

Objectifs

- À la fin de cette séance, vous devriez être capable de :
- reconnaître un algorithme glouton
 - déterminer s'il est ou non optimal
 - calculer sa complexité

Nota Bene. Commencez par terminer les exercices du TD précédent si nécessaire, ils sont plus abordables que le problème présenté ici.

Exercice 1 : Algorithme de Kruskal

Rappels On se place dans un graphe non orienté dont les arêtes sont pondérées (autrement dit on dispose d'une fonction $poids : R \rightarrow \mathbb{R}$).

Un **arbre** est un graphe connexe et sans cycle. On dispose de diverses caractérisations équivalentes à cette définition, en termes de nombre d'arêtes et/ou de graphe « connexe minimal » ou « acyclique maximal » (voir le cours si besoin).

On appelle **forêt** un graphe sans cycle, autrement dit un graphe dont chaque composante connexe est un arbre sur l'ensemble de ses sommets.

On appelle **arbre couvrant** de G un sous-graphe de G qui est un arbre, et dont l'ensemble des sommets est X . L'objectif de ce TD est d'étudier un algorithme de construction d'un **arbre couvrant de poids minimum** (ACPM) de G , c'est-à-dire un arbre couvrant de G dont la somme des poids des arêtes soit minimal.

L'algorithme de Kruskal

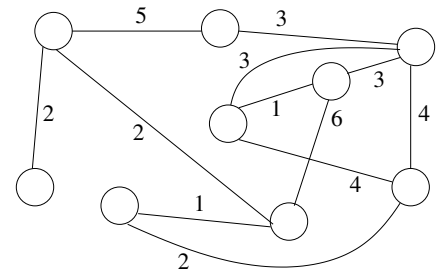
Voici le principe général de l'algorithme de Kruskal :

Initialiser le graphe Sol à (X, \emptyset) .

En parcourant les arêtes (x, y) de G par poids croissant :

- Si x et y sont dans des composantes connexes différentes alors
Ajouter (x, y) au graphe Sol

1. Faire tourner cet algorithme sur le graphe ci-contre.
2. Déterminer un critère d'arrêt efficace pour cet algorithme.
3. Si vous avez déjà fait le TD2 cette semaine : comment pouvez-vous utiliser la structure de données **Union-Find** pour répondre efficacement à la question : x et y sont-ils dans la même composante connexe ?
Écrire un algorithme détaillé utilisant notamment cette structure.



Correction de l'algorithme

1. Que contient Sol durant l'exécution de l'algorithme ?
2. Démontrer qu'à toute étape de l'algorithme, Sol est un sous-graphe d'un arbre couvrant de poids minimal.

Complexité Pour simplifier les calculs, on considère dans cette partie que le test « x et y sont dans des composantes connexes différentes » a une complexité en $\mathcal{O}(1)$ (grâce à Union-Find vue en TD2).

1. Évaluer le coût des différentes étapes de l'algorithme de Kruskal, en n'oubliant pas les étapes « préliminaires » qui pourraient être coûteuses.
2. En déduire la complexité de cet algorithme.

Membres du groupe et responsabilités

	Appliquer des méthodes algorithmiques connues (algorithmes gloutons, diviser pour régner)	Gérer le temps
	Évaluer l'efficacité des algorithmes	Réguler les prises de parole
	Rechercher des cas particuliers Déterminer des invariants	S'assurer que tous comprennent et sont en accord avec le rendu
	Représenter schématiquement les structures de données et leur traitement	Rédiger le rendu

Exercice 1 : Algorithme de Kruskal

1. Écrire l'algorithme de Kruskal détaillé pour résoudre le problème de l'arbre couvrant.
2. Que contient la solution en construction *Sol* durant l'exécution de l'algorithme ?