Plongement isométrique dans le plan réctilinéaire en temps optimal $O(n^2)$

Nicolas Catusse, Victor Chepoi, Yann Vaxès

Université de la Méditerranée

Faculté des Sciences de Luminy

Plongement dans le plan rectilinéaire

Plongement

Un espace métrique (X, d) est plongeable isométriquement dans un espace métrique (Y, d') si il existe une application $\varphi : X \to Y$ tel que $d'(\varphi(x), \varphi(y)) = d(x, y)$ pour tout $x, y \in X$.

Problème

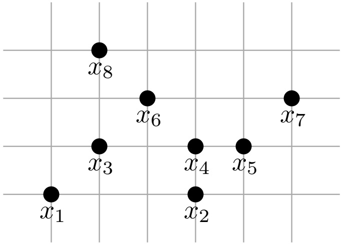
Étant donné un espace métrique (X, d), décider si cet espace est plongeable isométriquement dans $\mathbb{R}^2$ avec la métrique l_1 ou l_∞.

Formulation du problème

Problème

Étant donné un espace métrique (X, d), décider si cet espace est plongeable isométriquement dans $\mathbb{R}^2$ avec la métrique l_1 (ou l_∞).

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	0	3	2	4	5	4	7	4
x_2	3	0	3	1	2	3	4	5
x_3	2	3	0	2	3	2	5	2
x_4	4	1	2	0	1	2	3	4
x_5	5	2	3	1	0	3	3	5
x_6	4	3	2	2	3	0	3	2
x_7	7	4	5	3	2	3	0	5
x_8	4	5	2	4	5	2	5	0



Etat de l'art

- **Menger et Schönberg (1928) :** On peut décider en temps polynomial si (X, d) peut être plongé dans $\mathbb{R}^k$ avec la métrique Euclidienne.
- **Frechet :** Tout espace métrique peut être plongé dans $\mathbb{R}^k$ avec la norme l_∞.
- **Bandelt, Chepoi (1998) :** Un espace métrique peut être plongé dans le plan l_1 ssi n'importe quel sous espace d'au plus 6 points peut l'être.
- **Edmonds (2008) :** Algorithme de décision en $O(n^2 \log^2 n)$.
- **Eppstein (2009) :** Algorithme de décision en $O(n^2)$.

Définition

Enveloppe injective

Pour tout espace métrique (X, d) il existe un plus petit espace injectif $T(X)$ dans lequel on peut plonger (X, d).

FIGURE 1: Enveloppe injective sur 3 et 4 points

Enveloppe injective sur 5 points

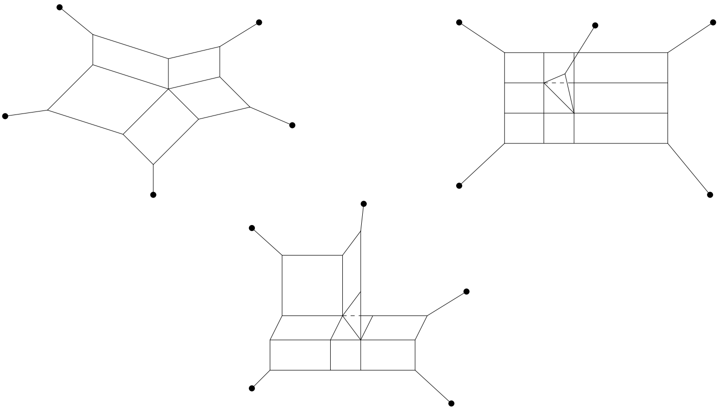
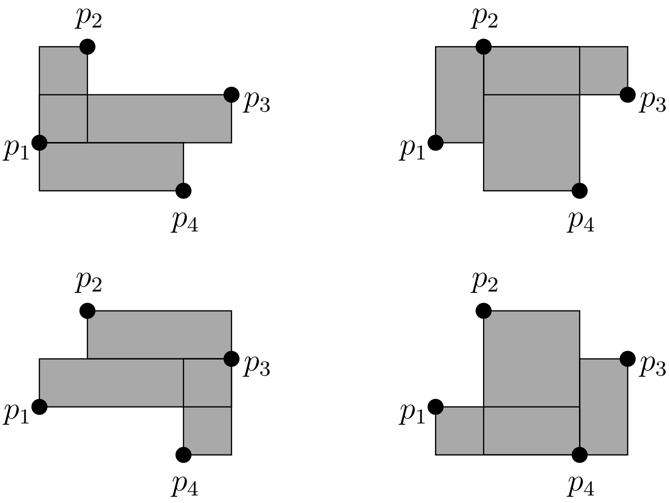


FIGURE 2: Enveloppe injective sur 5 points

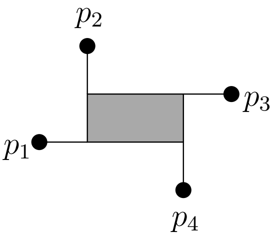
Enveloppe injective dans $(\mathbb{R}^2, l_1)$

Avec la métrique l_1 $T(X) = \cap_{i=1}^{n}(\cup_{j=1}^{n} I(x_i, x_j))$

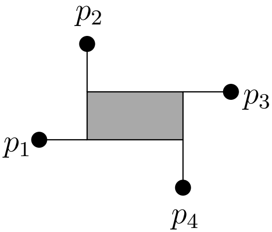


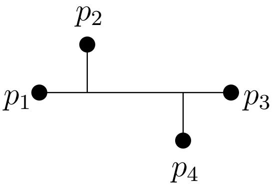
Enveloppe injective dans $(\mathbb{R}^2, l_1)$

Avec la métrique l_1 $T(X) = \cap_{i=1}^{n}(\cup_{j=1}^{n} I(x_i, x_j))$



Enveloppe injective dans $(\mathbb{R}^2, l_1)$





Algorithme

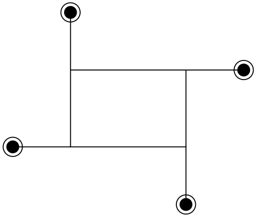
Entrée : Un espace métrique (X, d) de n points
Sortie : Un plongement isométrique φ de (X, d) dans $(\mathbb{R}^2, d_1)$

Résumé de l'algorithme

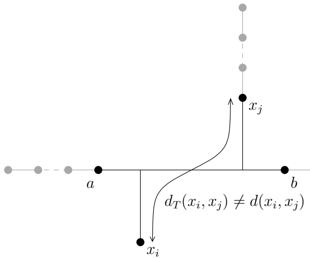
- Trouver un quadruplet de point P° dont l'enveloppe injective $T(X)$ contient un rectangle $R(P^\circ)$.
- Localiser les points de X par rapport à $R(P^\circ)$ et trouver un nouveau quadruplet P pour lequel on a des régions vides.
- Pour tous les plongements possibles de P, essayer d'étendre ce plongement à l'espace métrique entier.

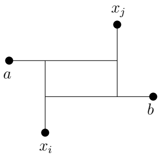
Algorithme

Étape 1. Trouver un quadruplet P° de X pour lequel l'enveloppe injective contient un rectangle.



Algorithme

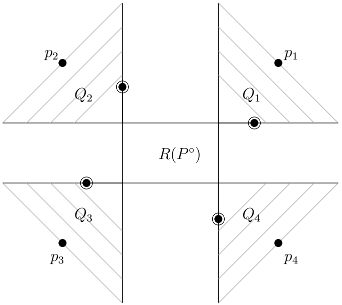




Si $T(X)$ a plus de 4 feuilles, alors retourner "non".

Algorithme

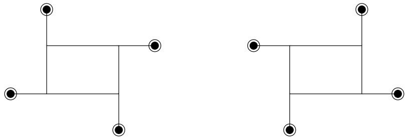
Étape 2. Prendre un plongement quelconque de $T(P^\circ)$ et localiser chaque terminal de $X \setminus P^\circ$ dans l'une des neuf régions du plan définies par $R(P^\circ)$.



Construire P en choisissant dans les régions Q_1, Q_2, Q_3, Q_4 le terminal le plus éloigné de $R(P^\circ)$.

Algorithme

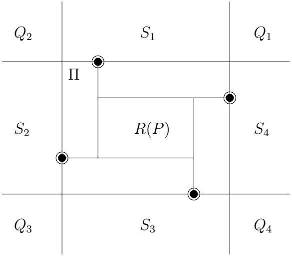
Étape 3. Plonger $T(P)$ dans le plan l_1 de toutes les façons possibles. Essayer d'étendre ces plongements à un plongement isométrique de (X, d). Si pour chacun d'eux la réponse est "non", alors retourner la réponse "non", sinon retourner un des plongements obtenus.



Algorithme

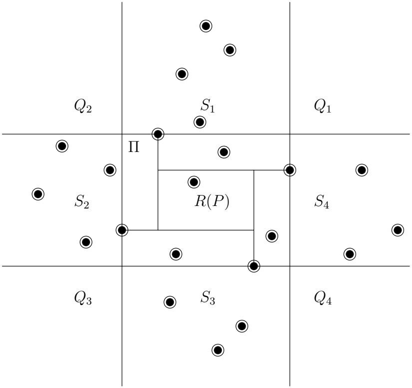
Détails de l'étape 3 :

Prendre un plongement φ_0 de $T(P)$, pour chaque terminal u de $X \setminus P$ déterminer dans laquelle des neuf régions définis par le rectangle Π sera localisé u dans n'importe quel plongement isométrique qui étend φ_0.



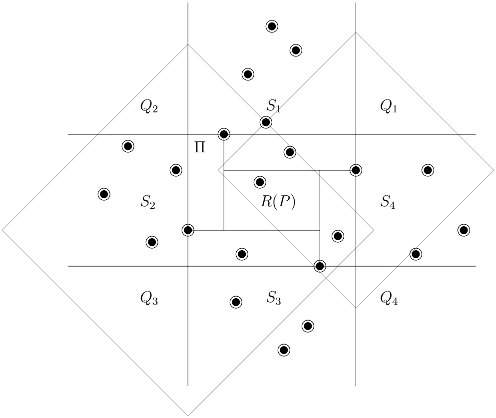
Algorithme

Localiser les terminaux associés au rectangle Π et aux quatre demi-bandes S_1, S_2, S_3, S_4.



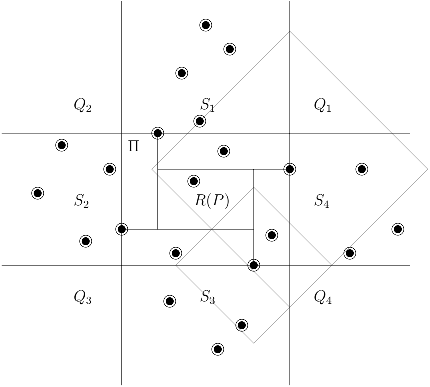
Algorithme

Les terminaux dans les demi-bandes sont fixés de façon unique.



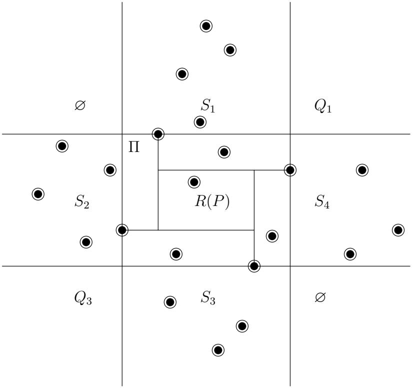
Algorithme

Les terminaux dans les quadrants sont localisés sur un segment.



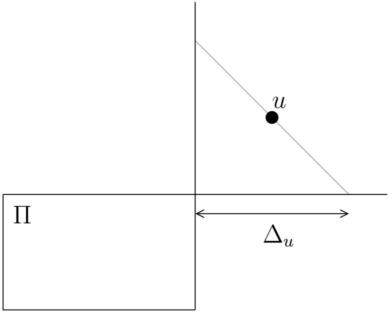
Algorithme

Le choix du quadruplet P permet d'obtenir deux quadrants opposés vides.



Algorithme

Soit Δ_u la distance l_1 entre un point u appartenant à un quadrant et le sommet de Π définissant ce quadrant.



Algorithme

Définir les ensembles de terminaux X_1 et X_3 associés aux quadrants Q_1 et Q_3, construire les graphes $G_1 = (X_1, E_1)$ et $G_3 = (X_3, E_3)$.

Algorithme

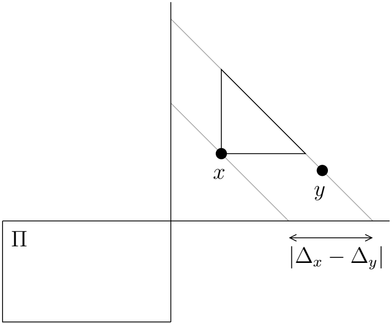
Définir les ensembles de terminaux X_1 et X_3 associés aux quadrants Q_1 et Q_3, construire les graphes $G_1 = (X_1, E_1)$ et $G_3 = (X_3, E_3)$.

Deux terminaux x, y sont adjacents dans G_i si la distance entre eux n'est pas réalisée par la distance entre leurs segments de niveaux : $|\Delta_x - \Delta_y| < d(x, y)$

Algorithme

Définir les ensembles de terminaux X_1 et X_3 associés aux quadrants Q_1 et Q_3, construire les graphes $G_1 = (X_1, E_1)$ et $G_3 = (X_3, E_3)$.

Deux terminaux x, y sont adjacents dans G_i si la distance entre eux n'est pas réalisée par la distance entre leurs segments de niveaux : $|\Delta_x - \Delta_y| < d(x, y)$

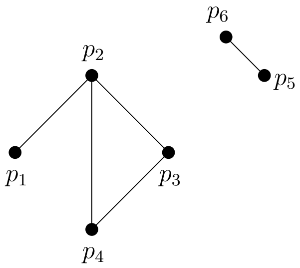


Algorithme

Définir les ensembles de terminaux X_1 et X_3 associés aux quadrants Q_1 et Q_3, construire les graphes $G_1 = (X_1, E_1)$ et $G_3 = (X_3, E_3)$.

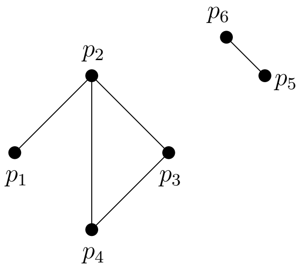
Deux terminaux x, y sont adjacents dans G_i si la distance entre eux n'est pas réalisée par la distance entre leurs segments de niveaux : $|\Delta_x - \Delta_y| < d(x, y)$

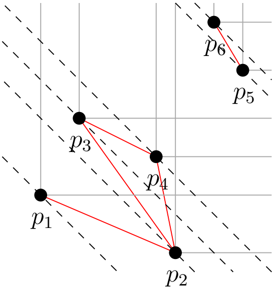
$$
\begin{aligned}
|\Delta_{p_1} - \Delta_{p_2}| &< d(p_1, p_2) \\
|\Delta_{p_1} - \Delta_{p_3}| &= d(p_1, p_2) \\
|\Delta_{p_1} - \Delta_{p_4}| &= d(p_1, p_2) \\
|\Delta_{p_2} - \Delta_{p_3}| &< d(p_1, p_2) \\
|\Delta_{p_2} - \Delta_{p_4}| &< d(p_1, p_2) \\
|\Delta_{p_3} - \Delta_{p_4}| &< d(p_1, p_2) \\
&\dots
\end{aligned}
$$



Algorithme

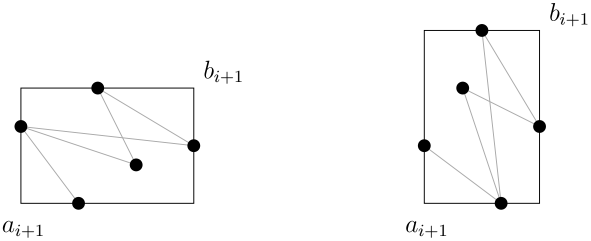
Trouver un plongement isométrique de chaque composante C_i de G_1 qui ne contient pas encore de terminaux fixés, ces terminaux sont donc localisés dans leur segment de niveau. Faire la même chose pour G_3.





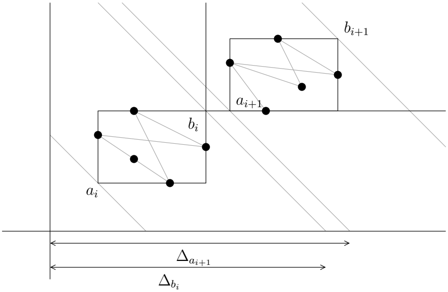
Algorithme

Les composantes sont fixées à une symétrie près.



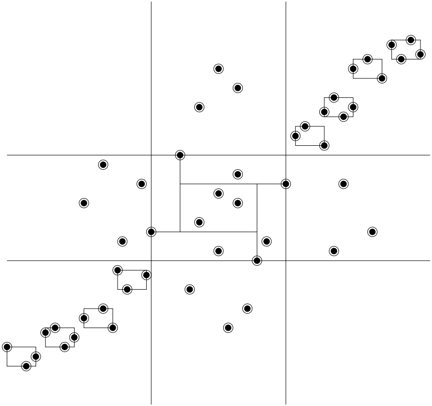
Algorithme

Tester si les composantes libres $C_{j+1}, \ldots, C_k$ de G_1 satisfont la condition $\Delta_{b_i} \leq \Delta_{a_{i+1}}$. Si ce n'est pas le cas, alors renvoyer la réponse "non", sinon localiser les composantes et fixer la position de tous les terminaux de X_1. Faire la même chose pour les composantes libres de G_3.



Algorithme

Vérifier si le résultat du plongement de X qui étend φ_0 est un plongement isométrique de (X, d). Si “oui”, alors le renvoyer comme plongement isométrique, sinon il n'existe pas de plongement isométrique de (X, d) qui étend le plongement φ_0.



Algorithme

Entrée : Un espace métrique (X, d) de n points
Sortie : Un plongement isométrique φ de (X, d) dans $(\mathbb{R}^2, d_1)$

Résumé de l'algorithme

- Trouver un quadruplet de point P° dont l'enveloppe injective $T(X)$ contient un rectangle $R(P^\circ)$. $O(n^2)$
- Localiser les points de X par rapport à $R(P^\circ)$ et trouver un nouveau quadruplet P pour lequel on a des régions vides. $O(n)$
- Pour tous les plongements possibles de P, essayer d'étendre ce plongement à l'espace métrique entier. $O(n^2)$
  - Localiser les points fixés par le quadruplet P. $O(n)$
  - Construire un graphe avec les points non fixés. $O(n^2)$
  - Plonger une par une les composantes connexes du graphe. $O(n)$
  - Vérifier que le plongement obtenu est isométrique. $O(n^2)$

Question

Décider si un espace métrique peut être plongé dans $\mathbb{R}^3$ avec la métrique l_∞ est un NP-difficile (Edmonds 2008). Est-il possible de décider en temps polynomial le plongement dans $\mathbb{R}^3$ avec la métrique l_1 ?