ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Section d'Informatique et de Systèmes de Communication

Corrigé de la série 11 7 Décembre 2007

1. 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.

2. Shift cyclique

a) k est la position du plus petit élément de la suite. Pour tout $x = 1, \ldots, n-1$ nous avons
$$x \neq k \implies a_{x-1} < a_x, \tag{1}$$
et de plus
$$k \neq 0 \implies a_{n-1} < a_0. \tag{2}$$

$\implies$: Si $a_j < a_i$, supposons par l'absurde que k ne se trouve pas entre i et j (donc $k \leq i$ ou $k > j$). Nous avons par (1)
$$a_i < a_{i+1} < \ldots < a_j,$$

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**
    $\mu^* \leftarrow \mu_{\text{tmp}}$

et donc $a_i < a_j$, ce qui est une contradiction. Notre supposition etait donc fausse et donc $i < k \leq j$.

$\Longleftarrow$: Si $i < k \leq j$, alors par (1) nous avons

$$a_j < a_{j+1} < \ldots < a_{n-1} \tag{3}$$

et

$$a_0 < \ldots < a_{i-1} < a_i. \tag{4}$$

De plus puisque $0 \leq i < k$, nous avond $k \neq 0$, et donc par (2)

$$a_{n-1} < a_0 \tag{5}$$

En combinant maintenant (3), (4) et (5) nous obtenons

$$a_j < a_{n-1} < a_0 < a_i,$$

et donc $a_j < a_i$.

b) Une fois que nous avons trouvé la position k du plus petit élément, nous savons qu'il faut shifter la suite de k positions vers la gauche (dans l'exemple de l'énoncé nous avions comme plus petit élément $a_4 = 1$, et il fallait shifter la suite de 4 positions vers la gauche).

La question précédente nous permet de déterminer, étant donné i, j si k se trouve dans l'intervalle $]i, j]$. En posant $i = 0$ et $j = \lfloor n/2 \rfloor$, nous pouvons déterminer si k se trouve dans la première moité ou la deuxième moitié de la suite. Puis nous pouvons appliquer ce procédé récursivement pour trouver la position exacte de k (de manière très similaire à une *recherche binaire*).

3.Induction et Récursion

a) $A_4 = (0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101, 1111, 1110, 1010, 1011, 1001, 1000)$

b) Nous procédons par induction sur i.
Base : Pour $i = 1$ nous avons

$$A_1 = (0, 1),$$

et les deux éléments diffèrent bien en exactement une position.

Pas d'induction : Supposons que l'affirmation est vraie pour $i - 1$. Donc deux éléments successifs de A_{i-1} diffèrent en exactement une position. Supposons que

$$A_{i-1} = (x_1, \ldots, x_\ell).$$

Alors par définition nous avons

$$A_i = (0 : x_1, \ldots, 0 : x_\ell, 1 : x_\ell, \ldots, 1 : x_1)$$

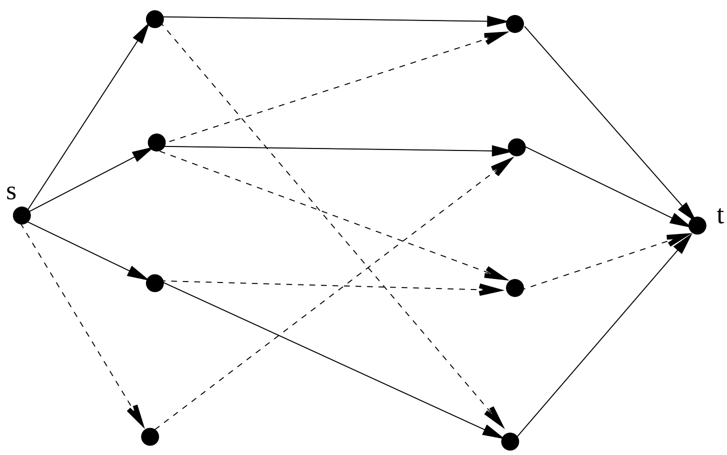
par l'hypothèse d'induction, pour tout $j = 1, \ldots, \ell - 1$, x_j et x_{j+1} diffèrent en exactement une position. Donc clairement, pour tout $j = 1, \ldots, \ell - 1$ $\quad 0 : x_j$ et $0 : x_{j+1}$ diffèrent en exactement une position. De même pour $\quad 1 : x_{j+1}$ et $1 : x_j$.
Finalement, $\quad 0 : x_\ell$ et $1 : x_\ell$ diffèrent en exactement une position (la première).

c) Nous pouvons montrer cette affirmation de nouveau par induction.

4. Matching

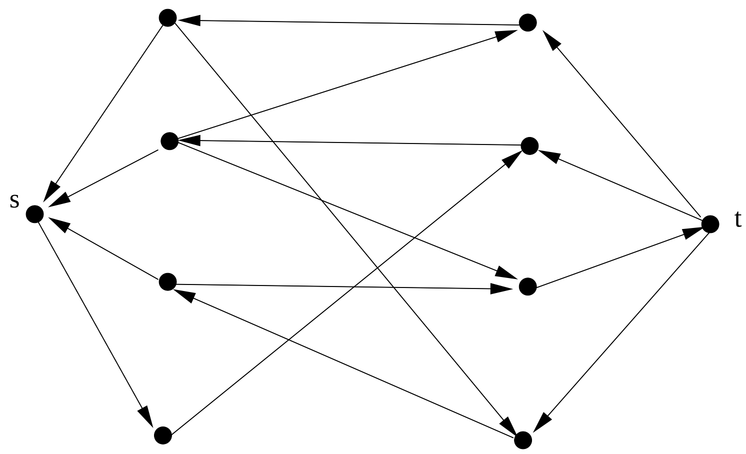
Nous utilisons l'algorithme MaxFlowMinCut comme décrit dans le cours.

Nous ajoutons 2 sommets s et t et transformons le graphe en réseau dans lequel toutes les arêtes ont une capacité de 1. Au matching de l'énnoncé correspond un flux (voir cours), nous dessinons ce flux ci-dessous :

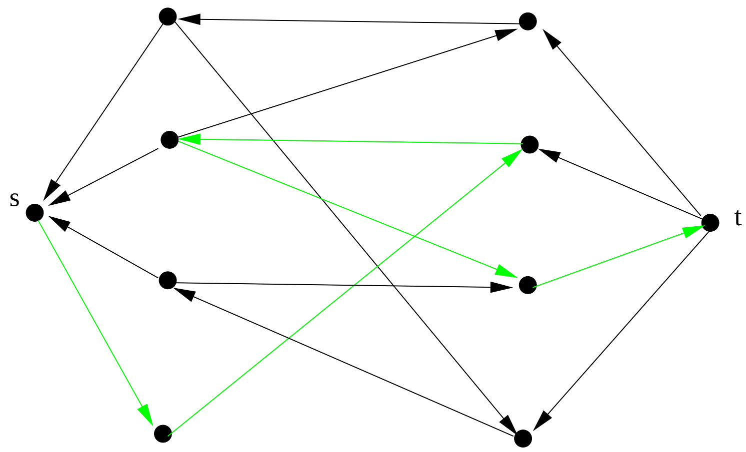


(Toutes les arêtes ont capacité 1, Celles en pointillé ont un flux de 0/1, alors que les autres ont un flux de 1/1). La valeur de ce flux est 3.

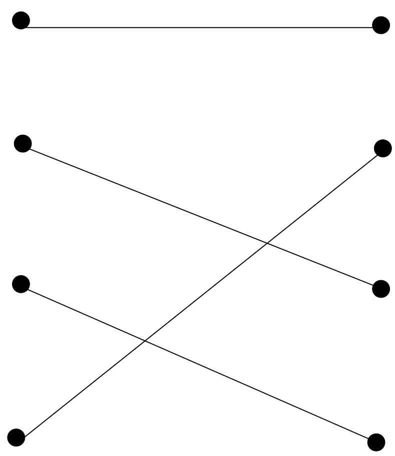
Nous dessinons ensuite le graphe résiduel par rapport à ce flux afin de voir s'il est possible de trouver un chemin augmentant :



Nous prenons le chemin augmentant ci-dessous :



Et obtenons ainsi le matching suivant :



Le matching proposé n'était donc pas maximal.