CPS234 Computational Geometry **15 September 2005**

Lecture 6: RIC for Segment Intersections

Lecturer: Pankaj K. Agarwal *Scribe: Amber Stillings*

6.1 Lecture Summary

This lecture will describe a Randomized Incremental Algorithm (RIC) for Segment Intersections. The analysis for this algorithm is given, and it is shown that the expected running time is $O(n \log n + k)$ where k is the number of intersection points, $0 \leq k \leq \binom{n}{2}$.

6.2 Randomized Incremental Construction Algorithm

6.2.1 Vertical Decomposition

$S = \{e_1, \ldots, e_n\} \subseteq \Re^d$

Vertical Decomposition of S ($VD(S)$)
- From each endpoint of S or intersection point of 2 segments in S, draw a vertical segment until it hits another segment of S or to infinity.
- Partitions space $\mid VD(S) \mid = O(n + k)$ trapezoids.

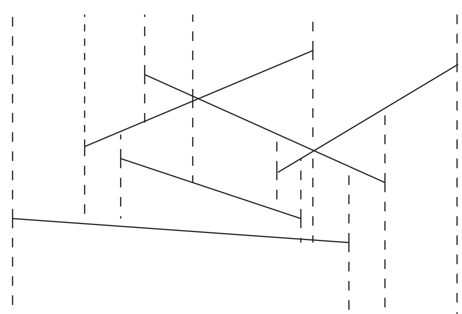

Figure 6.1: Planar Subdivision Graph

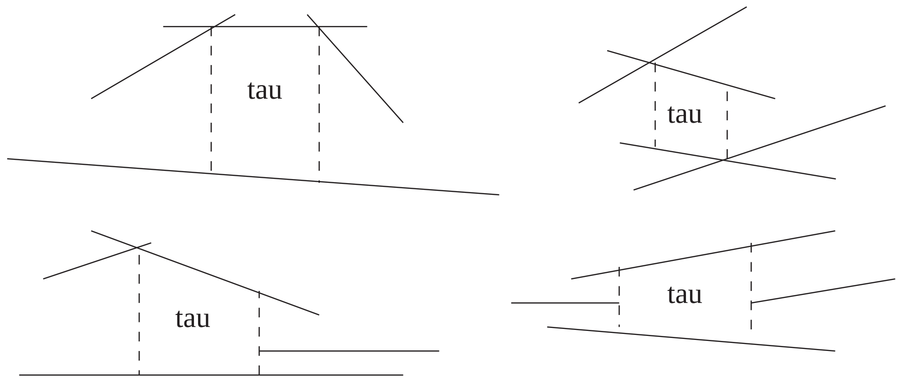

Figure 6.2: Types of Trapezoids (tau signifies τ)

If e is one of 4 segments bounding τ, then e defines τ.

$N(e, S) = \{\tau \in VD(S) \mid e \text{ defines } \tau\}$
$deg(e, S) = \mid N(e, S) \mid$
$I(e, S)_{e \notin S} = \{\tau \in VD(S) \mid e \text{ intersects } \tau\}$

6.2.2 RIC

Given a vertical decomposition, add another segment.

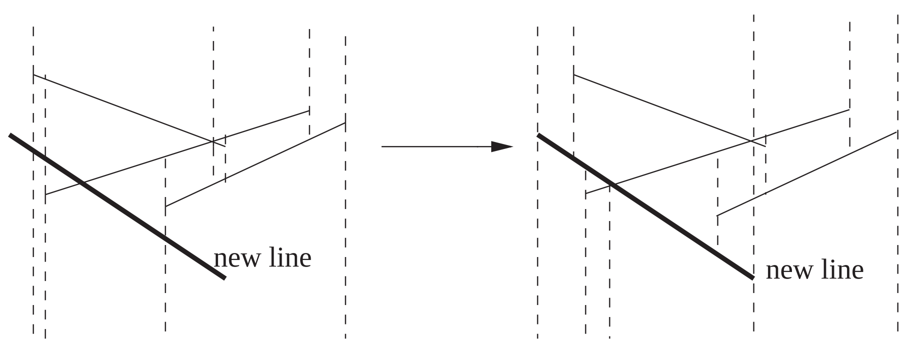

Figure 6.3: New Segment Added to Vertical Decomposition

6.2.3 Algorithm

$S_i = \{e_1, \ldots, e_i\}$
$VD(S_0) = \Re^2$
for i = 1 to n do
  Compute $I(e_i, S_{i-1})$
  Delete these cells
  Compute $N(e_i, S_{i-1} \cup e)$
  Add these cells $\Rightarrow VD(S_i)$
endfor

Claim 1 $VD(S_i) = VD(S_{i-1}) - I(e, S_{i-1}) + N(e, S_i)$

Find the cells (trapezoids) that intersect the new segment, modify them, and add any new cells.

6.2.4 Bookkeeping

Keep ”conflict lists” similar to bookkeeping with RIC and conflict hulls. More specifically, for each segment, keep track of the trapezoids it defines and vice versa.

Maintain $I(e_j, S_i)\ \forall j > i$
Foreach $\tau \in VD(S_i), S_\tau = \{e_j \mid \tau \in I(e_j, S_j)\}$

Time spent bookkeeping:

$$\sum_{\tau \in I(e, S_{i-1})} \mid S_\tau \mid$$

Now we need to look at the time spent creating/deleting cells, or just creating cells because deletion of cells is bounded by creation of cells.

6.3 Analysis

New cells created correspond to adjacent segments to that being inserted. So the number of cells created is $deg(s_i, S_i)$ where s_i is a random element of S_i.

$$\begin{aligned}
\frac{1}{\mid S_i \mid} \sum_{e \in S_i} deg(e, S_i) &= \frac{1}{\mid S_i \mid} \cdot \frac{1}{i} \cdot 4VD(S_i) \\
ExpectedValue &= \frac{1}{\binom{n}{i}} \sum_{S_i \in \binom{S}{i}} \frac{1}{i} \cdot 4 \mid VD(S_i) \mid \\
&= \frac{4}{i} \cdot \frac{1}{\binom{n}{i}} \sum_{S_i \in \binom{S}{i}} \mid VD(S_i) \mid \\
\Phi_i &= \frac{1}{\binom{n}{i}} \sum_{S_i \in \binom{S}{i}} \mid VD(S_i) \mid
\end{aligned}$$

Φ_i = Expected size of VD of a subset of size i of S.

Claim 2 $\Phi_i = O(i + k \cdot \frac{i^2}{n^2})$

$$\begin{aligned}
\Phi_i &= \frac{1}{\binom{n}{i}} \sum_{R \in \binom{S}{i}} \mid VD(R) \mid \\
&= O(\frac{1}{i}(i + k \cdot \frac{i^2}{n^2})) \\
\sum_{i=1}^{n} O(1 + k \cdot \frac{i^2}{n^2}) &= O(n + k)
\end{aligned}$$

Time Spent in Bookkeeping:
$(e, \tau) : e \notin S_i, \tau$ is a cell created in step i
$e \cap \tau \neq$ NULL

Probability Pr[τ is created in step i] = $\frac{4}{i}$
Expected time spent in bookkeeping in step i:

$$\frac{1}{\binom{n}{i}} \sum_{S_i \in \binom{S}{i}} \sum_{e \notin S_i} \frac{4}{i} \mid I(e, S_i) \mid$$

$(S_i \to R)$

$$\frac{1}{\binom{n}{i}} \sum_{R \in \binom{S}{i}} \frac{4}{i} \sum_{e \notin R} \mid VD(R) \mid - \mid VD(R \cup e) \mid + deg(e, R \cup e)$$

$\mid VD(R) \mid - \mid VD(R \cup e) \mid$ becomes a telescopic series; so ignore it

$$\begin{aligned}
\sum_{i=1}^{n} \frac{1}{\binom{n}{i}} \sum_{R \in \binom{S}{i}} \frac{4}{i} \sum_{e \notin R} deg(e, R \cup e) &= \frac{4}{i} \cdot \frac{1}{\binom{n}{i}} \sum_{R' \in \binom{S}{i+1}} \sum_{e \in R'} deg(e, R') \\
&= \frac{4}{i} \cdot \frac{1}{\binom{n}{i}} \sum_{R' \in \binom{S}{i+1}} 4 \mid VD(R') \mid \\
&= \frac{16}{i} \cdot \frac{1}{\binom{n}{i}} \sum_{R' \in \binom{S}{i+1}} \mid VD(R') \mid \\
\sum_{i=1}^{n} \frac{16(n-1)}{i(i+1)} \cdot \Phi_{i+1} &\leq \sum_{i=1}^{n} \frac{n}{i(i+1)} [(i+1) + k\frac{(i+1)^2}{n^2}] \\
&= \sum_{i=1}^{n} n[\frac{1}{i} + \frac{k}{n^2}] \\
&= O(n \log n + k)
\end{aligned}$$

Theorem 1 *Random Sampling Technique*
Choose Random Sample R
$R \subseteq S$: random sample of size r
$\tau \in VD(R)$
$w(\tau)$: number of segments in $S \setminus R$ that intersect τ

$$E[\sum_{\tau \in VD(R)} w(\tau)^P] \leq c \cdot \Phi_r \cdot (\frac{n}{r})^P$$

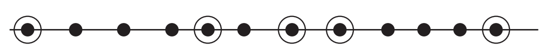

Figure 6.4: Random points, expect n/r elements in segments

6.4 Other Applications

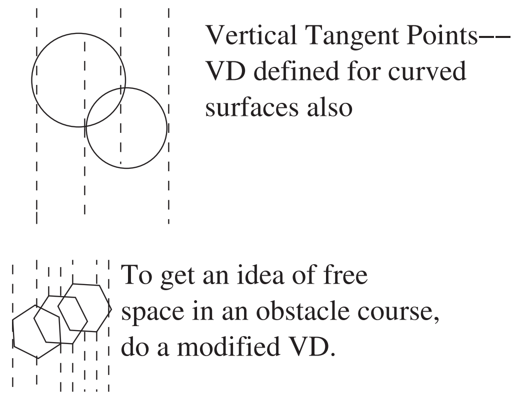

Figure 6.5: Shapes where VD is applicable

6.4.1 GIS

Elevation data collected: $M_1 = (x, y, f_1(x, y))$, f_1: elevation
Temperature data collected $M_2 = (x, y, f_2(x, y))$, f_2: elevation

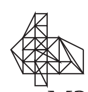

Figure 6.6: Shapes where VD is applicable

Do triangulation and piecewise interpolation.
Overlay maps. $M_3 = (x, y, g(x, y)), g = f_1 \bigoplus f_2$

6.4.2 Splines

This algorithm can also extend to splines as long as segments are monotone.

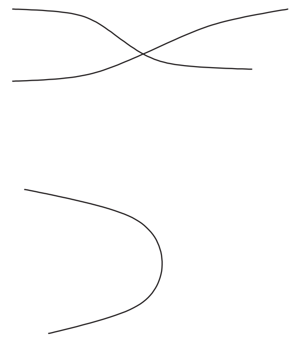

Figure 6.7: Splines – the first one monotone, the second not

6.5 Additional Information/Links

Boissonnat and Snoeyink discuss efficient algorithms using restricted predicates[1]. This algorithm works with both line segments and curves. This paper also discusses an algorithm for finding red/blue line and curve segment intersections, with the segments colored so that no 2 red and no 2 blue segments cross.

Hobby discusses a practical algorithm for segment intersection with finite precision output. [2] This paper is interesting because it summarizes the pros/cons of other algorithms.

Chazelle and Edelsbrunner present an optimal algorithm for finding line segment intersections in the plane [3]. This algorithm runs in $O(n \log n + k)$ time and uses at most $n + k$ storage. Amortized analysis is used to analyze the complexity of this algorithm.

References

[1] *Jean-Daniel Boissonnat and Jack Snoeyink, Efficient algorithms for line and curve segment intersection using restricted predicates In* Proceedings of the 15th Annual ACM Symposium on Computational Geometry, *1999*

[2] *John D. Hobby, Practical Segment Intersection with Finite Precision Output In* Technical Report 93/2-27, Bell Labs, *1993*

[3] *Bernard Chazelle and Herbert Edelsbrunner, An Optimal Algorithm for Intersecting Line Segments in the Plane In* Journal of the Association for Computing Machinery, *January 1992*