Advanced Algorithms **Fall 2015**

Lecture 3: Geometric Algorithms(Convex sets, Divide & Conquer Algo.)

Faculty: K.R. Chowdhary *: Professor of CS*

3.1 Computational Geometry

Computational geometry has evolved from the classical discipline of the design and analysis of algorithms. It is concerned with computational complexity of geometric problems that arise in disciplines like, pattern recognition, computer graphics, computer vision, robotics, and very-large scale integrated (VLSI) layout, operation research, statistics, etc. In contrast to the classical approach to proving mathematical theorems about geometry-related problems, this discipline emphasizes the computational aspect of these problem and attempts to exploit the underlying geometric properties, e.g., the metric space, to derive efficient algorithmic solutions. The classes of problems that we address in this chapter include proximity, intersection, searching, point location, convex set, convex hull, Voronoi diagrams, CAD/CAM, etc.

3.2 Divide and Conquer

This is a basic problem solving technique and has proven to be very useful for geometric problems as well. This technique mainly involves partitioning of the given problem into several subproblems, recursively solving each subproblem, and then combining the solutions to each of the subproblems to obtain the final solution to the optimal problem.

We demonstrate its working by considering the problem of computing the common intersection of n half planes in the plane. Given is the set S of n half-planes, h_i, represented by,

$$a_i x + b_i y \leq c_i, i = 1, 2, \ldots, n. \tag{3.1}$$

Example 3.1 *Let for $i = 1$, h_1 is: $5x + 4y \geq 20$. This gives all points in half plane as shown in figure 3.1.*

It is simple to understand that, common intersection of half-planes, denoted by,

$$C(S) = \bigcap_{i=1}^{n} h_i \tag{3.2}$$

is a convex set, which may or may not be bounded. If it is bounded, it is convex polygon. In figure 3.2 the shaded area is common intersection.

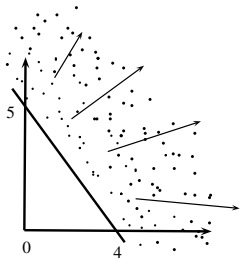

Figure 3.1: Half plane $5x + 4y \geq 20$.

The divide and conquer algorithm computes the following steps:

Algorithm 1 : Algorithm common-intersection-D&C(S);

1: **if** $|S| \leq 3$ **then**
2: compute the intersection $CI(S)$ explicitly
3: Return (C(S))
4: **end if**
5: Divide S into two approx. equal subsets S_1 and S_2
6: $CI(S_1)$ = Common-Intersection-D&C(S_1)
7: $CI(S_2)$ = Common-Intersection-D&C(S_2)
8: CI(S) = Merge (CI(S_1), CI(S_2))
9: Return (CI(S))

The key step is the merge of two common intersections. Because CI(S_1) and CI(S_2) are convex, the merge step basically calls for the computation of the intersection of two convex polygons, which can be solved in time proportional to the size of the polygons. The running time of D&C algorithm is $O(n \log n)$ due to following recurrence formula, where $n = |S|$.

$$T(3) = O(1)$$
$$T(n) = 2T\left(\frac{n}{2}\right) + O(n) + M\left(\frac{n}{2}, \frac{n}{2}\right),$$

where, $M\left(\frac{n}{2}, \frac{n}{2}\right) = O(n)$ denotes the merge time (step 5).

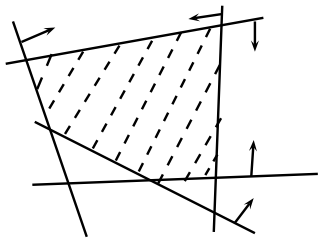

Figure 3.2: Divide-and-conquer scheme for common intersecting half-planes(half planes are on the side of arrows).

3.3 Convex Set

Definition 3.2 Convex Set. *In Euclidean space, a convex set is the region such that, for every pair of points within the region, every point on the straight line segment that joins the pair of points is also within the region.*

Let S be a vector space over the real numbers $\Re$, or, more generally, some ordered field. This includes Euclidean spaces. A set C in S is said to be *convex* if, for all p and q in C and all α in the interval $[0, 1]$, the point $(1-\alpha)p+\alpha q$ also belongs to C. In other words, every point on the line segment connecting p and q is in C. This implies that a convex set in a real or complex topological vector space is path-connected, thus connected. Furthermore, C is *strictly convex* if every point on the line segment connecting p and q other than the endpoints is inside the interior of C.

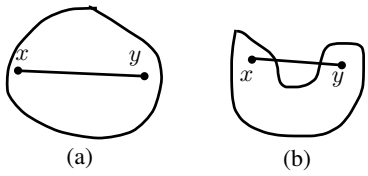

Figure 3.3: (a) Convex set, (b) Non-convex set.

In geometric terms, a body C in the Euclidean space is convex if and only if the line segment joining any two points in C lies totally in C. But, this theorem is not suitable for computational purpose as there are

infinitely many possible pairs of points to be considered. However, other properties of convexity C be utilized to yield an algorithm.

For example, a cube in $\Re^3$ is a convex but its boundary is not, for the boundary does not contain segment pq, unless p and q lie in the same two-dimensional face of the cube.

The importance of convexity theory stems from the fact that convex sets frequently arise in many areas of mathematics, and are helpful in elementary reasoning. Even infinite-dimensional theory is based on 2- and 3-dimensional reasoning.

Any two distinct points p and q of real vector space $\Re$ determine a unique *line*. It consists of all points of the form $(1-\alpha)*p+\alpha*q$, α ranging over all real numbers. Those points for which $\alpha \geq 0$ and those for which $0 \leq \alpha \leq 1$ form respectively the *ray* from p through q and the *segment* pq.

The convex subsets of $\Re$ (the set of real numbers) are simply the intervals of $\Re$. Some examples of convex subsets of the Euclidean plane are solid regular polygons, solid triangles, and intersections of solid triangles. Some examples of convex subsets of a Euclidean 3-dimensional space are the Archimedean solids and the Platonic solids. The Kepler-Poinsot polyhedra are examples of *non-convex* sets.

Example 3.3 *The figure 3.4(a) shows a line segment, while 3.4(b) shows a circle in 2-dimensional convex set.*

Solution. the case of line segment, let the points marked on line are 2, 3, 4, 5. Let us call the points on line as the set C. Hence, $(1-\alpha)*p+\alpha q \in C$. Let $\alpha = 0.5$, $p = 2$, and $q = 5$. We note that $(1-0.5)*2+0.5*5 = 3.5 \in C$.

Similarly, we can verify it for circle. Let the circle is $x^2+y^2=2$, i.e., centered at (0, 0), with radius of 1. Let $x = 0.5, y = 0.75$ is p and $x = -0.4, y = -0.8$ are the points p and q. Let $\alpha = 0.6$. Then first, $(1-\alpha)*p+\alpha q = (1-0.6)0.5+(-0.4)*0.6 = -0.04$. Similarly, for y, $(1-\alpha)*p+\alpha q = (1-0.6)0.75+(-0.8)*0.6 = -0.18$. Thus, we note that $(1-\alpha)*p+\alpha q = (-0.04, -0.18) \in C$, which satisfy the criteria of convexity. Like, this it should satisfy for all (x, y) in the set C.

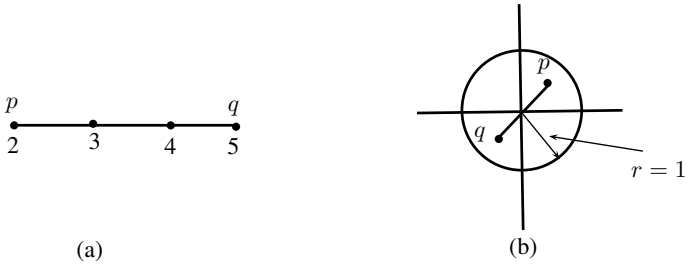

Figure 3.4: (a) Points ona line segment as a Convex set , (b) Points in a circle as a convex set.

Definition 3.4 *A function is convex if and only if its epigraph, the region (in hashed) above its graph (see fig. 3.5), is a convex set .*

Properties of Convex sets: If S is a convex set in n-dimensional space, then for any collection of r, $r > 1$, n-dimensional vectors $u_1, \ldots, u_r$ in S, and for any nonnegative numbers $\lambda_1, \ldots, \lambda_r$ such that $\lambda_1 + \cdots + \lambda_r = 1$, then we have:

$$\sum_{k=1}^{r} \lambda_k u_k \in S. \tag{3.3}$$

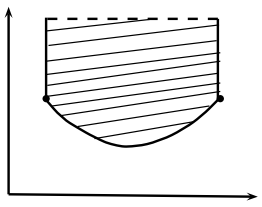

Figure 3.5: Convex function.

A vector of this type is known as a convex combination of $u_1, \ldots, u_r$.

3.3.0.1 Intersections and unions

The collection of convex subsets of a vector space has the following properties:

1. The empty set and the whole vector-space are convex.
2. The intersection of any collection of convex sets is convex(figure 3.6).
3. The union of a non-decreasing sequence of convex subsets is a convex set. For the preceding property of unions of non-decreasing sequences of convex sets, the restriction to nested sets is important: The union of two convex sets need not be convex.

   For example, the non-decreasing convex sets are $C_1 = \{a, b\}$, and $C_2 = \{a, b, c\}$. Then, their union, $C_1 \cup C_2 = \{a, b, c\}$, is obviously, a convex set.

Definition 3.5 Jordan curve theorem. *In topology, a Jordan curve is a non-self-intersecting continuous loop in the plane. The Jordan curve theorem asserts that every Jordan curve divides the plane into an "interior" region bounded by the curve and an "exterior" region containing all of the nearby and far away exterior points, so that any continuous path connecting a point of one region to a point of the other region intersects with that loop somewhere.*

Hence, any two points p, q outside a Jordan curve will intersect even number of times, and two pointer p, r, one is outside and other inside will intersect the curve odd number of times (see figure 3.7(a), and 3.7(b)).

Consider the following problem. Given a simple closed Jordan polygon curve, determine if the interior region enclosed by the curve is convex. This problem can be readily solved solved by observing that if line segment defined by all pairs of vertices of the polygon curve, $\overline{v_i, v_j}, i \neq j, 1 \leq i, j \leq n$, where n denotes the total number of vertices, lie totally inside the region, then the region is convex.

This would yield a straight algorithm with time complexity $O(n^3)$, as there are $O(n^2)$ line segments, and to test if each line segment lies totally in the region takes $O(n)$ time by comparing it against every polynomial segment.

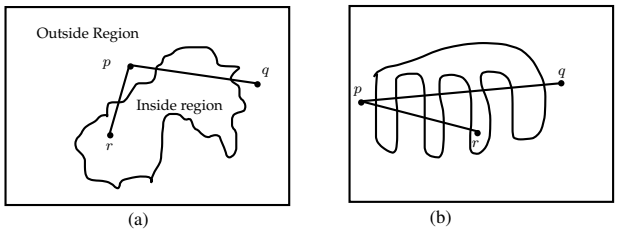

Figure 3.7: Jordan Curve.

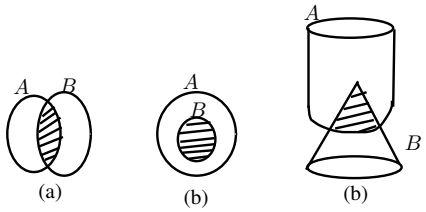

Figure 3.6: Intersection of Convex sets A and B, (a) 2-dimensional, (b) 2-dimensional, (c) 3-dimensional, are all convex.

It may be noted that the interior angle of each vertex must be strictly less than π, in order for the region to be convex. If it is more than π it is *reflex*. It may be noted that all the vertices must be convex, is the necessar condition for the region to be convex.

3.3.1 Is point inside the polygon

The general problem we'd like to solve is, given a point (x, y) and a polygon P (represented by its sequence of vertices), is (x, y) in P, on the boundary, or outside?

Fortunately the problem has a simple and elegant answer. Just draw a ray (portion of a line extending infinitely in one direction) down (or in any other direction) from (x, y), count crossings on ray.

Every time the polygon crosses this ray, it separates a part of the ray that's inside the polygon from a part that's outside. For this, all we really need to know is whether there's an even or odd number of them. In even crossings, the point is outside the polygon.

Algorithm 2 : function is-point-inside-polygon(polygon, point) return inside/outside;

```
1: for each line segment of the polygon do
2:   if ray down from (x,y) crosses segment then
3:     crossings++;
4:   end if
5:   if crossings is odd then
6:     return (inside);
7:   else
8:     return (outside);
9:   end if
10: end for
```

References

[1] Mikhail J. Atallah & Marina Blanta, "Algorithms and Theory of Computation Handbook, Special Topics and Techniques, II ed.," *CRC Press*, 2010.

[2] Allen B. Tucker, Jr., "The computer Science and Engineering Handbook," *CRC Press*, 1997.