MAC 5747 GEOMETRIA COMPUTACIONAL

LISTA DE EXERCÍCIOS 4

1. Instruções

Entregue os exercícios 1 e 6 na aula do dia 16 de maio, quarta-feira.

2. Exercícios

(1) (Exercício 35.2-2 de [1]) Dados dois segmentos de reta a e b que não se intersectam e são comparáveis na abscissa x , mostre como usando produto interno (i.e. as primitivas `Left`, ou `Lefton`, ou $\ldots$, ou o que você precisar) podemos decidir se $a >_x b$ em tempo $O(1)$.
[Note que no algoritmo `Intersecção-de-Segmentos` as operações `Insert`, `Delete`, `Above` e `Below` usam esta comparação.]

(2) (Exercício 2.1 de [2]) Seja S um conjunto de n segmentos de reta disjuntos cujo ponto extremo superior pertence a reta $y = 1$ e o ponto inferior pertence a reta $y = 0$. Estes segmentos particionam a faixa horizontal $[-\infty : \infty] \times [0 : 1]$ em $n + 1$ regiões (veja Figura 1). Descreva um algoritmo de complexidade de tempo $O(n \log n)$ que contrói uma árvore de busca binária com os segmentos de S tal que a região contendo um dado ponto possa ser determinada em tempo $O(\log n)$. Também descreva o algoritmo de busca em detalhes.

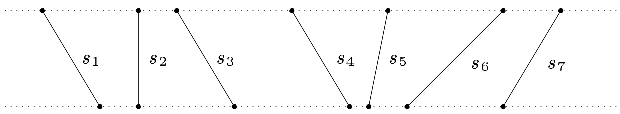

Figura 1. Segementos particionando a faixa $[-\infty : \infty] \times [0 : 1]$.

(3) (Exercício 35.2-3 de [1]) O professor Wellsmart sugeriu que modificássemos o algoritmo `Intersecção-de-Segmentos` de tal forma que ao invés do algoritmo parar logo após detectar uma intersecção o algoritmo imprime os segmentos que se intersectam e continua executando a próxima iteração do laço **para** (**for** *loop*). O professor chama o algoritmo resultante de `Imprime-Segmentos-Intersectantes` e afirma que esse algoritmo imprime todas as intersecções, da esquerda para a direita, a medida que elas vão ocorrendo. Mostre que o professor Wellsmart está duplamente enganado. Primeiro mostre um exemplo onde a primeira intersecção encontrada pelo algoritmo

`Imprime-Segmentos-Intersectante` não é a mais à esquerda e depois mostre um exemplo onde o algoritmo não encontra todas as intersecções.

(4) (Exercício 35.2-4 de [1]) Mostre um algoritmo de complexidade de tempo $O(n \log n)$ que decide se um polígono de n vértices é simples.

(5) (Exercício 35.2-5 de [1]) Mostre um algoritmo de complexidade de tempo $O(n \log n)$ que decide se dois polígonos simples com um total de n vértices se intersectam.

(6) (Exercício 35.2-2 de [1]) Um *disco* (ou circulo) é a região do plano limitada por um circunferência. Um disco é dado através das coordenadas do seu centro e do valor do seu raio. Dois discos intersectam se eles tem algum ponto em comum. Descreva um algoritmo de complexidade de tempo $O(n \log n)$ que dados n discos decide se qualquer dois deles intersectam.

(7) (Exercício 2.11 de [2]) Dado um conjunto S de n circunferências no plano. Descreve um algoritmo por linha-de-varredura que determina todas as intersecções entre as circunferências. Como estamos tratando de circunferências aqui, e não discos, duas circunferências não se intersectam se um esta inteiramente no interior da região limitada pela outra. O seu algoritmo deve ter complexidade de tempo $O((n + I) \log n)$, onde I é o número de intersecções.

(8) (Exercício 35.2-7 de [1]) Dado um conjunto de n segmentos de reta contendo um total de k intersecções, mostre um algoritmo que devolve as I intersecções em tempo $O((n + I) \log n)$.
[Sugestão. Veja página 284 de [3] ou as notas de aula da disciplina.]

(9) (Exercício 2.14 de [2]) Dado um conjunto S de n segmentos de retas disjuntos no plano e um ponto p que não pertence a nenhum dos segmentosde S. Desejamos determinar todos os segmentos de S que são visíveis a partir de p, isto é, todos os segmentos de S que contém algum ponto q tal que o segmento aberto pq não intersecta nenhum dos segmentos de S. Descreva um algoritmo de complexidade de tempo $O(n \log n)$ para este problema. O seu algoritmo deve usar uma semireta com extremidade p e rotacionando ao redor de p (como os radares que vocês veêm nos fimes).

Referências

[1] T.H. Cormen, C.E. Leiserson e R.L. Rivest, *Introduction to algorithms*, The MIT Electrical Engineering and Computer Scienece Series, The MIT Press, MacGraw-Hill Book Company, 1990, QA758 C811i.

[2] M. de Berg, M. van Kreveld, M. Overmars e O. Schwarzkopf, *Computational geometry: Algorithms and applications*, Springer, 1997, Second Edition, 2000.

[3] F.P. Preparata e M.I. Shamos, *Computational geometry: An introduction*, Texts and Monographs in Computer Science, Springer-Verlag, New York, 1985, QA758 P927c.