MAC 5747 – Geometria Computacional
Prof. José Coelho de Pina

LISTA DE EXERCÍCIOS 1 – RESPOSTAS

1. (Suficiente) Considere a fase de 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 $p \in P'_L$ é **suficiente** o algoritmo calcular a distância entre p e no máximo 6 pontos em P'_R.

   [**Resposta por Mário Leston Rey**].

   Denominaremos um conjunto de pontos de $\delta-$viável se a distância entre quaisquer dois pontos distintos do conjunto é $\geq \delta$. É suficiente mostrar que no retângulo R de dimensões $\delta \times 2\delta$ não existe um conjunto $\delta-$viável de 7 pontos. A justificativa segue da seguinte observação:

   Num quadrado Q de dimensão δ, qualquer conjunto $\delta-$viável tem cardinalidade ≤ 4; ainda mais, um conjunto $\delta-$viável com 4 pontos está diposto de tal maneira que cada ponto está sob um e só um vértice de Q. $(*)$

   Particione R em quadrados de dimensão δ. Denomine-os de Q_1 e Q_2. Em virtude de $(*)$, qualquer conjunto $\delta-$viável pode possuir no máximo quatro pontos em Q_2. Agora, se existem exatamente quatro pontos em Q_1, então dois de tais pontos estão em u e v. Assim Q_1 e Q_2 compartilham dois pontos e tal conjunto $\delta-$viável tem, no máximo, seis pontos.

   Obs.: A justificativa de $(*)$ é na seguinte linha: é claro que existe um conjunto $\delta-$viável com quatro pontos. Suponha que existe um tal conjunto que não esteja "colocado" sobre os vértices de Q. Então tal conjunto determina um quadrilátero (fácil ver) Q' tal que área(Q') < área$(Q) = \delta^2$. Mas, neste caso, um dos lados de (Q') tem tamanho $< \delta$; contradição. É evidente que um quinto ponto não pode ser inserido.

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 $p \in P^*$ e os 5 pontos que estão a seguir de p em P^*. A idéia é sempre colocar os pontos da reta l no conjunto P_L. Então, não poderia haver um par de pontos coincidentes sobre a reta l com um ponto em P_L e outro ponto em P_R. Portanto, no máximo 6 pontos podem estar no retângulo de dimensão $\delta \times 2\delta$. Onde está a bobagem no esquema proposto pelo professor Sperto?

   Duas respostas possíveis:

   (a) [**Resposta por Mário Leston Rey**].
   A bobagem (ou uma delas) está no seguinte: suponha que todos os pontos estejam sobre uma reta vertical l. Nesta caso, a fase de DIVIDIR nada divide e o algoritmo não termina!

   (b) Na Figura 1 podemos ver que mais de 6 pontos podem estar no retângulo de dimensão $\delta \times 2\delta$, bastando colocar um sétimo ponto na área hachurada.

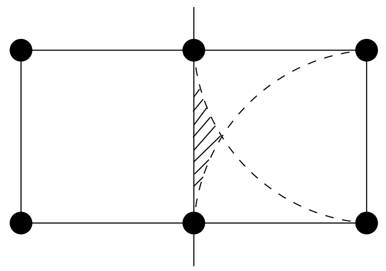

Figura 1:

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 concidentes. Prove que então é suficiente que o algoritmo verifique os 6 (e não 7) pontos que seguem cada ponto em P^*. Por que não é suficiente verificar somente 5 pontos?

   [**Resposta por Mário Leston Rey**].

   Ordene a coleção de pontos segundo a seguinte relação de ordem:

   $$(x_1, y_1) \leq (x_2, y_2) \text{ se } x_1 < x_2 \text{ ou } x_1 = x_2 \text{ e } y_1 \leq y_2.$$

   Seja x' o valor assim obtido. Execute o seguinte algoritmo:

   $x[1] \leftarrow x'[1]$
   $j \leftarrow 1$
   para $i \leftarrow 2$ **até** comprimento$[x']$ **faça**
       se $x'[i] \neq x[j]$ **então**
           $j \leftarrow j + 1$
           $x[j] \leftarrow x'[i]$

   Copie o elemento de x para um vetor y e ordene y segundo a y-coordenada. Isto finaliza a fase de pré-processamento. Basta agora executar o algoritmo com os vetores x e y. É claro que esta fase de pré-processamento tem consumo de tempo $O(n \log n)$, onde n é o número de pontos da coleção.

   Nota do Monitor

   A última parte deste exercício deve ser pensada da seguinte forma. Para mostrar que não é suficiente verificar somente 5 pontos, deve-se mostrar um exemplo em que os pontos $p \in P_L$ e $q \in P_R$ estão à distância $< \delta$ um do outro, existem três pontos entre p e q na ordenação (pela y-coordenada) P^*, e o algoritmo não devolve (p, q) como par mais próximo. De fato, não existe um tal exemplo.