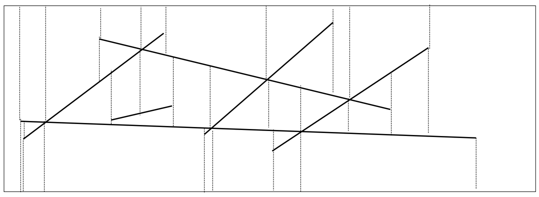
Randomized Methods in Computational Geometry

(Marco Pellegrini)

Lecture 3: Trapezoidal Decompositions

Trapezoidal decomposition

- We extend our geometric interpretation of quicksort to other geometric problems (in dimension 2 and higher)
- Given a set S of n segments in the real plane R^2, build the Trapezoidal decomposition H(S) of R^2 induced by S.

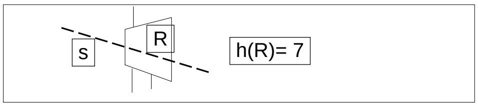


Trapezoidal decompositions.

- H(S) is obtained by drawing a vertical segment upward and downward from (1) each end-point of a segment in S, (2) each intersection point of two segments in S.
- Trapezoidal decomposition are used, for example in planar point location.
- The size of H(S) is O(n+k), where k is the number of intersections among pairs of segments in S.
- We aim at an expected running time O(k +n log n), which is optimal in the comparison model.

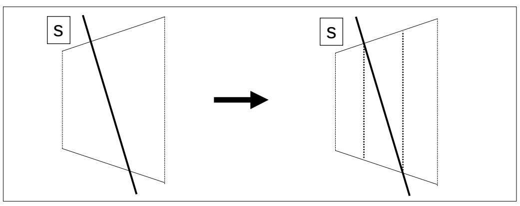
Incremental step

- We have built $H(S_i)$ and we want to insert the next segment s, to obtain $H(S_{i+1})$.
- We obtain from auxiliary functions the region R_0 in $H(S_i)$ containing the left endpoint of s p=p(s).
- From R_0, we navigate in $H(S_i)$ to the next region intersected by s, and so on until we reach the other endpoint q=q(s).
- The cost of traversing a region R is proportional to h(R), which is the number of vertices of $H(S_i)$ incident to R.



Split and Merge

Once we have the sequence of regions R_0,R_1.., met by s and the entry and exit point of s in each region, we : (1) split a region into trapezoids (2) clip vertical walls, (3) merge adiacent cells if wall was deleted.



Cost analysis

- The cost of the update operations to produce $H(S_{i+1})$ from $H(S_i)$, when we insert s is order of $\Sigma_R h(R)$, where R ranges over trapezoids in $H(S_i)$ met by s
- To locate the initial region containing the endpoint of s we maintain a function that for each endpoint gives the trapezoid containing it. And for each trapezoid R, the list L(R) of end points in it (conflict list).
- Updating the conflict list and auxiliary functions takes time $O(\Sigma_R |L(R)|)$, where R ranges over regions met by s in $H(S_i)$.

Backward analysis

- Now we apply the method of backward analysis.
- The cost of inserting s is expressed as: Σ_R h(R) + |L(R)| where R ranges over all trapezoids in $H(S_{i+1})$ adiacent to s.
- Now we consider a fixed set S_{i+1} and we consider every segment in it to be s with uniform probability.

$$E[Cost_i] = \frac{1}{i+1} \sum_{s \in S_{i+1}} \sum_{\substack{meet(R,s) \\ R \in H(S_{i+1})}} h(R) + |L(R)|$$

Backward analysis, cntd.

- Note that each trapezoid is adiacent to at most 4 segments.
- The sum of sizes of conflict lists over all faces is at most 2(n-i).
- The sum of values of h(R) over all R is proportional to $|H(S_{i+1})|$, which is $O(n+k_{i+1})$.
- Therefore the expected cost of the i-th phase is: $O((n + k_{i+1})/(i+1))$.

Bounding $E[k_i+1]$

- For each intersection v in H(S) we set a random variable I_v, where I_v =1 if v is in $H(S_i)$, 0 otherwise.
- Clearly for a fixed set S_i, $k_i = \Sigma_v I_v$. Thus $E[k_i] = \Sigma_v E[V]$.
- But E[V] is the probability that v is in S_i. Since this happens iff the two segments defining v are in S_i, this happens with probability i^2/n^2.
- Thus $E[k_i] = k(i^2/n^2)$.
- Finally the whole cost is O(Σ_i n/i + Σ_i $k(i/n^2)$ = O(n log n + k).

Point location in the decomposition

- Given a query point q, find the trapezoid in H(S) containing q.
- To answer this type of question we maintain the history(i) = $H(S_0)$,...,$H(S_i)$ as a compressed tree.
- Formally, history(0) is a root associated with the whole plane.
- Let s be the segment inserted at phase i. We obtain history(i+1) from history(i) by generating a new leaf for each new trapezoid and by placing these new nodes under the killed trapezoids meeting them.
- Note that the out-degree of each node is at most 4.
- Due to merges the data structure is a DAG not a tree.

Searching algorithm and analysis

- Searching for q. Basis: Locate q at the root. Step: If q is at a node at level i, check all children of that node at level i+1.
- The cost is constant for each level.
- Lemma: for a fixed point q, the expected length of the search path is O(log n).
- Proof: For each level i, let s_i be the segment added at phase i, and X_i a 0-1 random variable that is 1 iff the trapezoid containing q in $H(S_{i+1})$ is adiacent to s_i.
- Clearly the length of the search path is $\Sigma_i X_i$

Analysis, cntd.

- So the expected length of a search path is $\Sigma_i E[X_i]$
- Again, using backward analysis, the trapezoid in $H(S_{i+1})$ containing q is adiacent to 4 segments, each segment in S_{i+1} is equally likely to be s_i, so $E[X_i] = 4/(i+1)$.
- Summing over all levels we obtain O(log n).
- Using Chernoff technique this bound can be made to hold with high probability for any query point.
- Note that we can use this search data structure instead of the auxiliary functions, thus making this method completely on-line.

Simple polygons (sketch)

- For the special case of a simple polygon a better bound is obtainable (in lecture 8)
- Exercise: 3.1.3 in Mulmuley's.

A monotone chain is a sequence of segments sharing endpoints and meeting each vertical line at most once. Modify the algorithm so to build the trapezoidal map for a set of c monotone chains with total n segments and k intersections obtaining an expected time a bound O(n +k + clog c).