Geometria Computacional

Departamento de Ciência da Computação – IME/USP

Segundo Semestre de 2007

Lista 3

(1) [Exercício 2.1 do livro de Berg e outros] 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 $O(n \lg n)$ 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 $O(\lg n)$. Também descreva o algoritmo de busca em detalhes.

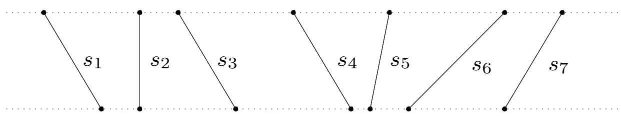

FIGURA 1. Segmentos particionando a faixa $[-\infty : \infty] \times [0 : 1]$.

(2) [CLRS 33.2-2] 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`, `Lefton`, ou o que você precisar) podemos decidir se $a >_x b$ em tempo $O(1)$.
[Note que, nos algoritmos `Detecta-Intersecção` e `Acha-Intersecções` (das transparências da aula 7), as operações `Insere`, `Remove`, `Predecessor` e `Sucessor` usam esta comparação.]

(3) [CLRS 33.2-3] O professor Wellsmart sugeriu que modificássemos o algoritmo `Detecta-Intersecção` de tal forma que, em vez do algoritmo parar logo após detectar uma intersecção, o algoritmo imprimisse os segmentos que se intersectam e continuasse executando a próxima iteração do laço **para**. 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) [CLRS 33.2-4] Mostre um algoritmo de complexidade de tempo $O(n \lg n)$ que decide se um polígono de n vértices é simples.
(5) [CLRS 33.2-5] Mostre um algoritmo de complexidade de tempo $O(n \lg n)$ que decide se dois polígonos simples com um total de n vértices se intersectam.
(6) [CLRS 33.2-6] 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 $O(n \lg n)$ que dados n discos decide se qualquer dois deles intersectam.
(7) [Exercício 2.11 do livro de Berg e outros] 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 $O((n + I) \lg n)$, onde I é o número de intersecções.
(8) [CLRS 33.2-7] Dado um conjunto de n segmentos de reta contendo um total de k intersecções (com multiplicidades), mostre um algoritmo que devolve as k intersecções em tempo $O((n + k) \lg n)$.

    [Observação. Este é o algoritmo que está parcialmente escrito nas transparências da aula 7.]
(9) [Exercício 2.14 do livro de Berg e outros] Dado um conjunto S de n segmentos de reta disjuntos no plano e um ponto p que não pertence a nenhum dos segmentos de 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 $O(n \lg n)$ 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 filmes).