Sabrina Araújo da Silva nºUSP 12556018 2

MAC0338 - ANÁLISE DE ALGORITMOS

LISTA 4

18. A remoção da superfície escondida é um problema em computação gráfica que raramente precisa de introdução: quando o João tá na frente da Maria, você pode ver o João, mas não a Maria; quando a Maria tá na frente do João, ... Você entendeu a idéia.

    A beleza desse problema é que você pode resolvê-lo mais rapidamente do que a intuição em geral sugere. Aqui está uma versão simplificada do problema onde já podemos apresentar um algoritmo mais eficiente do que a primeira solução em que se pode pensar. Imagine que são dadas n retas não verticais no plano, denotadas por $L_1, \ldots, L_n$. Digamos que L_i é dada pela equação $y = a_i x + b_i$, para $i = 1, \ldots, n$. Suponha que não há três retas entre as retas dadas que se interseptam mutuamente num mesmo ponto. Dizemos que a reta L_i é a *mais alta* numa dada coordenada $x = x_0$ se sua coordenada y em x_0 é maior que a coordenada y em x_0 de todas as outras retas dadas. Ou seja, se $a_i x_0 + b_i > a_j x_0 + b_j$ para todo $j \neq i$. Dizemos que L_i é *visível* se existe uma coordenada x na qual ela é a mais alta. Intuitivamente, isso corresponde a uma parte de L_i ser visível se você olhar para baixo a partir de $y = \infty$.

    Escreva um algoritmo $O(n \lg n)$ que recebe uma sequência de n retas, como descrito acima, e devolve a subsequência delas que é visível.

Seja A o vetor de retas

```
SUBSEQUENCIA (A)
1   n ← tamanho da lista A
2   se n > 3
3       lista1 = SUBSEQUENCIA (A[1 ... n/2])
4       lista2 = SUBSEQUENCIA (A[(n/2)+1 ... n])
5       retorna VISIVEIS (lista1, lista2)
6   senão se n = 3
7       interseccao1 = |(b2 - b1) / (a1 - a2)|
8       interseccao2 = |(b3 - b1) / (a1 - a3)|
9       se interseccao1 > interseccao2 então
10          remove A[2]
11  senão se n = 2
12      se a1 é igual a2
13          se b1 > b2
14              remove A[2]
15          senão
16              remove A[1]
17  retorna A
```

```
VISIVEIS (lista 1, lista 2)
1    B ← INTERCALA (lista 1, lista 2)  // intercala em tempo O(n)
2    V[1] ← B[1]    // adiciona as duas primeiras retas no vetor de retas visíveis
3    V[2] ← B[2]
4    para cada reta l em B
5        interseccao1 = intersecção da última reta com a penúltima
6        interseccao2 = intersecção da reta l com a penúltima
7        enquanto o tamanho de V[] for ≥ 2 e interseccao1 < interseccao2
8            // a reta l intercepta a penúltima anterior antes da anterior
9            remove o último item de V[]
10       adiciona l ao vetor V[]
11   retorna V[] // vetor de retas visíveis
```

- Na função VISIVEIS() o algoritmo ordena a lista em tempo $O(n \lg n)$
- A linha 4 até a linha 10 do algoritmo consome tempo $O(n)$, pois em cada passo $O(n)$ uma linha é adicionada ou removida da lista de retas visíveis, assim, a mesma linha vão é adicionada e removida da lista mais de uma vez.
- Em VISIVEIS() o algoritmo consome tempo $O(n)$
- Em SUBSEQUENCIA() o algoritmo divide o problema em dois subproblemas de tamanho $\frac{n}{2}$.
- Tem-se que:

$$T(n) = 2T(n/2) + O(n) = O(n \lg n)$$

Portanto, o algoritmo consome tempo $O(n \lg n)$.