Exercises 3 - Range Queries[1]

Discussion: Wednesday, 25. November 2015

Lecture 5 – Range Queries I (18.11.2015)

Exercise 1 – Worst-Case Running Time. The running-time for range-queries in *kd*-trees is essentially based on the number of regions that must be checked. Let $Q(n)$ be that number.

1. Show that $Q(n)$ is described by the following recurrence:

$$Q(n) = \begin{cases} \mathcal{O}(1) & \text{, für } n = 1 \\ \mathcal{O}(1) + 2Q(n/4) & \text{, für } n > 1 \end{cases}$$

2. Show that $Q(n) = \mathcal{O}(\sqrt{n})$.
3. Show that $\Omega(\sqrt{n})$ is a lower bound for querying in *kd*-trees by defining a set of n points and a query rectangle appropriately.

Hint: For the details of range-queries on *kd*-trees see *Computational Geometry: Algorithms and Applications*

Exercise 2 – Partial Match Queries. *kd*-Trees can be used for *partial match queries*. A 2-dimensional partial match query specifies a value for one of the coordinates and asks for all points that have that value for the specified coordinate. Example: We query all points with x-coordinate 7.

1. Show that 2-dimensional kd-trees can answer partial match queries in $\mathcal{O}(\sqrt{n} + k)$ time, where k is the number of reported points.
2. Explain how to use a 2-dimensional range tree to answer partial match queries. What is the resulting running time?
3. Describe a data structure that uses linear storage and solves partial match queries in $\mathcal{O}(\log n + k)$ time.

Exercise 3 – Range Counting Queries. In some applications one is interested only in the number of points that lie in a range rather than in reporting all of them. Such queries are often referred to as *range counting queries*. In this case one would like to avoid having an additive term of $O(k)$ in the running time.

[1]Based on: M. de Berg, O. Cheong, M. van Kreveld, M. Overmars: Computational Geometry, 3rd ed., Springer-Verlag, 2008.

1. Describe how a 1-dimensional range tree can be adapted such that a range counting query can be performed in $O(\log n)$ time. Prove the query time bound.
2. Describe how to perform d-dimensional range counting queries.

Exercise 4 – Complex Objects. In many applications one wants to apply range queries on complex objects.

1. Let S be a set of n axis-aligned rectangles in the plane. We want to report all rectangles in S that lie in the query rectangle $[x, x'] \times [y, y']$. Describe a data structure that solves this problem using $O(n \log^3 n)$ storage and $O(\log^4 n + k)$ time for a query, where k is the number of reported rectangles.
2. Let P be a set of n polygons in the plane. We want to report all polygons in P that completely lie in the query rectangle $[x, x'] \times [y, y']$. Describe a data structure that solves this problem using $O(n \log^3 n)$ storage and $O(\log^4 n + k)$ time for answering a query, where k is the number of reported polygons.

Lecture 6 – Range Queries II (23.11.2015)

Exercise 5 – Interval-Trees. In the lecture we have considered the following problem for querying segments with arbitrary orientation.

SegmentQuery
Given: n disjoint segments and a axis-aligned rectangle $R = [x, x'] \times [y, y']$
Find: All segments that intersect R.

We have solved this problem using segment trees. Can we also use interval trees? Which problems may arise?

Exercise 6 – Segment-Trees – Construction. In the lecture we have seen the following theorem:

Theorem 1. *Let S be a set of horizontal (axis-parallel) line segments in the plane. All k line segments that intersect a vertical query segment (an axis-parallel rectangle R) can be found in $O(\log^2(n) + k)$ time. The data structure requires $O(n \log n)$ space and $O(n \log n)$ construction time.*

We now show that the data structure can be constructed in $O(n \log n)$ time. Let $s, s' \in S$ be two line segments. The segment s lies *below* s' ($s \prec s'$), if there are points $p \in s$ and $p' \in s'$ with $x(p) = x(p')$ and $y(p) < y(p')$, where $x(p)$ denotes the x-coordinate of p and $y(p)$ denotes y-coordinate of p.

1. Show that the relation $\prec$ defines a acyclic relation on S.
2. Describe an algorithm, that computes such an order on S in $O(n \log n)$ time.
   Hint: Use a sweep-line-procedure.
3. Show that the data structure of Theorem 1 can be constructed in $O(n \log n)$ time.

Exercise 7 – Counting Intervals. Let I be a set consisting of n intervals. Describe a data structure, by means of which the number of intervals containing a given point $p \in \mathbb{R}$ can be determined. To that end consider the following variants.

1. The data structure is based on interval trees.
2. The data structure is based on segment trees.
3. The data structure uses neither interval trees nor segment trees.

Exercise 8 – Intersection of Rectangles. Let $\mathcal{R}$ denote a set of n axis-aligned rectangles in the plane. For a point $p \in \mathbb{R}^2$ the number of rectangles containing p is denoted by $w_{\mathcal{R}}(p)$. Describe an algorithm that computes $\max_{p \in \mathbb{R}^2} w_{\mathcal{R}}(p)$ in $O(n \log n)$ time.

Hint: Use segment trees and a sweep-line-procedure.