Examen DEA Images et Vision 2003-2004 : géométrie algorithmique
3h — 9 mars 2004

1 Recherche de plus proche voisin

Étant donné un ensemble $\mathcal{S}$ de n points dans le plan et un point du plan p on cherche à déterminer $NN_{\mathcal{S}}(p)$ le point de $\mathcal{S}$ le plus proche de p. On note $DT(\mathcal{S})$ la triangulation de Delaunay de $\mathcal{S}$.

w est initialisée avec un sommet quelconque de $DT(\mathcal{S})$.

```
N = un sommet de DT(S);
dmin = |pN|;
Faire {
        v = N;
        Pour chaque voisin w de v dans DT(S) {
                si ( dmin > |pw| ) {
                dmin = |pw|;
                N = w;
                }
        }
} tant que (N ≠ v);
return N;
```

1.1

Démontrer que ce programme calcule le plus proche voisin de v.

1.2

Quelle est la complexité dans le cas le pire de cet algorithme? Donner un exemple réalisant cette pire complexité ($\forall n$).

1.3

Si on modifie l'algorithme comme suit:
Variante :

$N =$ un sommet de $DT(\mathcal{S})$;
$dmin = |pN|$;
Faire {
    $v = N$;
    Pour chaque voisin w de v dans $DT(\mathcal{S})$ {
        si ($dmin > |pw|$) {
        $dmin = |pw|$;
        $N = w$;
        break; *// Sortir de la boucle "Pour"*
        }
    }
} tant que ($N \neq v$);
return N;

Qu'apportent la variante?
Le nombre d'exécutions de l'instruction $N = w$ est-il modifié?
Dans quel sens?
Est-ce que l'on calcule bien toujours le plus proche voisin?
Commentaires?

2 Triangulation optimisant les cercles circonscrits

Étant donné un ensemble de n points du plan $\mathcal{S}$ et une triangulation $\mathcal{T}(\mathcal{S})$ de cet ensemble de points, on définit $\mu(\mathcal{T}(\mathcal{S}))$ le maximum des rayons des cercles circonscrits aux triangles de $\mathcal{T}(\mathcal{S})$.

2.1

Trouver une triangulation qui minimise $\mu(\mathcal{T}(\mathcal{S}))$. Algorithme, complexité...

2.2

Trouver une triangulation qui maximise $\mu(\mathcal{T}(\mathcal{S}))$ **dans le cas particulier** où les points de $\mathcal{S}$ forment un polygone convexe. Algorithme, complexité...

3 Plus petit carré englobant.

Étant donné un ensemble de n points du plan $\mathcal{S}$ on cherche le plus petit carré contenant ces n points. On supposera que l'on n'a pas trois points alignés.

3.1

Pour commencer on cherche un carré dont les cotés sont parallèles à une certaines direction (repéré par son angle θ avec l'axe des x). Donner un algorithme et sa complexité. En général, combien de points de $\mathcal{S}$ sont-ils sur le bord de ce carré.

3.2

A quel moments les points définissant le plus petit carré peuvent-ils changer lorsque θ augmente?

3.3

On appelle ces valeurs de θ valeurs critiques.
Entre deux valeurs critiques comment varie la taille du carré?

3.4

Lorsque l'on se situe à un certain θ comment déterminer la prochaine valeur critique.

3.5

Déduire un algorithme, donner sa complexité.

3.6

Et si on voulait trouver le rectangle englobant d'aire minimale?

Examen DEA Images et Vision 2003-2004 : géométrie algorithmique
3h — 9 mars 2004

1 Recherche de plus proche voisin

Étant donné un ensemble $\mathcal{S}$ de n points dans le plan et un point du plan p on cherche à déterminer $NN_{\mathcal{S}}(p)$ le point de $\mathcal{S}$ le plus proche de p. On note $DT(\mathcal{S})$ la triangulation de Delaunay de $\mathcal{S}$.

w est initialisée avec un sommet quelconque de $DT(\mathcal{S})$.

```
N = un sommet de DT(S);
dmin = |pN|;
Faire {
        v = N;
        Pour chaque voisin w de v dans DT(S) {
                si ( dmin > |pw| ) {
                dmin = |pw|;
                N = w;
                }
        }
} tant que (N ≠ v);
return N;
```

1.1

Démontrer que ce programme calcule le plus proche voisin de v.

Correction : On regarde le cercle C de centre p passant par N, supposons que ce cercle contiennent des points de $\mathcal{S}$ alors en considérant les cercle tangents à C en N à l'intérieur de C on va trouver un cercle vide passant par N et un de ces points. N a donc un voisin plus près de p que N ce voisin aurait donc du être considéré dans la boucle et remplacer N. suivant.

1.2

Quelle est la complexité dans le cas le pire de cet algorithme? Donner un exemple réalisant cette pire complexité ($\forall n$).

Correction : $O(n)$. Exemple avec des points presque alignés.

1.3

Si on modifie l'algorithme comme suit:
Variante :

$N =$ un sommet de $DT(\mathcal{S})$;
$dmin = |pN|$;
Faire {
    $v = N$;
    Pour chaque voisin w de v dans $DT(\mathcal{S})$ {
        si ($dmin > |pw|$) {
        $dmin = |pw|$;
        $N = w$;
        break; // *Sortir de la boucle "Pour"*
        }
    }
} tant que ($N \neq v$);
return N;

Qu'apportent la variante?
Le nombre d'exécutions de l'instruction $N = w$ est-il modifié?
Dans quel sens?
Est-ce que l'on calcule bien toujours le plus proche voisin?
Commentaires?

Correction : Variante : la condition d'arrêt est la même, il faut qu'aucun des voisins du dernier candidat n'ait été plus proche et donc inséré dans Q. Par contre au lieu de prendre toujours le meilleur w, on va s'arrêter au premier qui marche, donc on exécutera plus souvent l'instruction $N = w$, mais on regardera moins de points. Au total on peut espérer gagner. C'est vrai pour des points uniformément répartis.

2 Triangulation optimisant les cercles circonscrits

Étant donné un ensemble de n points du plan $\mathcal{S}$ et une triangulation $\mathcal{T}(\mathcal{S})$ de cet ensemble de points, on définit $\mu(\mathcal{T}(\mathcal{S}))$ le maximum des rayons des cercles circonscrits aux triangles de $\mathcal{T}(\mathcal{S})$.

2.1

Trouver une triangulation qui minimise $\mu(\mathcal{T}(\mathcal{S}))$. Algorithme, complexité...

Correction : C'est Delaunay. La bascule de diagonale "Delaunay" optimise localement le rayon maximum. Donc Delaunay optimise cette propriété.

2.2

Trouver une triangulation qui maximise $\mu(\mathcal{T}(\mathcal{S}))$ **dans le cas particulier** où les points de $\mathcal{S}$ forment un polygone convexe. Algorithme, complexité...

Correction : Si on choisit 3 points de $\mathcal{S}$ on sait toujours faire une triangulation qui contient ce triangle. Donc le problème se réduit à trouver le triangle ayant le plus grand cercle. Un tel triangle est forcément défini par 3 points consécutifs sur le bord de l'enveloppe convexe (sinon le point intermédiaire et 2 des autres (bien choisis) définissent un cercle plus grands). Il suffit donc de calculer l'enveloppe convexe et d'essayer tous les triplets de points consécutifs. $O(n \log n)$ voire $O(n)$ si le convexe est connu. Dans le cas général on peut le faire trivialement en n^3.

3 Plus petit carré englobant.

Étant donné un ensemble de n points du plan $\mathcal{S}$ on cherche le plus petit carré contenant ces n points. On supposera que l'on n'a pas trois points alignés.

3.1

Pour commencer on cherche un carré dont les cotés sont parallèles à une certaines direction (repéré par son angle θ avec l'axe des x). Donner un algorithme et sa complexité. En général, combien de points de $\mathcal{S}$ sont-ils sur le bord de ce carré.

Correction : Trivial recherche des min et max, dans les directions θ et $\theta + \frac{\pi}{2}$ $O(n)$. En général 3 points sur le bord du carré, éventuellement 2 si ils définissent sa diagonale, éventuellement plus si on a des points alignés dans les directions θ ou $\theta + \frac{\pi}{2}$ ou une égalité entre les "largeurs" (max-min) dans les directions θ et $\theta + \frac{\pi}{2}$.

3.2

A quel moments les points définissant le plus petit carré peuvent-ils changer lorsque θ augmente?

Correction : Le plus petit carré changent quand on se trouve dans une direction θ où on a des points alignés dans les directions θ ou $\theta + \frac{\pi}{2}$ ou une égalité entre les "largeurs" (max-min) dans les directions θ et $\theta + \frac{\pi}{2}$.

3.3

On appelle ces valeurs de θ valeurs critiques.

Entre deux valeurs critiques comment varie la taille du carré?

Correction : Si p_1 et p_3 sont le minimum et le maximum dans la direction $\theta + \frac{\pi}{2}$ et définissent le carré (sinon on prend p_0 et p_2 dans la direction θ) et en notant ϕ_{13} l'angle de p_1p_3 avec la direction x la taille du carré est $|p_1p_3|sin(|\phi_{13} - \theta|)$.

3.4

Lorsque l'on se situe à un certain θ comment déterminer la prochaine valeur critique.

Correction : Soit c'est un p_i qui change parce que le point suivant sur l'enveloppe convexe arrive sur le bord du carré. Soit on a $|p_1p_3|sin(|\phi_{13} - \theta|) = |p_0p_2|sin(|\phi_{02} - \theta - \frac{\pi}{2}|)$. On a donc 5 possibilités, il suffit de sélectionner la première des 5.

3.5

Déduire un algorithme, donner sa complexité.

Correction : On calcule l'enveloppe convexe. On calcule les points p_0, p_1, p_2 et p_3 pour $\theta = 0$. On calcule les points critiques de proche en proche. On sélectionne le point critique ayant le plus petit carré. (le minimum se produit nécessairement en un point critique). Complexité $O(n \log n)$ pour l'enveloppe convexe, $O(n)$ pour la suite.

3.6

Et si on voulait trouver le rectangle englobant d'aire minimale?

Correction : Dans notre balayage en θ on visite tous les rectangles englobant, au lieu de maintenir le max des deux cotes (pour le carré) il suffirait de regarder l'aire et de repérer le plus petit. Les points où $|p_1p_3|sin(|\phi_{13}-\theta|) = |p_0p_2|sin(|\phi_{02}-\theta-\frac{\pi}{2}|)$ ne sont plus critiques. Attention, l'aire c'est $|p_1p_3|sin(|\phi_{13}-\theta|)|p_0p_2|sin(|\phi_{02}-\theta-\frac{\pi}{2}|)$ et il n'ait pas clair que cette fonction n'ai pas de minimum entre deux points critiques.