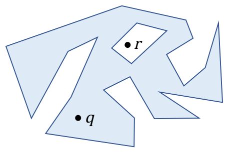
Example sheet week 8

Space.

Algorithms—DJW—2016/2017

Questions labelled FS are from Dr Stajano's list of exercises. Questions labelled $*$ involve more coding or more thinking and are best tackled after the other questions, if you have time.

Question 1. Consider a horizontal line starting from q and going infinitely to the right. How many times does it cross an edge? What about from r? Devise an algorithm to detect whether a given point is inside a given shape (like q) or outside (like r), where the shape is specified as a list of non-intersecting polygons.



Question 2. Two line segments are moving. The first line segment has endpoints $(-0.2+0.1t, -0.1+0.1t)$ and $(0.8+0.1t, -0.4+0.2t)$, the second has endpoints $(-0.36+0.52t, 1.1-0.3t)$ and $(2.1-0.3t, 0.1-0.3t)$, and $t \geq 0$. Do they collide? If so, at what time t? How would you write code to solve this problem for arbitrary coefficients?

Question 3 (FS64). Imagine a complex CAD model consisting of millions of points. Which of the two algorithms we've studied, Jarvis's march and Graham's scan, would you use to compute the convex hull if you expected it to have about 1,000 vertices? What is the rough threshold at which you'd switch to using the other algorithm? (And is this an answerable question?)

Question 4. To compute the convex hull of a set of points P, we started by picking a point $q_0 \in P$ with the lowest y-coordinate, breaking ties by choosing the larger x-coordinate. Give an example to show that, without this tie breaking rule, q_0 might not be a corner point.

Question 5*. In the setup of Question 4, prove rigorously that q_0 is a corner point. (A corner point is a point $p \in P$ such that p is not in the convex hull of $P \setminus \{p\}$.)

Question 6. Graham's algorithm scans points in order, according to a certain angle. Explain carefully what happens when two points have exactly the same angle. Does the algorithm's output depend on which of these two points it scans first?

Question 7*. Generate n points uniformly in the unit square $[0,1] \times [0,1]$, and compute the number of corner points in the convex hull. How does your answer grow with n? *You should easily be able to work with a million points on a low-end laptop.*

Question 8*. The A^* routing algorithm is for finding routes from one vertex to another, on a directed graph with edge weights that is embedded in 2D space. It assumes we know a heuristic distance function $h()$, and it uses this to prioritize which paths to explore. For example, for a Roomba robot vacuum cleaner trying to plan a route to the next room, $h(u \text{ to } v)$ might be the straight line distance from u to v, whereas the true distance $d(u \text{ to } v)$ is the length of the shortest path taking obstacles into account. Read the description at `http://www.redblobgames.com/pathfinding/a-star/introduction.html`.

Given a graph, and a start vertex s, and an end vertex t, A^* proceeds exactly like Dijkstra's algorithm, except that it uses a different key to sort the priority queue:

$$\texttt{key}(v) = v.\texttt{distance} + h(v \text{ to } t)$$

where $v.\texttt{distance}$ is its best guess so far for the distance from s to v, and $h(v \text{ to } t)$ is the heuristic distance function. Assuming that the heuristic distance is always less than or equal to the true distance, i.e. that

$$h(v \text{ to } t) \leq d(v \text{ to } t) \qquad \text{for all } v,$$

prove that A^* is correct, i.e. that when it terminates $t.\texttt{distance} = d(s \text{ to } t)$.