Recherche des deux points les plus rapprochés

Manon Ruffini

Entrée : Un tableau Q de $n \geqslant 2$ points du plan
Sortie : Les deux points de Q les plus rapprochés (pour la distance euclidienne)
Un algorithme naïf consisterait à tester toutes les paires de points. Il y en a $\binom{n}{2} = \Theta(n^2)$.

Étape 1

On va utiliser le paradigme diviser pour régner.

Soit P un sous-ensemble de Q. Soit X un tableau contenant les points de P triés par ordre croissant des abscisses ; Soit Y un tableau contenant les points de P triés par ordre croissant des ordonnées. Si $|P| \leqslant 3$, on utilise l'algorithme naïf. Comme ça, on appelle jamais l'algorithme sur un ensemble de cardinal $\leqslant 1$. Sinon :

Diviser : On trouve une droit Δ qui sépare les points de P en P_G et P_D qui sont les points à droite et à gauche de Δ, avec $|P_G| = \lfloor \frac{|P|}{2} \rfloor$ et $|P_D| = \lfloor \frac{|P|}{2} \rfloor + 1$.

Régner : On appelle l'algorithme récursivement sur P_G et P_D. Soient δ_G et δ_D les plus petites distances pour P_G et P_D. Soit $\delta = \min(\delta_G, \delta_D)$

Combiner : La paire de points recherchée est soit celle espacée de δ, soit une telle qu'un point soit dans P_G et l'autre dans P_D. On cherche s'il existe une telle paire espacée de moins de δ. Si ces deux points existent, ils doivent être espacés de moins de δ de la droite Δ.

Étape 2

Implémentation de l'algorithme

Algorithme 1 : PPR(P)

$X \leftarrow$ trier Q par abscisses croissantes;
$Y \leftarrow$ trier Q par ordonnées croissantes;
PPR-rec(X, Y)

Algorithme 2 : PPR-rec(X, Y)

---

si $X \leqslant 3$ **alors**
  | **retourner** *PPR-naif*(X)
$(X_G, X_D, Y_G, Y_D) \leftarrow$ PPR-diviser(X, Y);
$\delta_G \leftarrow$ PPR-rec(X_G, Y_G);
$\delta_D \leftarrow$ PPR-rec(X_D, Y_D);
$\delta \leftarrow \min(\delta_G, \delta_D)$;
$Y' \leftarrow$ extraire de Y les points espacés de moins de δ de Δ;
`/* Comme dans PPR-diviser */`
$\delta' \leftarrow$ PPR-bande(Y');
retourner $\min(\delta, \delta')$

Étape 3

Description de PPR-diviser

On veut divise X et Y en X_G, X_D, Y_G, Y_D qui sont les points de P_G et P_D triés par ordre croissant des abscisses ou des ordonnées.

Algorithme 3 : PPR-diviser(X, Y)

$X_G \leftarrow X\,[1, \lfloor |X|/2 \rfloor]$;
$X_D \leftarrow X\,[\lfloor |X|/2 \rfloor + 1, |X|]$;
$x \leftarrow X[\lfloor |X|/2 \rfloor]$.abs ;
$Y_G \leftarrow$ tableau de taille $|X_G|$;
$Y_D \leftarrow$ tableau de taille $|X_D|$;
$t_G \leftarrow 1$;
$t_D \leftarrow 1$;
pour $1 \leqslant i \leqslant |Y|$ **faire**
  si $Y[i].abs \leqslant x$ **alors**
    $Y[t_G] \leftarrow T[i]$;
    $t_G \leftarrow t_G + 1$
  sinon
    $Y[t_D] \leftarrow T[i]$;
    $t_D \leftarrow t_D + 1$
  fin
fin
retourner (X_G, X_D, Y_G, Y_D)

Cet algorithme renvoie bien les tableaux X_G, X_D, Y_G, Y_D des points à droite et à gauche de Δ triés par abscisses et ordonnées croissantes. La complexité est en $O(n)$, où $n = |X| = |Y|$.

Étape 4

Description de PPR-bande

On note $b_1 \dots b_r$ les points de Y' triés par ordonnées croissantes. On va utiliser le résultat suivant :

Proposition 1

Il existe $b_i \neq b_j \in Y'$, avec $d(b_i, b_j) < \delta$ ssi il existe $i, j \in [|1, r|]$ tels que : $i < j \leqslant i + 7$ et $d(b_i, b_j) < \delta$

Démonstration : — " $\Leftarrow$ " : immédiat.

— " $\Rightarrow$ " : On suppose qu'il existe $b_i \neq b_j \in Y'$, avec $d(b_i, b_j) < \delta$. (OPS $i < j$)
Alors b_i et b_j sont dans un rectangle R de dimension $\delta \times 2\delta$ centré en Δ. Quitte à translater, on peut supposer que b_i est sur le côté inférieur de R. Montrons qu'au plus 8 points de P se trouvent dans R.
Pour cela, considérons le carré $\delta \times \delta$ formant la moitié gauche de ce rectangle. Puisque tous les points de P_G sont distants d'au moins δ, il y en a au plus 4 qui se trouvent dans ce carré (1 par sous-carré $\delta/2 \times \delta/2$ de diagonale $\sqrt{2}\delta/2 < \delta$).
De même, il y a au plus 4 points qui se trouvent dans la moitié droite de R. Donc, R contient au plus 8 points et on a bien (les b_i sont classés par ordonnées croissantes) :

$$i < j \leqslant i + 7 \text{ et } d(b_i, b_j) < \delta$$

■

Ainsi, on a l'algorithme suivant :

Cet algorithme a une complexité en $O(|Y'|)$ donc en $O(n)$.

Algorithme 4 : PPR-bande(Y')

```
δ' ← δ;
pour 1 ⩽ i ⩽ |Y'| − 1) faire
    pour i + 1 ⩽ j ⩽ max(i + 7, |Y'|) faire
        si d(Y'[i], Y'[j]) < δ' alors
            δ' ← d(Y'[i], Y'[j])
    fin
fin
retourner (δ')
```

Étape 5

Complexité de PPR.

Soit $T(n)$ la complexité de PPR-rec pour n points. On a :

$$T(n) = \begin{cases} 2T(n/2) + O(n) \quad \text{si } n > 3 \\ O(1) \text{ si } n \leqslant 3 \end{cases}$$

D'où

$$T(n) = O(n \log n)$$

Finalement, l'algorithme PPR est en $O(n \log n)$

Références

[1] Thomas H. Cormen, *Algorithmique*. Dunod, 3è édition, 2010.