Examen d'algorithmique

Jeudi 5 janvier 2016 15h30–18h / Aucun document autorisé

Mode d'emploi : Le barème est donné à titre indicatif. **La qualité de la rédaction des algorithmes et des explications sera très fortement prise en compte pour la note.**
On peut toujours supposer une question résolue et passer à la suite.
L'annexe contient plusieurs algorithmes vus en cours.

Exercice 1 : Graphes – (6 points)

On considère un graphe orienté $G = (S, A)$.

1. Ecrire une fonction `Test-Accessibilité`(G, x) qui étant donnés un graphe G et un sommet x retourne *Vrai* si et seulement si tous les sommets de G sont accessibles depuis x.
2. Ecrire une fonction `Affiche-Chemin`(Π, x) qui étant donnés une table des prédécesseurs calculée par un parcours de graphe (en profondeur ou en largeur) et un sommet x, affiche le chemin conduisant à x (depuis l'origine du parcours).
3. Ecrire une fonction `Acces-Circuit`(G, x) qui étant donnés un graphe G et un sommet x retourne *Vrai* si et seulement si il est possible d'atteindre un circuit [1] depuis le sommet x.
4. Ecrire une fonction `Circuit`(G, x) qui étant donnés un graphe G et un sommet x affiche un circuit contenant x si un tel circuit existe.

Exercice 2 : Recherche d'arbres couvrants minimaux – (4 points)

1. Rappeler ce qu'est un arbre couvrant minimal d'un graphe non-orienté valué $G = (S, A, w)$. Donner deux ACM du graphe de la figure ci-dessous.

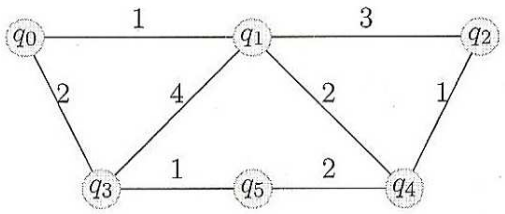


FIGURE 1 – Graphe pour l'exercice 2.

2. Expliquer comment procède un des algorithmes pour la recherche d'ACM vus en cours.
3. Appliquer l'algorithme choisi au graphe ci-dessus (on indiquera quelle arête est ajoutée au résultat à chaque étape).

1. *i.e.* une séquence de sommets $s_1 s_2 \dots s_k$ avec $k \geqslant 2$, $s_1 = s_k$ et $(s_i, s_{i+1}) \in A$ pour $i = 1, \dots, k-1$.

Problème : Plus courts chemins – (10 points)

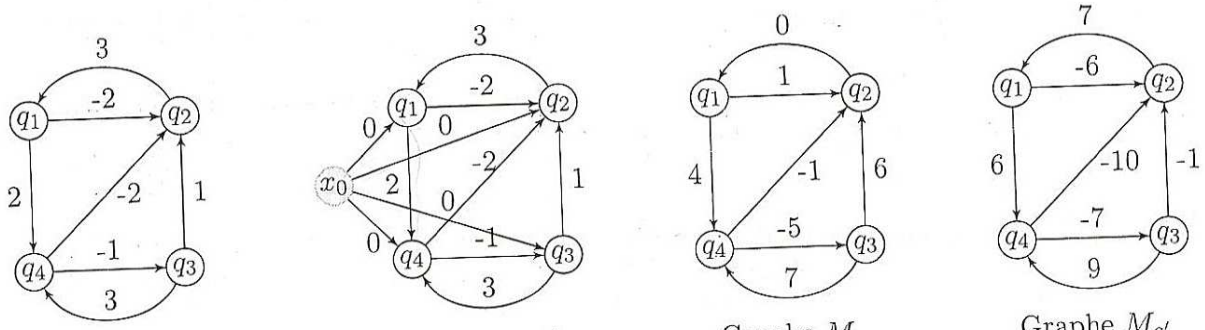
On s'intéresse au calcul des longueurs des plus courts chemins (PCC) dans un graphe orienté et valué $G = (S, A, w)$. L'objectif est de calculer, **lorsqu'elles existent**, ces longueurs pour toutes les paires de sommets : à la fin de l'algorithme, on veut connaître $\delta_G(x, y)$ pour tout $x, y \in S$ lorsque des PCC existent (NB : comme en cours, on note $\delta_G(x, y)$ la longueur d'un PCC entre x et y dans le graphe G).

Dans cet exercice, on va utiliser deux transformations de G :

- La première transformation est le graphe $G^+ = (S^+, A^+, w^+)$ tel que $S^+ = S \cup \{x_0\}$ où x_0 est un nouveau sommet, $A^+ = A \cup \{(x_0, u) \mid u \in S\}$ et : $w^+(u, v) = \begin{cases} w(u, v) & \text{si } (u, v) \in A \\ 0 & \text{si } u = x_0 \text{ et } v \in S \end{cases}$

  G^+ contient donc les sommets et les arcs de G avec les mêmes poids, mais contient en plus un nouveau sommet x_0 relié à tous les autres par des arcs de poids nul.
- Et étant donnée une fonction f qui associe à chaque sommet de S une valeur dans $\mathbb{R}$ (c.-à-d. $f : S \to \mathbb{R}$), on définit la seconde transformation comme le graphe $G_f = (S, A, w_f)$ où $w_f(u, v) = w(u, v) + f(u) - f(v)$. **$G_f$ a donc les mêmes sommets et les mêmes arcs que G, seuls les poids des arcs changent.**

La figure 2 présente un graphe M, son graphe associé M^+ ainsi que deux exemples de graphe "M_f" : le graphe M_g pour la fonction g définie par : $g(q_1) = 2$, $g(q_2) = -1$, $g(q_3) = 4$ et $g(q_4) = 0$ et le graphe $M_{g'}$ pour $g'(q_1) = -1$, $g'(q_2) = 3$, $g'(q_3) = 1$ et $g'(q_4) = -5$.



Graphe M — Graphe M^+ — Graphe M_g — Graphe $M_{g'}$

FIGURE 2 – Exemple d'un graphe M, sa transformation M^+ et deux transformations M_g et $M_{g'}$ pour les deux fonctions différentes g et g'.

Propriétés. Pour les questions suivantes, on suppose donnés un graphe $G = (S, A, w)$ et une fonction $f : S \to \mathbb{R}$.

1.a Soit $\rho = u \to \ldots \to v$ un chemin de u à v dans G (et dans G_f).
Exprimer la longueur $w_f(\rho)$ du chemin ρ dans G_f en fonction de $w(\rho)$ (c'est-à-dire la longueur du même chemin dans G). En déduire que ρ est un plus court chemin entre u et v dans G **si et seulement si** ρ est un plus court chemin de u à v dans G_f.

1.b Montrer que G contient un circuit strictement négatif si et seulement si G_f contient un circuit strictement négatif.

✓1.c Montrer que G contient un circuit strictement négatif si et seulement si G^+ contient un circuit strictement négatif.

✓2.a **Exemple :** Donner les longueurs des PCC de x_0 aux autres sommets dans M^+ (c'est-à-dire : $\delta_{M^+}(x_0, q_1)$, $\delta_{M^+}(x_0, q_2)$, $\delta_{M^+}(x_0, q_3)$, et $\delta_{M^+}(x_0, q_4)$).

2.b **Exemple :** Construire le graphe M_h où h est la fonction suivante : $h(q_i) \stackrel{\text{def}}{=} \delta_{M^+}(x_0, q_i)$ pour $i = 1, \ldots, 4$.

Pour les 3 questions suivantes (3.a, 3.b, et 3.c), on suppose donné un graphe G sans circuit strictement négatif.

On note $h : S \to \mathbb{R}$ la fonction donnant la longueur des PCC entre x_0 et les sommets de S dans G^+ : c'est-à-dire $h(u) = \delta_{G^+}(x_0, u)$ pour tout $u \in S$. Et on va utiliser cette fonction pour construire le graphe $G_h = (S, A, w_h)$ défini comme précédemment.

3.a Montrer que l'on a toujours $h(v) \leqslant w(u, v) + h(u)$. En déduire que l'on a $w_h(u, v) \geqslant 0$ pour tout $(u, v) \in A$.

3.b Quel algorithme doit-on utiliser pour trouver la longueur d'un PCC entre u et v dans G_h ? Justifier votre réponse.

3.c Utiliser la question 1.a pour exprimer la longueur $\delta_G(u, v)$ d'un PCC entre u et v dans G en fonction de la longueur $\delta_{G_h}(u, v)$ d'un PCC entre u et v dans G_h.

Algorithme. L'algorithme de Johnson est décrit par l'algorithme 1. On y utilise une matrice de taille $|S| \times |S|$ pour représenter les distances des PCC entre chaque couple de sommets : pour les sommets u et v, le coefficient correspondant est noté d_{uv}.

4 **Exemple :** Appliquer l'algorithme de Johnson sur le graphe M de la figure 2 (sans détailler pas les appels des procédures `PCC-Bellman-Ford` et `PCC-Dijkstra`).

5 Expliquer ce que fait cet algorithme. Pourquoi utilise-t-il d'abord l'algorithme de Bellman-Ford ? Quelle est sa complexité ?

6 Pour le même objectif, quelle serait la complexité d'un algorithme qui n'utiliserait que Bellman-Ford pour les différents calculs ?

7 Pour quelles valeurs de $|S|$ et $|A|$, l'algorithme de Johnson est-il plus intéressant que l'algorithme de Floyd ?

Procédure `Algo-Johnson`(G)
//$G = (S, A, w)$: un graphe orienté, valué avec $w : A \to \mathbb{R}$.
begin
  $D :=$ `matrice`$(S \times S)$ telle que $d_{uv} = \infty$ pour tout $u, v \in S$
  $G^+ :=$ `Graphe`$(S \cup \{x_0\}, A \cup \{(x_0, u) \mid u \in S\}; \overline{w})$ avec $\overline{w}(u, v) = \begin{cases} w(u, v) & \text{si } u, v \in S \\ 0 & \text{si } u = x_0 \end{cases}$
  $(\text{res}, \text{d}, \Pi) :=$ `PCC-Bellman-Ford`(G^+, x_0)
  si res $== \perp$ **alors**
    Afficher("le graphe G contient un cycle strictement négatif.")
  sinon
    pour chaque $v \in S$ **faire** $h(v) := \text{d}(x_0, v)$
    pour chaque $(u, v) \in A$ **faire** $w_h(u, v) := w(u, v) + h(u) - h(v)$
    $G_h :=$ `Graphe`(S, A, w_h)
    pour chaque $u \in S$ **faire**
      $(\text{d}', \Pi) =$ `PCC-Dijkstra`(G_h, u)
      pour chaque $v \in S$ **faire** $d_{uv} := \text{d}'[v] + h(v) - h(u)$
  return D
end

Algorithme 1 : algorithme de Johnson

Annexe

On rappelle ci-dessous l'algorithme de parcours en profondeur : la procédure de base PP et la procédure principale PPC.

Procédure PP(G, s)
$//G = (S, A)$
begin
  Couleur$[s]$:= gris;
  $temps++$;
  d$[s] := temps$;
  pour chaque $(s, u) \in A$ **faire**
    si *Couleur*$[u]$ = *blanc* **alors**
      $\Pi[u] := s$;
      PP(G, u) ;
  Couleur$[s]$:= noir ;
  $temps++$;
  f$[s] := temps$;
end

Procédure PPC(G)
$//G = (S, A)$
begin
  pour chaque $x \in S$ **faire**
    Couleur$[x]$:= blanc ;
    $\Pi[x]$:= nil ;
  $temps := 0$;
  pour chaque $x \in S$ **faire**
    si *Couleur*$[x]$ = *blanc* **alors**
      PP(G, x) ;
end

On rappelle ci-dessous l'algorithme de Dijkstra et l'algorithme de Bellman-Ford :

Procédure PCC-Dijkstra(G, s)
$//G = (S, A, w)$: un graphe orienté valué
$//s \in S$: un sommet origine.
begin
  pour chaque $u \in S$ **faire**
    $\Pi[u]$:= nil
    $\mathsf{d}[u] := \begin{cases} 0 & \text{si } u = s \\ \infty & \text{sinon} \end{cases}$
  F := File(S, d)
  tant que $F \neq \emptyset$ **faire**
    u := Extraire-Min(F)
    pour chaque $(u, v) \in A$ **faire**
      si d$[v] >$ d$[u] + w(u, v)$ **alors**
        d$[v]$:= d$[u] + w(u, v)$
        $\Pi[v] := u$
        MaJ-F-Dijkstra(F, d, v)
  return (d, Π)
end

Procédure PCC-Bellman-Ford(G, s)
$//G = (S, A, w)$: un graphe orienté valué
$//s \in S$: un sommet origine
begin
  pour chaque $u \in S$ **faire**
    $\Pi[u]$:= nil
    $\mathsf{d}[u] := \begin{cases} 0 & \text{si } u = s \\ \infty & \text{sinon} \end{cases}$
  pour $i = 1$ *à* $|S| - 1$ **faire**
    pour chaque $(u, v) \in A$ **faire**
      si $d[v] > d[u] + w(u, v)$ **alors**
        d$[v]$:= d$[u] + w(u, v)$
        $\Pi[v] := u$
  pour chaque $(u, v) \in A$ **faire**
    si $d[v] > d[u] + w(u, v)$ **alors**
      return $(\bot, -, -)$
  return $(\top, d, \Pi)$
end