

Conception et Pratique de l'Algorithmique

<http://www-apr.lip6.fr/~buixuan/cpa2019>

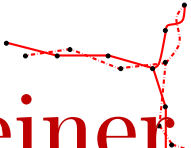
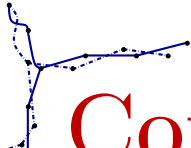
Binh-Minh Bui-Xuan



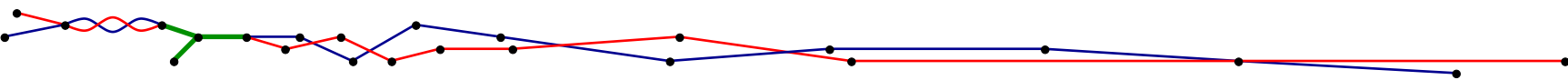
**SORBONNE
UNIVERSITÉ**
CRÉATEURS DE FUTURS
DEPUIS 1257

PARIS, Avril 2020



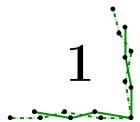
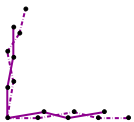


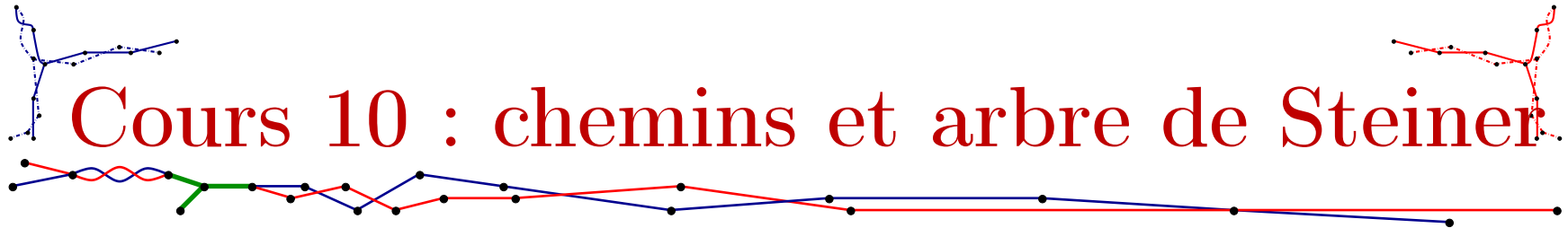
Cours 10 : chemins et arbre de Steiner



RAPPEL COURS + TME 9 :

- arbre couvrant : classique vs. arbre de Steiner
- problème ARBRECOUVRANTMIN : glouton est optimal
- complexité en $O(m \log n)$ théorique ; en $O(m \log n + n^2)$ en tme





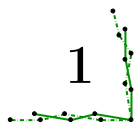
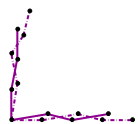
Cours 10 : chemins et arbre de Steiner

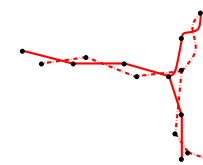
RAPPEL COURS + TME 9 :

- arbre couvrant : classique vs. arbre de Steiner
- problème ARBRECOUVRANTMIN : glouton est optimal
- complexité en $O(m \log n)$ théorique ; en $O(m \log n + n^2)$ en tme

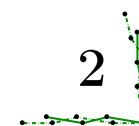
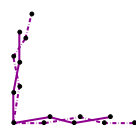
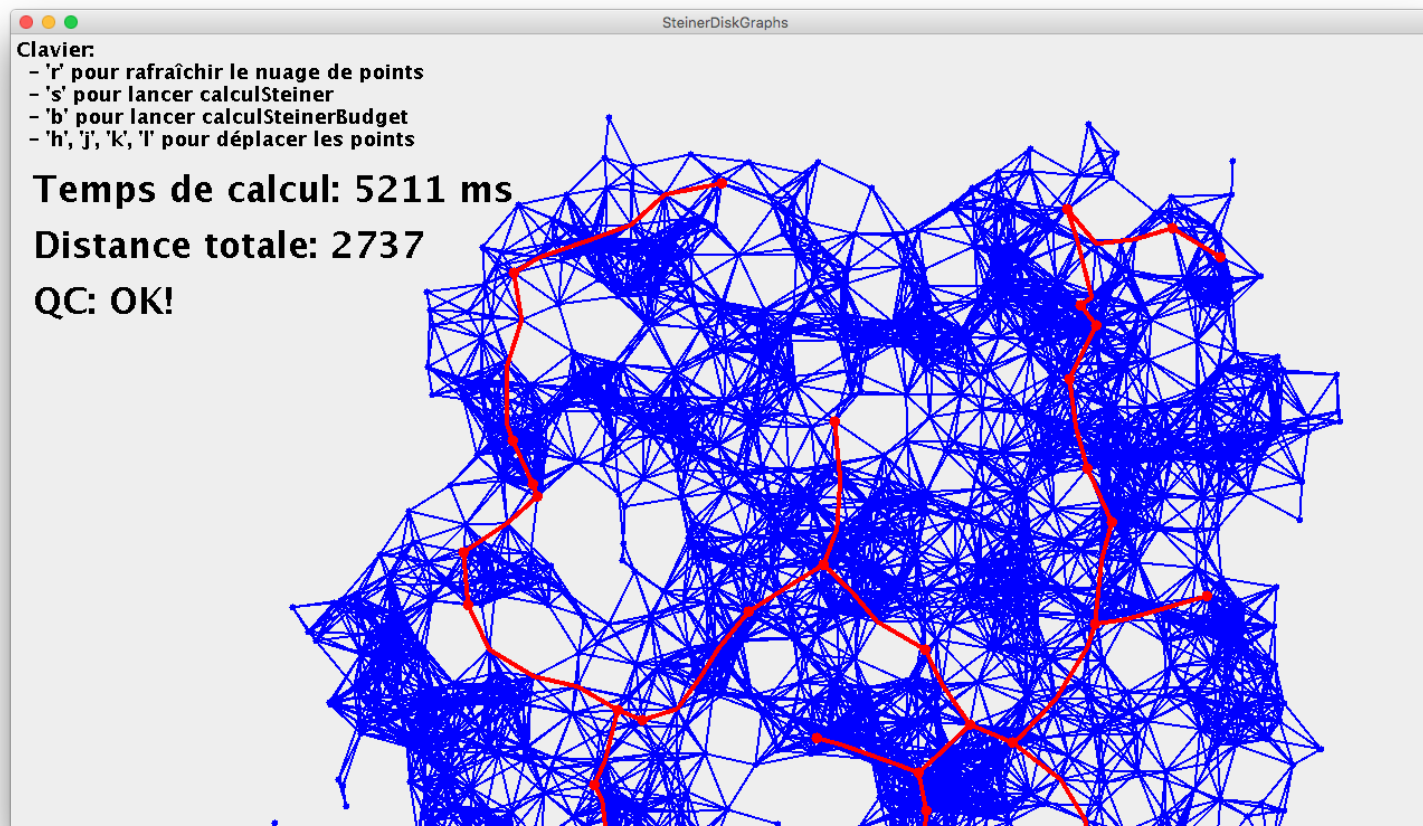
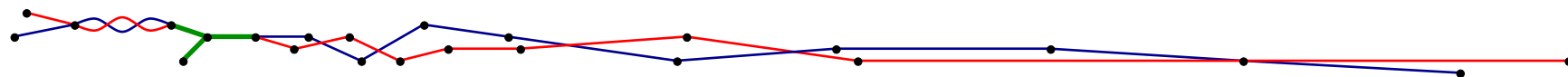
AUJOURD'HUI :

- problème ARBRESTEINER : NP-difficile
- heuristique du glouton avec réduction à ARBRECOUVRANTMIN
- problème ALLTOALLPATHS : en $O(n^3)$ par prog. dynamique
- techniques : glouton, programmation dynamique
- algorithmes : Kruskal, Floyd-Warshall

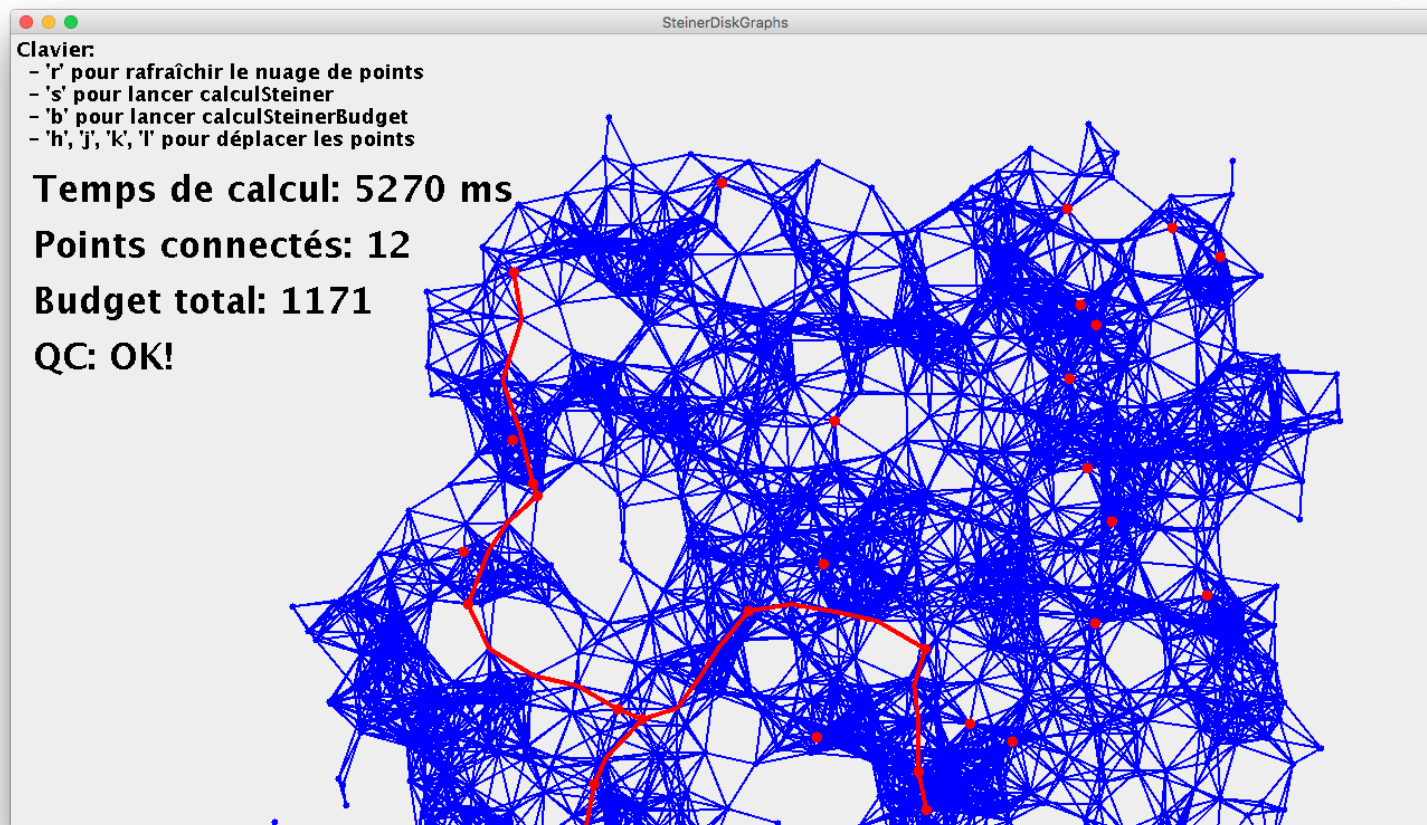


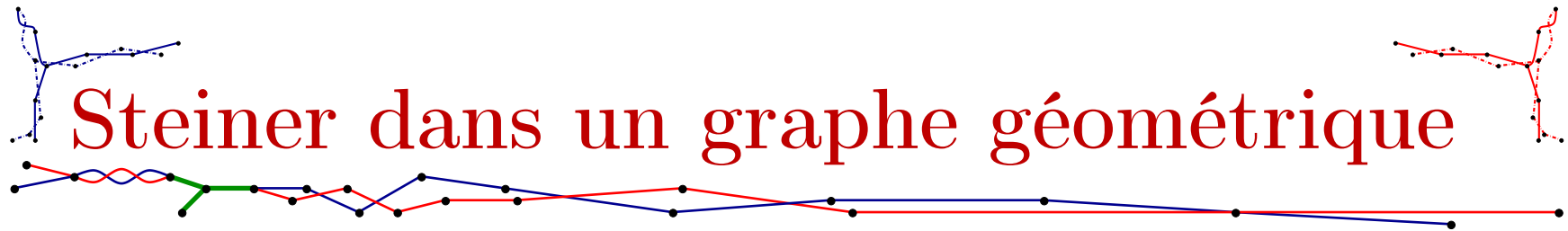


Réseaux connexe :



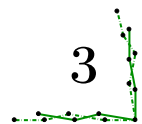
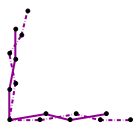
Réseaux connexe : budget

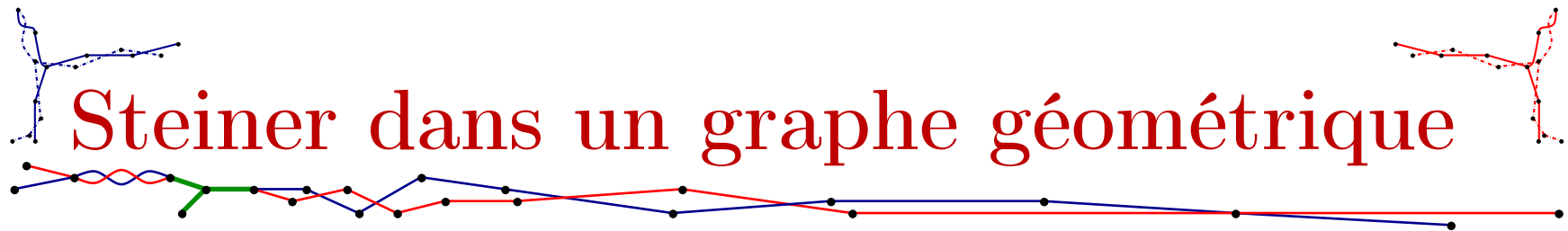




IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold
un réel ; hitPoints sous liste de Points

OUT : arbre Tree, de poids total minimum, couvrant tous les points
de hitPoints, dont tout sommet est dans Points, dont toute arête
est de distance inférieur au seuil edgeThreshold.



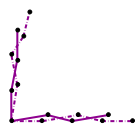


IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold
un réel ; hitPoints sous liste de Points

OUT : arbre Tree, de poids total minimum, couvrant tous les points
de hitPoints, dont tout sommet est dans Points, dont toute arête
est de distance inférieur au seuil edgeThreshold.

N.B. : Structure de graphe ?

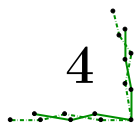
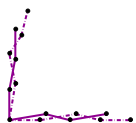
THÉORÈME : ARBRESTEINER *est NP-difficile sur les graphes géométriques*

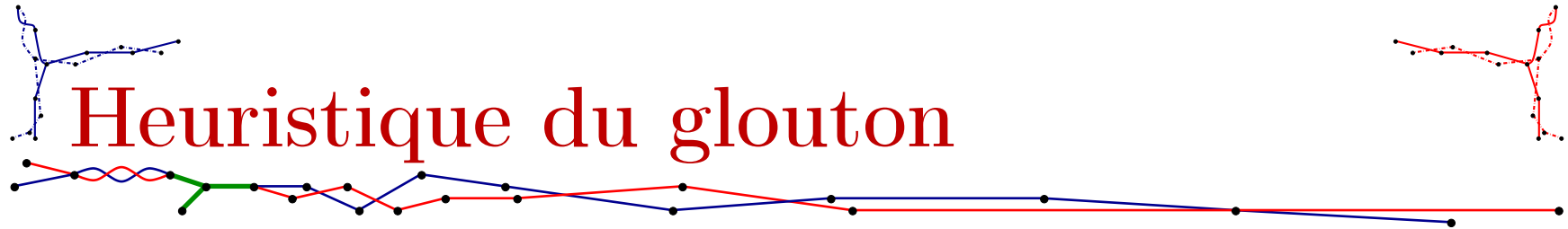




PRINCIPE : construire un graphe complet K , dont les sommets sont les points, dont toute arête uv (dans K) a pour poids la distance d'un plus court chemin entre u et v dans G . Principe de l'heuristique :

$G \rightarrow$ Steiner dans $K \leftrightarrow$ Kruskal dans $K \leftrightarrow$ sous-graphe H dans G
 \rightarrow Kruskal dans H



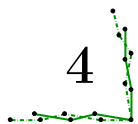
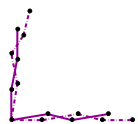


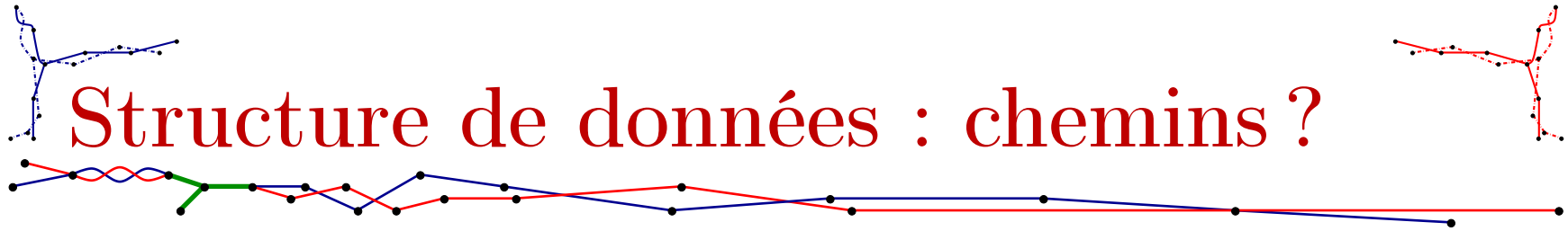
Heuristique du glouton

PRINCIPE : construire un graphe complet K , dont les sommets sont hitPoints, dont toute arête uv (dans K) a pour poids la distance d'un plus court chemin entre u et v dans G . Principe de l'heuristique :

$G \rightarrow$ Steiner dans $K \leftrightarrow$ Kruskal dans $K \leftrightarrow$ sous-graphe H dans G
 \rightarrow Kruskal dans H

EXERCICE : Pseudo-code ?





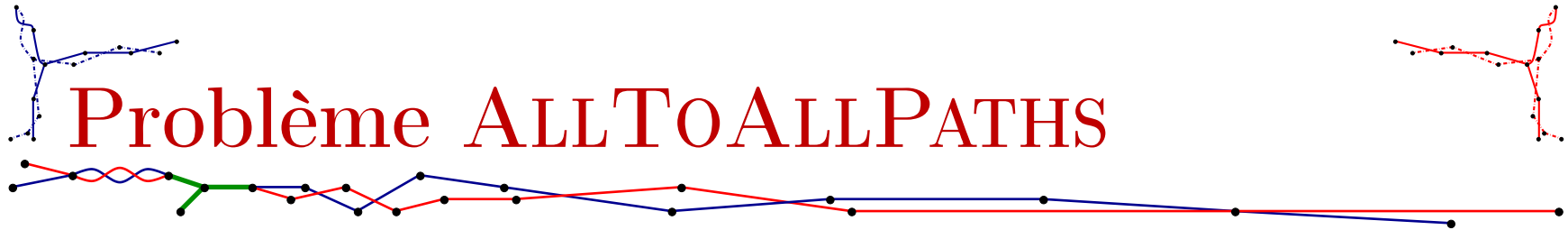
Chemin 1-to-1 : liste chaînée, tas, ...

Chemin all-to-all : matrice de distance + accès, p.e.

- $\text{dist}[i][j]$: distance dans G entre i et j
- $\text{paths}[i][j]$: le sommet k , successeur de i dans un plus court chemin de i à j

QUESTION : matrice d'accès \rightarrow liste chaînée ?





IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold un réel. Soit G le graphe géométrique représenté par la liste Points et le réel edgeThreshold

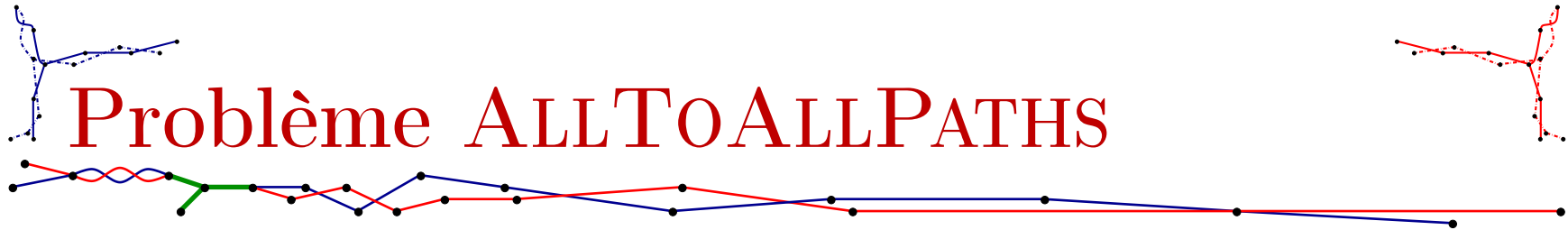
OUT : matrices de distance et d'accès représentant les plus courts chemins dans G entre toute paire de points dans Points

EXERCICE : algorithmes en $O(n^4)$?

PRINCIPE : soit M^p la matrice où $M^p(i, j)$ est la distance entre i et j des chemins utilisant au plus p sommets dans G . Question :

– procédure calculant $M^p \rightarrow M^{p+1}$?

– procédure calculant M^2 ?



IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold un réel. Soit G le graphe géométrique représenté par la liste Points et le réel edgeThreshold

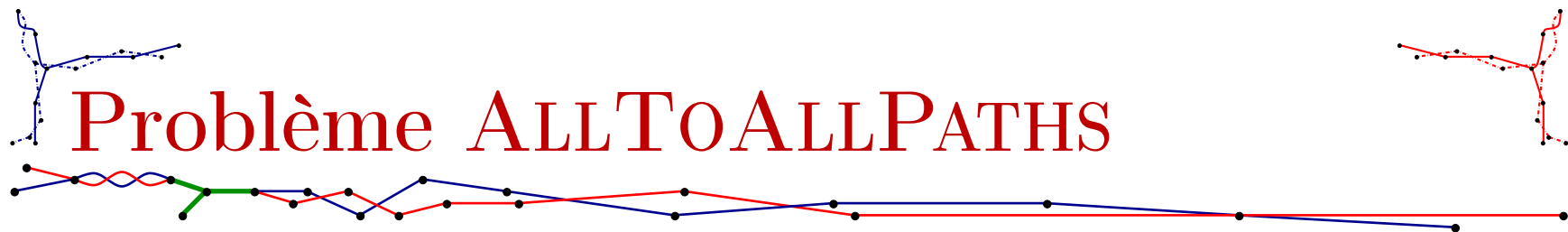
OUT : matrices de distance et d'accès représentant les plus courts chemins dans G entre toute paire de points dans Points

EXERCICE : algorithmes en $O(n^4)$?

PRINCIPE : soit M^p la matrice où $M^p(i, j)$ est la distance entre i et j des chemins utilisant au plus p sommets dans G . On a :

$$- M^{p+1}(i, j) = \min_k M^p(i, k) + M^2(k, j)$$

$$- M^2(i, j) = (d(i, j) < \text{edgeThreshold}) ? d(i, j) : \infty$$



IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold un réel. Soit G le graphe géométrique représenté par la liste Points et le réel edgeThreshold

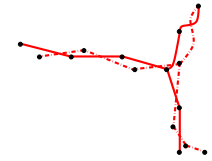
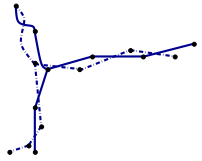
OUT : matrices de distance et d'accès représentant les plus courts chemins dans G entre toute paire de points dans Points

EXERCICE : complexité de l'algorithme suivant ?

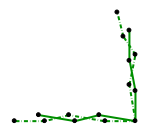
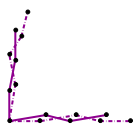
PRINCIPE : $M^p(i, j)$: distance entre i et j des chemins passant uniquement par les p premiers sommets de G , i.e. :

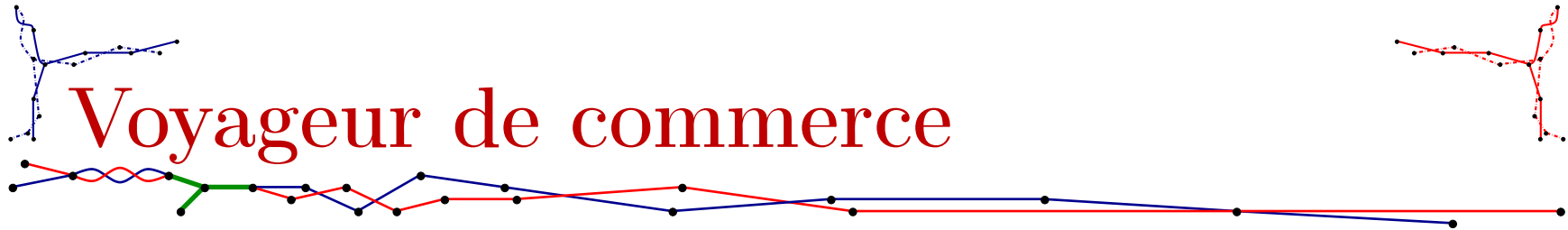
$$- M^{p+1}(i, j) = \min \left(M^p(i, j), M^p(i, p+1) + M^p(p+1, j) \right)$$

$$- M^0(i, j) = (d(i, j) < \text{edgeThreshold}) ? d(i, j) : \infty$$



Rappel : cette technique a un nom :
Programmation dynamique



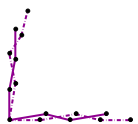


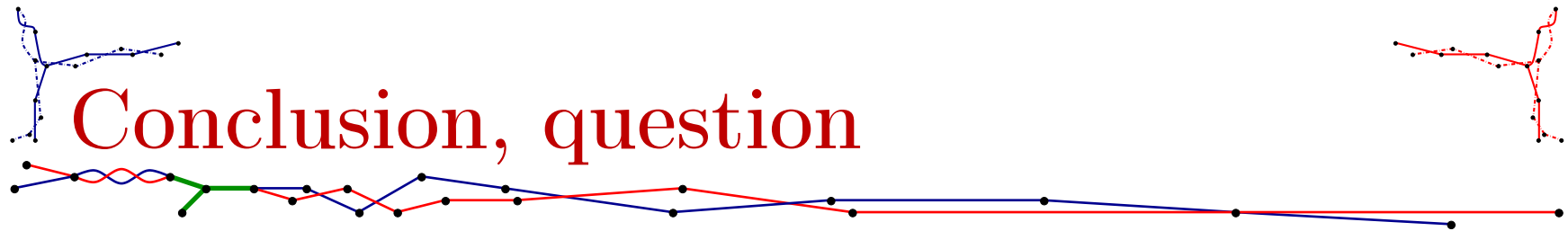
IN : Points, une liste de coordonnées de points en 2D ; edgeThreshold un réel. Soit G le graphe géométrique représenté par la liste Points et le réel edgeThreshold

OUT : permutation Circuit des points de Points, de distance consecutive totale minimum, dont toute paire consecutive de points a pour distance inférieure à edgeThreshold

QUESTION :

- complexité de l'algorithme brute-force ?
- formule de récursion pour programmation dynamique ?





CONCLUSION :

- ARBRESTEINER : heuristique par ARBRECOUVRANTMIN
- ALLTOALLPATHS : récursion de Floyd-Warshall

QUESTION :

- implantation ? (voir TME)

