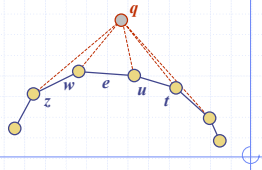


Incremental Convex Hull

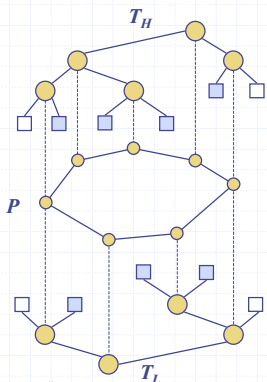


Outline and Reading

- ◆ Point location
 - Problem
 - Data structure
- ◆ Incremental convex hull
 - Problem
 - Data structure
 - Insertion algorithm
 - Analysis

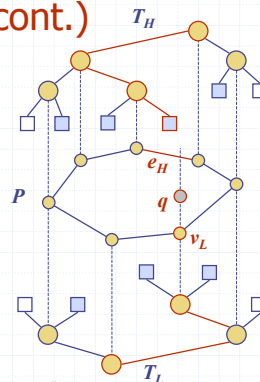
Point Location

- ◆ Given a convex polygon P , a point location query $locate(q)$ determines whether a query point q is inside (IN), outside (OUT), or on the boundary (ON) of P
- ◆ An efficient data structure for point location stores the top and bottom chains of P in two binary search trees, T_L and T_H of logarithmic height
 - An internal node stores a pair $(x(v), v)$ where v is a vertex and $x(v)$ is its x -coordinate
 - An external node represents an edge or an empty half-plane



Point Location (cont.)

- ◆ To perform $locate(q)$, we search for $x(q)$ in T_L and T_H to find
 - Edge e_L or vertex v_L on the lower chain of P whose horizontal span includes $x(q)$
 - Edge e_H or vertex v_H on the upper chain of P whose horizontal span includes $x(q)$
- ◆ We consider four cases
 - If no such edges/vertices exist, we return OUT
 - Else if q is on e_L (v_L) or on e_H (v_H), we return ON
 - Else if q is above e_L (v_L) and below e_H (v_H), we return IN
 - Else, we return OUT

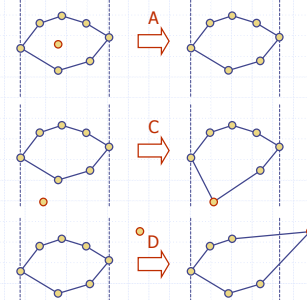


Incremental Convex Hull

- ◆ The incremental convex hull problem consists of performing a series of the following operations on a set S of points
 - $locate(q)$: determines if query point q is inside, outside or on the convex hull of S
 - $insert(q)$: inserts a new point q into S
 - $hull()$: returns the convex hull of S
- ◆ Incremental convex hull data structure
 - We store the points of the convex hull and discard the other points
 - We store the hull points in two red-black trees
 - T_L for the lower hull
 - T_H for the upper hull

Insertion of a Point

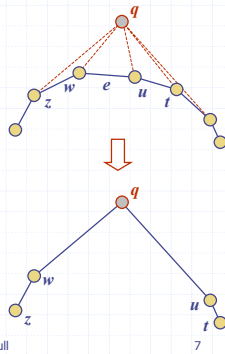
- ◆ In operation $insert(q)$, we consider four cases that depend on the location of point q
 - A IN or ON: no change
 - B OUT and above: add q to the upper hull
 - C OUT and below: add q to the lower hull
 - D OUT and left or right: add q to the lower and upper hull



Insertion of a Point (cont.)

- Algorithm to add a vertex q to the upper hull chain in Case B (boundary conditions omitted for simplicity)

- We find the edge e (vertex v) whose horizontal span includes q
- $w \leftarrow$ left endpoint (neighbor) of e (v)
- $z \leftarrow$ left neighbor of w
- While $\text{orientation}(q, w, z) = \text{CW}$ or COLL
 - We remove vertex w
 - $w \leftarrow z$
 - $z \leftarrow$ left neighbor of w
- $u \leftarrow$ right endpoint (neighbor) of e (v)
- $t \leftarrow$ right neighbor of u
- While $\text{orientation}(t, u, q) = \text{CW}$ or COLL
 - We remove vertex u
 - $u \leftarrow t$
 - $t \leftarrow$ right neighbor of u
- We add vertex q



6/8/2002 2:01 PM

Incremental Convex Hull

7

Analysis

- Let n be the current size of the convex hull
 - Operation locate takes $O(\log n)$ time
 - Operation insert takes $O((1+k)\log n)$ time, where k is the number of vertices removed
 - Operation hull takes $O(n)$ time
 - The amortized running time of operation insert is $O(\log n)$

6/8/2002 2:01 PM

Incremental Convex Hull

8