ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Section d'Informatique et de Systèmes de Communication

Corrigé de la série 12 12 Dec. 2011

1. Augmenter les poids

a) Par la formule d'Euler on sait que tous les arbres couvrants ont $|V| - 1$ arêtes. Donc si T_1 est un arbre couvrant de poids $w(T_1)$, l'augmentation des poids sur les arêtes par une constante c fait que le nouveau poids de T_1 sera $w(T_1) + (|V| - 1) \cdot c$.

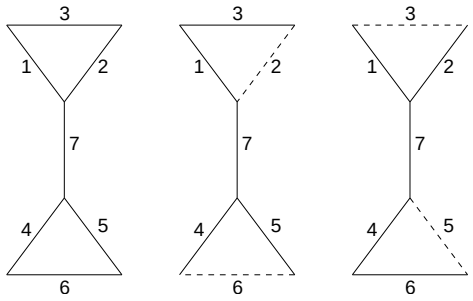
Soit T_2 un autre arbre couvrant. Comme on a $w(T_1) + (|V| - 1) \cdot c \leq w(T_2) + (|V| - 1) \cdot c$ si et seulement si $w(T_1) \leq w(T_2)$, l'ordre partiel défini par le poids est préservé et donc en particulier, les éléments minimaux restent les mêmes.

b) Oui. Si on diminue le poids de l'une des arêtes de T d'une constante c, son nouveau poids devient $w(T) - c$. Pour tout autre arbre couvrant T' le nouveau poids sera soit $w(T') - c$ (si l'arête dont on a diminué le poids appartient à T'), soit $w(T')$ (si elle n'appartient pas à T'). Dans les deux cas le nouveau poids de T est plus petit que le nouveau poids de T', car $w(T) \leq w(T')$.

2. Deuxième meilleur arbre couvrant

a) Supposons qu'on a deux arbres couvrants minimaux T_1 et T_2. Considerer les coûts des arêtes de T_1 et T_2 de manière croissante, soit e la première arête qui apparait dans un arbre mais pas dans l'autre. Si $e \in T_1$ on ajoute e dans T_2. Il y aura un cycle dans T_2, et dans ce cycle il existe une arête f telle que $w(f) > w(e)$ (sinon toutes les arête dans le cycle sont dans T_1 mais T_1 est un arbre). Si on enlève f de T_2 on aura un arbre T_3 de coût plus petit que le coût de T_2 ce qui est une contradiction puisque T_2 est un arbre couvrant minimum.

On montre que le deuxième meilleur arbre couvrant n'est pas unique à l'aide d'un exemple.



b) Nous montrons qu'un arbre qui n'est pas optimal peut toujours être transformé en un meilleur arbre en ajoutant une arête et enlevant une autre arête.

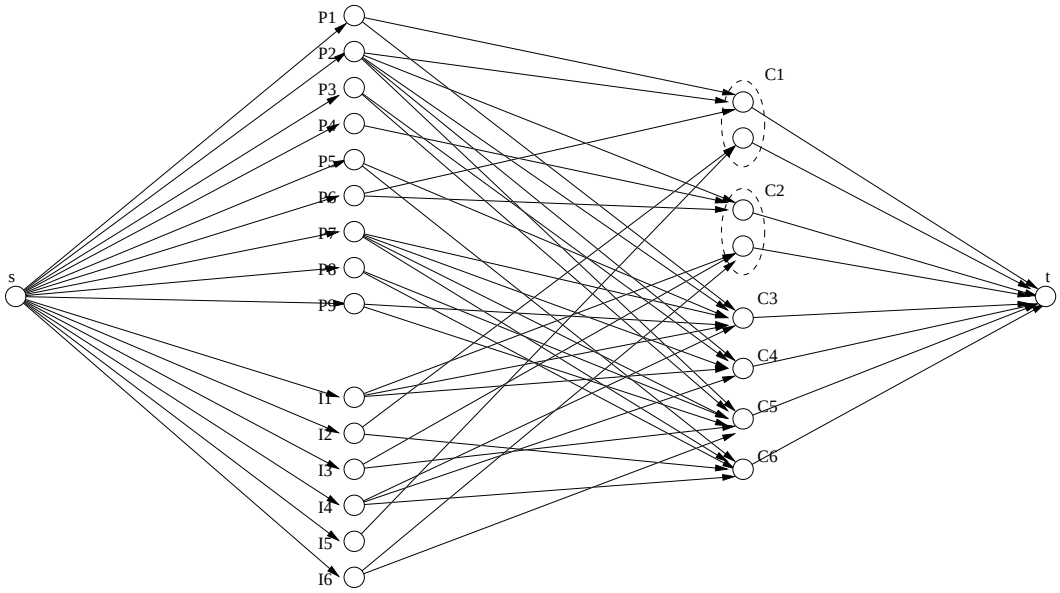
Comme n'importe quelle amélioration du deuxième meilleur arbre résulte en un arbre optimal, ceci permet de conclure.

Soit T un arbre couvrant non-optimal quelconque, et T_{opt} l'arbre optimal obtenu en appliquant l'algorithme de Kruskal. Soit $E = \{e_1, \ldots, e_m\}$ avec $e_i < e_{i+1}$ si $1 \leq i \leq m-1$. Il faut montrer qu'on peut améliorer T en ajoutant une arête et enlevant une autre. Soit i le plus petit indice tel que soit $e_i \in T_{\text{opt}} \setminus T$ ou $e_i \in T \setminus T_{\text{opt}}$. Alors $e_i \in T \setminus T_{\text{opt}}$: si ce n'était pas le cas, par construction de T_{opt}, ajouter e_i à T_{opt} créera un cycle se constituant d'arêtes d'indice $\leq i$ dans T_{opt}. Comme les mêmes arêtes sont dans T, le même cycle serait aussi dans T, mais T est un arbre. Donc $e_i \in T_{\text{opt}} \setminus T$. En ajoutant e_i dans T, on obtient un cycle contenant au moins une arête j d'indice $> i$. On enlève cette arête, ce qui résulte en un arbre de poids $w(T) - w(e_j) + w(e_i) < w(T)$, comme $j > i$.

3. Attribution optimale

a) Le problème ressemble à un problème de bipartite matching, et donc on veut appliquer l'algorithme max-flow-min-cut pour le résoudre. La seule subtilité est celle des chambres doubles. On peut la résoudre en travaillant plutôt sur des "lits" qu'on met dans les chambres. Dans les chambres simples on met un lit simple, dans les chambres doubles on en met deux: un pour un informaticien et un pour un physicien.

b) On obtient le graphe suivant:



4. L'algorithme de Karp

a) Soit $s \to \cdots \to x$ un chemin de poids minimal et de longueur k. On suppose $k \geq 2$ (l'autre cas est immédiatement réglé), et alors on peut écrire ce chemin $s \to \cdots \to u \to x$, où u est l'avant-dernier sommet sur ce chemin. Le chemin $s \to \cdots \to u$ est de longueur $k-1$ et certainement minimal, parce que sinon on pourrait le remplacer par le chemin minimal et on aura aussi un meilleur chemin de s à x.

Donc par hypothèse d'induction le poids du morceau $s \to \cdots \to u$ est $F_{k-1}(u)$, et le chemin $s \to x$ a certainement poids

$$F_{k-1}(u) + w(u, x).$$

Le reste est évident.

b) Comme le graphe est connexe, certainement $|V| \leq |E| + 1$, ce qui montre que F_0 peut bien être calculé en $O(|E|)$ opérations. La suite est par induction sur k.

Supposons qu'on a déjà calculé $F_{k-1}, \ldots, F_0$ en $O(|E|k)$ opérations. Si nous montrons que F_k peut alors être trouvé en $O(|E|)$, nous avons fini. Il s'agit d'appliquer la formule du point a). On procède comme suit:

for $x \in V$ **do**
  $F_k(x) \leftarrow \infty$
for $(x, y) \in E$ **do**
  $u \leftarrow F_{k-1}(x) + w(x, y)$
  if $u < F_k(y)$ **then**
    $F_k(y) \leftarrow u$.

Comme $|V| = O(|E|)$, cet algorithme utilise évidemment $O(|E|)$ opérations. Il est correct à cause du point précédent.

c) D'abord on calcule les F_k et ensuite le μ^*. Voir l'Algorithme 1 pour les détails.

Algorithme 1 Karp

Choisir $s \in V$ arbitraire.
for $x \in V$ **do**
  $F_0(x) \leftarrow \infty$
$F_0(s) \leftarrow 0$
for $k = 1, \ldots, n$ **do**
  for $x \in V$ **do**
    $F_k(x) \leftarrow \infty$
  for $(x, y) \in E$ **do**
    $u \leftarrow F_{k-1}(x) + w(x, y)$
    if $u < F_k(y)$ **then**
      $F_k(y) \leftarrow u$.
$\mu^* \leftarrow \infty$
for $x \in V$ **do**
  $\mu_{\text{tmp}} \leftarrow -\infty$
  for $k = 0, \cdots, n-1$ **do**
    $u \leftarrow \frac{F_n(x) - F_k(x)}{n-k}$
    if $u > \mu_{\text{tmp}}$ **then**
      $\mu_{\text{tmp}} \leftarrow u$
  if $\mu_{\text{tmp}} < \mu^*$ **then**
    $\mu^* \leftarrow \mu_{\text{tmp}}$