INFO601 : algorithmique et graphes
TD 4 : arbre couvrant de poids minimal

Pierre Hyvernat, Gérald Cavallini
Laboratoire de mathématiques de l'université de Savoie
bâtiment Chablais, bureau 17, poste : 94 22
email : Pierre.Hyvernat@univ-smb.fr
www : http://www.lama.univ-smb.fr/~hyvernat/

Question 1. Comment peut-on calculer un arbre couvrant d'un graphe connexe non pondéré ? Quelle est la complexité au pire cas du calcul d'un tel arbre ?

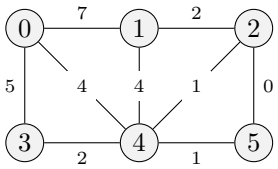
Exercice 1 : algorithme de Prim

L'algorithme de Prim permet de calculer un arbre couvrant de poids minimal en modifiant le parcours basique :

1- En plus du tableau `Pred`, l'algorithme sauvegarde également, pour chaque sommet `v`, le poids de l'arête entre `v` et `Pred[v]`. (Lorsque `Pred[v]` n'est pas défini, `W[v]` vaut $+\infty$.)
2- Au début de la boucle, on choisit le sommet s dans `ATraiter` qui a la petite valeur de `W[`s`]`.
3- Lorsqu'on regarde les voisins v de s, on mets à jour les tableau `Pred` et `D` seulement si le nouveau poids est plus petit :

```
for (v, w) in G[s]:      # w est le poids de l'arête s -> v
    if not Vu[v] and w < W[v]:
        ...                  # on ajoute v dans ATraiter
        Pred[v] = s
        W[v] = w
```

Question 1. Faites tourner l'algorithme à la main sur le graphe suivant, en partant du sommet $s_0 = 0$, puis du sommet $s_0 = 5$.



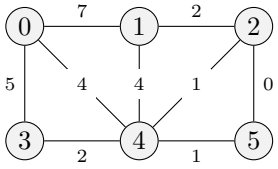
Dessinez les arbres couvrants obtenus. Sont ils identiques ? Leurs poids sont ils les mêmes ?

Question 2. Comment peut-on trouver le sommet s tels que `W[s]` est minimal ?
Quelle est la complexité de cette rercherche ?

Exercice 2 : algorithme de Kruskal

Initialement, l'algorithme de Kruskal considère tous les sommets comme autant de sommets isolés. Il prend ensuite l'arête de poids minimale parmis toutes les arêtes possibles. Si cette arête produit une boucle, on la supprime. Sinon, on l'ajoute dans la forêt courante.

Question 1. Faites tourner l'algorithme à la main sur le graphe suivant.



Vous commencerez donc avec les 6 sommets isolés, auquels vous ajouterez des arêtes.
L'arbre obtenu est il unique ? Quel est son poids ?

Question 2.
- Comment peut-on chercher l'arête de poids minimal rapidement ?
- Comment peut-on détecter rapidement si l'ajout d'une arête forme un cycle ?

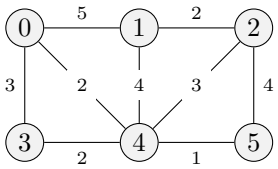
Exercice 3 : application au voyageur de commerce

Le problème du voyageur de commerce consiste a trouver un circuit *de poids minimal* passant exactement une fois par tous les sommets d'un graphe pondéré non orienté. Il s'agit d'un problème NP-complet, et les seuls algorithmes connus sont donc de complexité exponentielle.

Il existe algorithmes approchés efficaces dans le cas des graphes qui vérifient l'inégalité triangulaire : le poids de l'arête entre v_1 à v_2 est toujours plus petite que la somme des poids entre v_1 et v et celui entre v et v_2. (Autrement dit, il est toujours plus rapide d'aller directement à sa destination que de faire une escale supplémentaire.)

Question 1. Donnez un exemple de graphe pondéré qui ne satisfait pas l'inégalité triangulaire.

Question 2. On considère le graphe pondéré suivant, qui vérifie l'inégalité triangulaire :



Donnez un arbre couvrant de poids minimal. Quel est son poids ?

Question 3. Si T est l'arbre couvrant de poids minimal trouvé précédemment, on "double" chacune des arêtes en remplaçant chaque

On peut parcourir ce multigraphe "T-doublé" de manière à passer exactement une fois par toutes les arêtes et à revenir à son sommet de départ.
- Expliquez pourquoi.
- Quel est le poids du circuit parcouru ?
- Proposez un algorithme efficace pour obtenir ce parcours.

Question 4. Le circuit obtenu à la question précédente n'est pas très bon car il repasse plusieurs fois par certains sommets. Améliorez le en prenant des racourcis qui évitent de repasser plusieurs fois par le même sommet. (Par l'inégalité triangulaire, on obtiendra un chemin plus court.)

Quel est le poids du circuit final ?

Question 5. On s'intéresse à l'erreur faite par l'algorithme précédent (doublage des arêtes d'un arbre couvrant de poids minimal + simplification du chemin).

Justifiez les affirmations suivantes.
- "Le poids d'un circuit optimal pour le voyageur de commerce est forcément plus petit que le poids du circuit trouvé par l'algorithme précédent."
- "Le poids d'un circuit optimal pour le voyageur de commerce est forcément plus grand que le poids d'un arbre couvrant de poids minimal."
- "Le poids du circuit obtenu par l'algorithme précédent est forcément plus petit que le double de l'optimal."