MITRO210 : Automates et données structurées

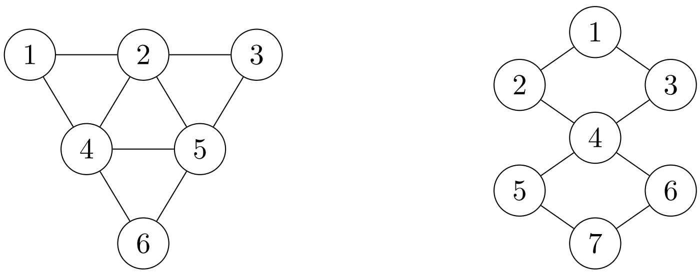
Feuille d'exercices 7

Antoine Amarilli

1 Quelques exemples pour la largeur arborescente

Le but de cet exercice est de s'approprier la notion de largeur arborescente en la calculant sur des exemples simples.

On considère les graphes G_1 et G_2 suivants :



Question 0. Ces graphes sont-ils de largeur arborescente 1 ?

Question 1. Pour chaque graphe, construire une décomposition arborescente de largeur 2.

Question 2. Calculer une décomposition arborescente normalisée de largeur 2 de G_1.

Question 3. Montrer que, pour tout graphe $G = (V, E)$, pour tout sous-graphe $G' = (V, E')$ de G (avec $E' \subseteq E$), la largeur arborescente de G' est au plus celle de G.

Question 4. Soit G'_1 le graphe obtenu en ajoutant une arête entre les sommets 1 et 5 du graphe G_1. Déduire de la question précédente et d'un résultat du cours que G'_1 a une largeur arborescente d'au moins 3.

Question 5. Proposer une décomposition arborescente de largeur 3 de G'_1.

2 Propriétés élémentaires de la largeur arborescente

Le but de cet exercice est de vérifier deux propriétés élémentaires de la notion de largeur arborescente.

Question 0. Soient $G = (V, E)$ et $G' = (V', E')$ deux graphes de largeur arborescente respectivement k et k'. On construit le graphe H comme l'union disjointe de G et de G', c'est-à-dire, en supposant sans perte de généralité quitte à renommer les sommets que V et V' sont disjoints, on pose $H = (V \sqcup V', E \sqcup E')$.

Exprimer la largeur arborescente de H en fonction de k et de k'.

Question 1. Soit $G = (V, E)$ un graphe arbitraire et k sa largeur arborescente. On construit un nouveau graphe G' de la manière suivante : on crée un nouveau sommet $x \notin V$, et on connecte x à certains sommets de G. Formellement, on prend un sous-ensemble $U \subseteq V$ et on définit $G' = (V \sqcup \{x\}, E \sqcup \{\{x, v\} \mid v \in U)$.

Montrer que la largeur arborescente de G' est au plus de $k + 1$.

3 Quelques révisions

Le but de cet exercice est de rappeler certaines notions et algorithmes vus en cours.

Question 0. Calculer le tableau préfixe de l'algorithme KMP pour le mot *atamatatamama*.

Question 1. Calculer les éléments du monoïde de transition de l'automate minimal du langage $\Sigma^*(baa + ab)\Sigma^*$. Ce monoïde est-il apériodique ?

Question 2. Calculer le trie suffixe compressé du mot *abracadabra*.

Question 3. On considère la condition suivante sur l'alphabet $\Sigma = \{a, b, c\}$: il n'y a jamais plus de deux a consécutifs, et à chaque fois qu'il y a deux a consécutifs il y a un b immédiatement après. Combien y a-t-il de mots de longueur 42 qui satisfont cette condition ? Exprimer la réponse comme $\vec{u}M^{42}\vec{v}$ pour un certain choix de matrice M et de vecteurs $\vec{u}$ et $\vec{v}$.

4 3-coloriage de coût minimal

Dans cet exercice, on étudie de bout en bout une variante du problème de 3-coloriage avec contraintes : le problème du 3-coloriage de coût minimal.

On fixe l'alphabet $\Sigma = \{R, G, B\}$. Un *graphe pondéré* est la donnée $G = (V, E, \kappa)$ d'un graphe non-orienté (V, E) et d'une fonction de coût $\kappa \colon V \times \Sigma \to \mathbb{N}$. Intuitivement, la valeur $\kappa(u, x)$ pour $u \in V$ et $x \in \Sigma$ indique le coût de colorier le nœud u avec la couleur x.

Un *coloriage* de G est une fonction $c \colon V \to \Sigma$ associant une couleur à chaque nœud. Le *coût* du coloriage c est $\sum_{u \in V} \kappa(u, c(u))$. Le coloriage est *légal* s'il ne donne jamais la même couleur à deux sommets voisins : pour chaque arête $\{u, v\} \in E$, on a $c(u) \neq c(v)$. Dans le problème du *coloriage de coût minimal*, on reçoit en entrée un graphe pondéré et on cherche à connaître le coût minimum d'un coloriage légal.

Question 0. Le coût minimum d'un coloriage légal est-il toujours bien défini ?

Question 1. Proposer un algorithme naïf pour résoudre le problème du coloriage de coût minimal. Quelle est sa complexité ?

Question 2. Montrer que le problème de 3-coloriage avec contraintes (3CC) sur un graphe étiqueté se réduit au problème de 3-coloriage de coût minimal : pour tout graphe étiqueté $G = (V, E, \lambda)$ on peut efficacement construire une fonction de coût κ tel que la réponse au problème 3CC sur G se déduise de la réponse au problème du coloriage de coût minimal sur (V, E, κ).

Question 3. Expliquer pourquoi on peut supposer sans perte de généralité que le graphe d'entrée est connexe.

Question 4. On suppose d'abord que le graphe d'entrée est un chemin : $V = \{u_1, \ldots, u_n\}$, et $E = \{\{u_i, u_{i+1}\} \mid 1 \leq i < n\}$. Expliquer comment résoudre le problème de 3-coloriage de coût minimal sur un tel graphe. On pourra utiliser la programmation dynamique.

Question 5. On suppose à présent que le graphe d'entrée est un arbre, et que chaque nœud est de degré au plus 3. Expliquer comment résoudre efficacement le problème de 3-coloriage de coût minimal sur un tel graphe.

Question 6. On ne suppose toujours que le graphe d'entrée est un arbre, mais on ne fait plus d'hypothèse sur le degré des nœuds. Expliquer comment résoudre efficacement le problème de 3-coloriage de coût minimal.

Question 7. On suppose enfin que le graphe d'entrée est de largeur arborescente k pour une certaine constante $k \in \mathbb{N}$. Expliquer comment résoudre efficacement le problème de 3-coloriage de coût minimal. Quelle est la complexité obtenue ?