whether or not a point lies in a polygon;
– deciding wheter or not a line intersects a polygon and, if it does not, detecting on which side of the line lies the polygon;
– computing the two supporting lines of a polygon from an external point;
– finding the minimum distance to a polygon from an external point;
– finding the maximum vertical chord of a polygon;
– finding the largest inscribed circle in a polygon;
and involving two polygons:
– deciding whether or not two polygons intersect;
– finding a separating line of two disjoint polygons;
– computing the two common external tangents of two disjoint polygons;
– computing the two separating tangents of two disjoint polygons;
– finding the minimum distance between two disjoint polygons.
Examples of problems that have complexity $\Theta(n \log n)$ include:
– finding a diametral pair of points in a polygon;
– find the minimum enclosing circle of a polygon;
– find a pair of points at maximum distance between two polygons.

Most of the algorithms that solve these problems in a positive way, are based on a prune-and-search strategy. In fact, some of the problems enumerated above were solved by Megiddo [5, 6] and Dyer [2, 3] as a by-product of their solution to the linear programming problem.

The solutions we propose apply an oracle that allows to discard a constant fraction of the halfplanes H_i in linear time. Recursively applied, the algorithm ends by finding the solution in linear time. In most of the cases, it is possible to obtain an oracle that relies on the solution of a linear programming problem in a convenient direction, in order to ensure that a significant number of input elements will be eliminated. This direction is usually found by computing a median value that can be obtained in linear time by any of the known algorithms [1, 4]. As for the linear programming problem, both algorithms from [5, 8] can be used. Notice that Megiddo's algorithm for linear programming does not require much more than median computing. So, it seems interesting to obtain for our problems a modified solution, in which the oracle uniquely relies on median computing.

As for the problems that have complexity $\Omega(n \log n)$, we offer some examples, and also some hints on how to detect if a problem falls in this class.

Finally, it is important to notice that the technique developped in this work can be extended to three-dimensional problems, optimally solving analogous problems for polyhedra defined by intersection of halfspaces.

References

[1] M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *J. Comput. and Syst. Sci.*, 7:448–461, 1973.

[2] M. E. Dyer. Linear time algorithms for two- and three-variable linear programs. *SIAM J. Comput.*, 13:31–45, 1984.

[3] M. E. Dyer. On a multidimensional search technique and its application to the Euclidean one-centre problem. *SIAM J. Comput.*, 15:725–738, 1986.

[4] C. A. R. Hoare. Algorithm 63 (partition) and algorithm 65 (find). *Communications of the ACM*, 4(7):321–322, 1961.

[5] N. Megiddo. Linear-time algorithms for linear programming in R^3 and related problems. *SIAM J. Comput.*, 12:759–776, 1983.

[6] N. Megiddo. Linear programming in linear time when the dimension is fixed. *J. ACM*, 31:114–127, 1984.

[7] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[8] R. Seidel. Linear programming and convex hulls made easy. In *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, pages 211–215, 1990.