Devoir maison ALGO1

1 Algorithmes gloutons et matroïdes

On a vu (en cours) que l'algorithme de Kruskal fournit une solution au problème de recherche d'un arbre couvrant de poids minimal. L'objectif de cette section est de caractériser les problèmes pour lesquels une approche gloutonne permet d'obtenir une solution optimale.

Par la suite, on notera $\mathcal{S} = (E, \mathcal{F})$ la donnée d'un ensemble fini E et d'une collection $\mathcal{F}$ de sous-ensembles de E. On nommera `base` de $\mathcal{S}$ tout élément $F \in \mathcal{F}$ maximal pour l'inclusion. Un `matroïde` est un système $\mathcal{S} = (E, \mathcal{F})$ tel que $\emptyset \in \mathcal{F}$ et qui vérifie les propriétés suivantes :

`hérédité`	$\forall\, X \in \mathcal{F}, \forall\, Y \subset E, \quad Y \subset X \Rightarrow Y \in \mathcal{F}$
`augmentation`	$\forall\, X, Y \in \mathcal{F}, \quad \lvert X\rvert = \lvert Y\rvert + 1 \Rightarrow \exists x \in X \setminus Y, Y \cup \{x\} \in \mathcal{F}$

Soit $G = (S, A)$ un graphe, posons $E_G = A$ et soit $\mathcal{F}_G$ la collection des ensembles acycliques de G. Le système $\mathcal{S}_G = (E_G, \mathcal{F}_G)$ est appelé `matroïde graphique` associé à G.

Question 1 – Soit G un graphe. Montrer que le matroïde graphique associé à G est un matroïde.

`Lien avec les algorithmes gloutons`
On considère un matroïde $\mathcal{S} = (E, \mathcal{F})$ et une fonction de pondération $w : E \to \mathbb{R}^+$. On étend cette fonction aux éléments de $\mathcal{F}$ en posant

$$w(F) = \sum_{x \in F} w(x)$$

On souhaite trouver une base de $\mathcal{S}$ de poids maximum. On propose l'algorithme suivant :

```
GLOUTON (S, w)
1  Trier les éléments de E par ordre de poids décroissants w(e_1) ≥ w(e_2) ≥ ··· ≥ w(e_n)
2  F ← ∅
3  pour i = 1 à n faire
4      si F ∪ {e_i} ∈ F alors
5          F ← F ∪ {e_i}
6      fin si
7  fin pour
8  retourner F
```

Cet algorithme vérifie le théorème suivant.

Théorème. *Soit $\mathcal{S} = (E, \mathcal{F})$ un système vérifiant la propriété d'hérédité et tel que $\emptyset \in \mathcal{F}$.*
L'algorithme GLOUTON fournit une solution optimale pour toute pondération w (positive) si et seulement si $\mathcal{S}$ est un matroïde.

Question 2 – Discuter de la complexité de l'algorithme GLOUTON.

Question 3 – Soit $\mathcal{S} = (E, \mathcal{F})$ un matroïde. Montrer que l'ensemble calculé par l'algorithme GLOUTON est une base de $\mathcal{S}$.

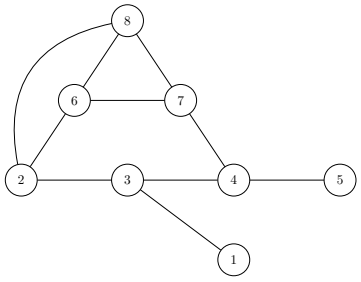
Question 4 – À l'aide du théorème précédent (que l'on ne demande pas de démontrer), montrer que l'algorithme de Kruskal calcule une solution au problème de recherche d'un arbre couvrant de poids minimal.

2 Coloration de graphes

Une coloration d'un graphe non-orienté G est une fonction qui associe une couleur à chaque sommet de G sous la contrainte suivante : deux sommets d'un même arc possèdent des couleurs différentes. On s'intéresse ici à trouver une coloration minimale, *i.e.* utilisant le minimum de couleurs différentes.

Question 5 – Que peut-on dire du nombre de couleurs nécessaire à la coloration d'un graphe possédant une clique de cardinal k ? (rappel : une clique est un sous-graphe complet)

Question 6 – Proposer un algorithme glouton calculant une coloration. Cet algorithme devra calculer une coloration minimale pour tout graphe biparti.



Question 7 – Colorer le graphe ci-dessus à l'aide de votre algorithme. On impose la contrainte suivante : à chaque étape, si des sommets différents peuvent être colorés, on colorera en priorité les nœuds dont l'étiquette est la plus petite.

Question 8 – Reprendre la question précédente en départageant maintenant les cas d'égalité en choisissant en priorité les sommets dont l'étiquette est la plus grande. Commenter le résultat obtenu.

3 Programmation dynamique

Quand un correcteur orthographique trouve une erreur dans un texte, il cherche dans son dictionnaire pour trouver un mot le plus proche. Pour mesurer la distance entre deux mots, on cherche à les *aligner*. Par exemple, voici deux alignements de `SNOWY` et `SUNNY` :

```
S - N O W Y          - S N O W - Y
S U N N - Y          S U N - - N Y
  Coût : 3             Coût : 5
```

Le symbole '`-`' indique que l'on n'avance pas dans le mot. Le coût d'un alignement est défini comme le nombre de colonnes avec une différence.

Ce type de distance s'appelle *distance d'édition* : en effet elle correspond au nombre minimum d'opérations d'édition (insertion, suppression ou modification d'un caractère) pour passer du premier mot au deuxième. Par exemple l'alignement de gauche correspond à :

- l'insertion de `U` ;
- la modification d'un `O` en `N` ;
- et la suppression d'un `W`.

Le but de cet exercice est d'utiliser la programmation dynamique pour calculer la distance d'édition d'un mot $x[1..m]$ à un mot $y[1..n]$.

Question 9 – Identifier les sous-problèmes puis exprimer et expliquer la relation de récurrence qui permet de résoudre le problème.

Question 10 – Écrire l'algorithme en programmation dynamique et analyser les besoins en temps d'exécution et en en espace de votre algorithme.

Question 11 – Exécuter votre algorithme sur les mots `caveau` et `cravate`.