Codage de Huffman et graphes

Exercice 1 Plus courts chemins et arbres couvrants de poids minimal

Soit un graphe non-orienté $G = (V, E)$. On considère que chacune de ses arrêtes $e \in E$ a un poids w_e. On considère que l'on a calculé un arbre couvrant de poids minimal T_c de G, ainsi qu'un arbre des plus courts chemins T_s entre un sommet $s \in V$ de G et tous les autres sommets. On ajoute alors 1 au coût de chaque arrête ($w'_e = w_e + 1$).

1. T_c est-il toujours un arbre couvrant de poids minimal de G ? Si non, donner un contre exemple.

2. T_s est-il toujours un arbre de plus courts chemins entre s et les autres sommets de G ? Si non, donner un contre exemple.

3. Est-ce que arbres des plus courts chemins et arbres couvrants de poids minimal coïncident en général ? Si non, donner un contre exemple.

Exercice 2 Codage de Huffman

Les codages de Huffman sont utilisés pour la compression de données sans perte. L'objectif est de montrer que ces codages sont en fait optimaux.

Soit Σ un alphabet. Soit $\phi : \Sigma \to \{0,1\}^*$ un code (c'est-à-dire que la fonction $\phi^* : \Sigma^* \to \{0,1\}^*$ telle que $\phi^*(a_1 \dots a_n) = \phi(a_1) \dots \phi(a_n)$ est injective). Le code ϕ est optimal pour le compression d'un texte T (sur Σ) si $|\phi(T)|$ est minimale.

Par exemple, pour $\Sigma = \{a, b, c, d, e, f\}$, la fonction suivante est un code :

lettre	a	b	c	d	e	f
ϕ	000	001	010	011	100	101

Dans cet exemple, chaque lettre a un code de même longueur. Lorsqu'on connaît la fréquence $freq(x)$ d'apparition de chaque lettre x dans T donnÈ, il est souhaitable d'associer un code plus court aux lettres très fréquentes et un code plus long aux lettres rares. La longueur du texte compressé est alors $\sum_{x \in \Sigma} freq(x)|\phi(x)|$.

On se limite à l'étude des *codes préfixes*, c'est-à-dire des codes ϕ tels que pour tout $x, y \in \Sigma$, $\phi(x)$ n'est pas un préfixe de $\phi(y)$. (On peut montrer qu'il existe toujours un code préfixe optimal).

1. Expliquer l'intérêt des codes préfixes.

Un code préfixe se représente sous la forme d'un arbre binaire où les feuilles sont les lettres de Σ et où le chemin menant de la racine à x fournit le code $\phi(x)$ avec la convention suivante : une bifurcation à gauche représente 0 et une bifurcation à droite représente 1.

2. Quel est l'arbre correspondant à la fonction de codage suivante ?

lettre	a	b	c	d	e	f
ϕ	0	1010	100	110	111	1011

3. Si l'on prend $freq(a) = 45$, $freq(b) = 13$, $freq(c) = 12$, $freq(d) = 16$, $freq(e) = 9$ et $freq(f) = 5$, ce code est-il optimal ?

4. Montrer qu'un arbre binaire qui n'est pas localement complet ne peut pas correspondre à un code préfixe optimal. Montrer de même que les deux lettres de plus basses fréquences sont nécessairement à même profondeur, et qu'il n'y a pas de lettres plus profond dans l'arbre.

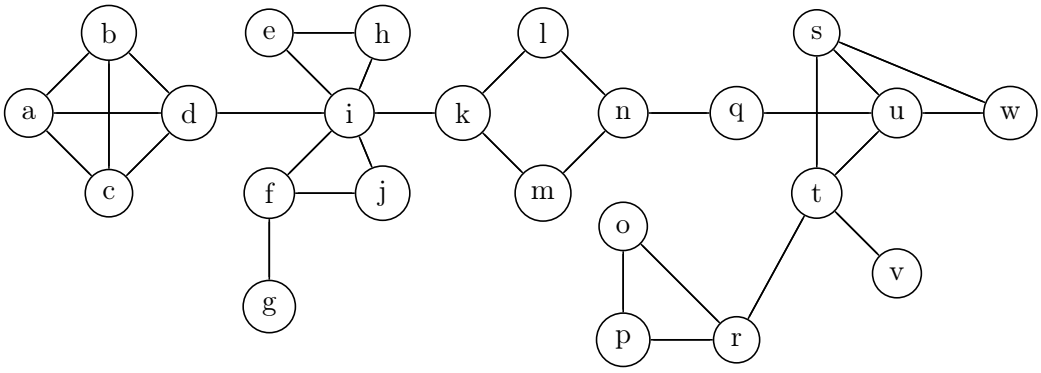
5. Donner un algorithme *glouton* qui construit un arbre binaire correspondant à un code préfixe optimal. Prouver sa correction et évaluer sa complexité. Appliquer votre algorithme aux fréquences données plus haut.

`Exercice 3` `Points d'articulation, isthmes, composantes 2-connexes`

Soit $G = (V, E)$ un graphe non orienté connexe. Dans G :
- un *point d'articulation* est un sommet dont la suppression déconnecte le graphe ;
- un *isthme* est une arrête dont la suppression déconnecte le graphe ;
- une *composante 2-connexe* est un ensemble maximal connexe de sommets tel que le sous-graphe induit par ces sommets n'a pas de point d'articulation.

On notera $T = (V, F)$ un arbre de parcours de G obtenu par parcours en profondeur et *debv* le début de visite de v dans la numérotation associée à ce parcours en profondeur.

1. Sur l'exemple ci-dessous, déterminer les points d'articulation, les isthmes et les composantes 2-connexes.



2. Expliquer pourquoi dans le parcours en profondeur d'un graphe non orienté et connexe on n'obtient que deux sortes d'arcs : les arcs de l'arbre et les arcs retour.

3. Montrer que la racine de T est un point d'articulation si et seulement si elle a au moins deux fils dans T.

Dans toute la suite « descendant »est pris au sens large : x est un descendant de x.

4. Soit x un sommet de T distinct de la racine. Prouver que x est un point d'articulation si et seulement s'il existe un fils y de x dans l'arbre tel qu'il n'existe pas d'arc retour de la forme (z, t) où z est un descendant de y et t est un ancêtre propre de x.

On définit la fonction `au_dessous`, de V dans $\mathbb{N}$ par :

```
au_dessous(x) := min({deb(x)} U A(x))
        A(x) := {deb(z)|(y,z) est un arc retour avec y descendant de x}
```

5. Donner un algorithme en $O(|V| + |E|)$ de calcul de la fonction `au_dessous`. On en prouvera la complexité.

6. Montrer comment calculer les points d'articulation en $O(|V| + |E|)$.

7. Prouver qu'une arête est un isthme si et seulement si elle n'appartient à aucun cycle élémentaire.

8. Montrer comment calculer les isthmes en $O(|V| + |E|)$.

9. Considérer le graph H dont les sommets sont les composantes 2-connexes de G et tel que deux composantes sont adjacentes si et seulement si elles ont au moins un sommet en commun. Que pouvez vous dire de ce graphe ?

Exercice 4 Fermeture réflexive et transitive d'un graphe dynamique

Soit $G = (S, A)$ un graphe orienté auquel on ajoute au fur et à mesure de nouveaux arcs. On souhaite calculer sa fermeture réflexive et transitive $G^* = (S, A^*)$ et pouvoir la mettre à jour efficacement. On représente G^* par sa matrice d'adjacence.

1. Montrer comment G^* peut Ítre mis à jour quand un nouvel arc est inséré en un temps $O(|S|^2)$. Donner un exemple dans lequel cette borne est nécessairement atteinte.

2. Décrire un algorithme efficace de mise à jour : pour une suite quelconque d'arcs $a_1, \ldots, a_n$, à partir d'un ensemble d'arcs vide, le temps total d'exécution de l'algorithme devra être en $\sum_{i=1}^{n} t_i = O(|S|^3)$, où t_i est le temps nécessaire à la i-ème mise à jour.