MAC0331 GEOMETRIA COMPUTACIONAL

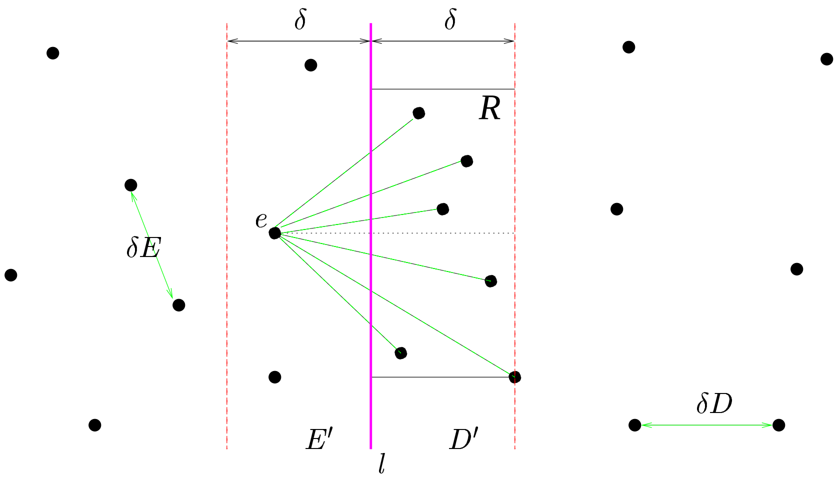
LISTA DE EXERCÍCIOS 1

1. Instruções

Entregue os exercícios 1, 3 e 6 na aula do dia 29 de agosto ou até as 17:00 do mesmo dia na secretaria do Departamento de Ciência da Computação (sala 256 do Bloco A).

2. Exercícios

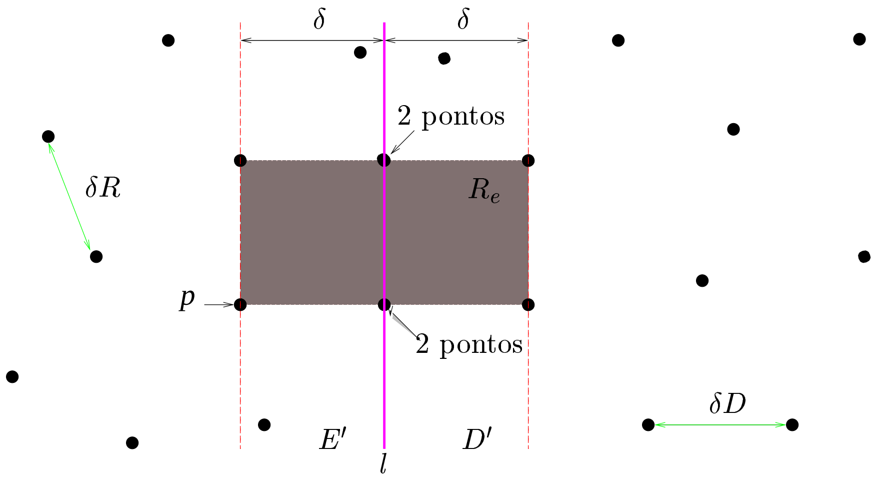
1. (Suficiente) Considere a fase Combinar do algoritmo de divisão-e-conquista para o Problema do Par Mais-Próximo, visto em sala de aula. Mostre que, nesta fase, para cada ponto $e \in E'$ é **suficiente** o algoritmo calcular a distância entre e e no máximo 6 pontos em D'.
2. (Necessário) Considere a fase Combinar do algoritmo por divisão-e-conquista para o Problema do Par Mais-Próximo, visto em sala de aula. Nesta fase, para cada ponto $e \in E'$ o algoritmo calcula a distância entre e e no máximo 6 pontos em D'. Você consegue mostrar um exemplo onde é realmente **necessário** calcularmos a distância entre e e 6 pontos $d_1, \ldots, d_6$ de D'? (Ou seja, o seu exemplo deve mostrar um ponto e em E e pontos $d_1, \ldots, d_6$ contidos no retângulo R_e de tal forma que se o algoritmo calcula a distância entre e e cada um dos pontos (digamos) $d_1, \ldots, d_5$ então o algoritmo não devolve o par mais-próximo (que por azar é o par $\{e, d_6\}$.) Se você não conseguir encontrar um tal exemplo, então tente mostrar que é **suficiente** o algoritmo calcular a distância entre cada ponto de E' e menos do que 6 pontos de D'.



3. Considere agora a seguinte pequena variação da fase COMBINAR do algoritmo. Seja M a lista dos pontos em $E' \cup D'$ ordenados em relação as suas y-coordenadas (ordenadas). Percorra a lista M e para cada ponto $m \in M$ calcule a distância entre m e os 7 pontos que seguem m na lista M. (Note que algumas vezes poderemos estar calculando a distância entre pontos de E' ou entre pontos de D'.) Mostre que com esta variação da fase COMBINAR o algoritmo corretamente encontra um par mais-próximo.

   [Note que com esta variação não precisamos mais ficar testando as y-coordenadas dos pontos que terão as suas distâncias calculadas.

   Sugestão: Veja a figura abaixo ou de uma olhada em [1] página 910.]



4. Considere ainda a variação da fase COMBINAR proposta no exercício anterior. Você mostrou que é **suficiente** calcularmos as distâncias entre cada ponto $m \in M$ e os 7 pontos que seguem m na lista M para que o algoritmo esteja correto. Será que é **necessário** calcularmos essas 7 distâncias? Ou seja, será que não poderiamos calcular a distância entre cada ponto $m \in M$ e os (digamos) 6 pontos que seguem m na lista M e ainda assim o algoritmo estaria correto? Apresente um exemplo mostrando que é necessário calcularmos essas 7 distâncias ou mostre que é possível encontrarmos o par mais próximo calculando menos do que 7 distâncias para cada $m \in M$.

   [Note que para a ordem de complexidade do algoritmo tanto faz se para cada $m \in M$ calculamos 7, 6 ou 100000 distâncias—o algoritmo continua tendo complexidade de tempo $O(n \log n)$.]
5. (Exercício 33.4-1 de [1]) Considere ainda a variação da fase COMBINAR proposta no Exercício 3. O professor Maqui Sperto teve uma idéia genial e veio com um novo esquema para que o algoritmo encontre o par mais-próximo verificando, na sua fase COMBINAR, somente a distância entre cada ponto $m \in M$ e os 5 pontos que estão a seguir de m em M. A idéia é sempre colocar os pontos da reta l no conjunto E. Então, não poderá haver um par de pontos coincidentes sobre a reta l com um ponto em E e outro ponto em D. Portanto, no máximo 6 pontos podem estar na

retâgulo de dimensões $\delta \times 2\delta$. Onde está a bobagem no esquema proposto pelo professor Sperto.

6. (Exercício 33.4-2 de [1]) Sem aumentar a complexidade de tempo assintótica do algoritmo, mostre como garantir que o conjunto de pontos passados para a primeira chamada recursiva do algoritmo não contenha pontos coincidentes. Prove que então é suficiente que o algoritmo verifique os 6 (e não 7) pontos que seguem cada ponto em M. Por que não é suficiente verificar somente 5 pontos? Ou é suficiente?
7. (Exercício 33.4-3 de [1]) A distância entre dois pontos pode ser definida de diversas maneiras além da Euclidiana. No plano, a L_m-distância entre dois pontos $p = (p_x, p_y)$ e $q = (q_x, q_y)$ é dada por $(|p_x - q_x|^m + |p_y - q_y|^m)^{1/m}$. Portanto, a distância Euclidiana é a L_2-distância. Modifique o algoritmo de divisão-e-conquista para o Problema do Par Mais Próximo de tal forma que ele devolva o par de pontos mais-próximo em relação a L_1-distância (também conhecida como *Manhattan distance*).
8. (Exercício 33.4-4 de [1]) Dados dois pontos p e q no plano, a L_∞-distância entre eles é dada por $\max\{|p_x - q_x|, |p_y - q_y|\}$. Modifique o algoritmo para o par mais próximo para que ele encontre um par mais-próximo de acordo com a L_∞-distância.

Referências

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