UE Androide/MAOA : Novembre 2015.

Durée 2h00. Notes de cours et de TD autorisées.

Exercice 1 (9 points)

Partie 1

On considère le problème $1|r_i|L_{max}$, où n tâches $J_1, \cdots, J_n$ de durée p_i soumises à des dates de disponibilité r_i et des dates d'échéance d_i, doivent être exécutées sur une machine de telle sorte que le retard (algébrique) $L_{max} = \max_{i=1}^{n}(C_i - d_i)$ soit minimum (C_i est la date de fin d'exécution de la tâche J_i).
Question 1 $(0,5/9)$ — Rappelez la complexité de ce problème.
Question 2 $(1/9)$ — Supposons pour cette question que $r_i = r$ pour $J_1, \cdots, J_n$. En vous appuyant sur un résultat vu en cours, expliquez comment résoudre $1|r_i = r|L_{max}$ en temps polynomial. Précisez la complexité.
Question 3 $(1/9)$ — Supposons pour cette question que $d_i = d$ pour $J_1, \cdots, J_n$. Montrez que l'ordonnancement obtenu en séquençant les tâches dans l'ordre des dates de disponibilité croissantes est optimal pour le problème $1|r_i|L_{max}$ avec $d_i = d$.
Question 4 $(1/9)$ — Supposons pour cette question que $p_i = 1$ (tâches de durées unitaires) pour $J_1, \cdots, J_n$. Montrez que l'ordonnancement obtenu en ordonnançant à chaque instant la tâche prête de plus petite date d'échéance est optimal pour le problème $1|r_i, p_i = 1|L_{max}$.

Partie 2

On considère le problème $1|r_i, p_i = 1, prec|L_{max}$, où n tâches $J_1, \cdots, J_n$ de durée unitaire soumis à des dates de disponibilité r_i et des dates d'échéance d_i, doivent être exécutées sur une machine de telle sorte que le retard (algébrique) $L_{max} = \max_{i=1}^{n}(C_i - d_i)$ soit minimum. On suppose que les dates d'échéance sont *compatibles avec les contraintes de précédence*, c'est-à-dire que : $(J_i, J_j) \in prec \Rightarrow d_i \leq d_j - p_j$.
On note $\hat{S}$ l'ordonnancement obtenu par l'algorithme glouton G suivant (une tâche est dite prête à la date t si, à cette date, tous ses prédécesseurs sont terminés) :
Question 5 $(2/9)$ — Déterminez l'ordonnancement $\hat{S}$ et son coût pour la donnée représentée sur la figure 1.

L'algorithme glouton G fournit un ordonnancement optimal pour $1|r_i, p_i = 1, prec|L_{max}$. Les questions suivantes permettent d'établir une partie de la preuve d'optimalité de G.
Question 6 $(1,5/9)$ — On note $C_i(\hat{S})$ la date de terminaison de la tâche J_i dans l'ordonnancement $\hat{S}$ et l'on note $C_i(S^*)$ la date de terminaison de la tâche J_i dans un ordonnancement *optimal* S^*. Soit t la plus petite unité de temps (slot) où les ordonnancements $\hat{S}$ et S^* diffèrent.

Initialiser le temps à la date de disponibilité minimale.;
tant que *Toutes les tâches n'ont pas été ordonnancées* **faire**
  si *l'ensemble des tâches prêtes est vide* **alors**
    placer un temps mort jusqu'à la prochaine date de disponibilité;
  fin
  sinon
    exécuter dans le premier slot libre une tâche prête de date d'échéance minimum;
  fin
fin

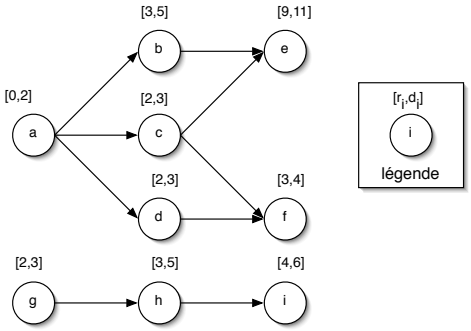


FIGURE 1 – une instance de $1|r_i, p_i = 1, prec|L_{max}$

1. Démontrez qu'il n'est pas possible que le slot t soit libre dans $\hat{S}$ et occupé par la tâche J_j dans S^*.
2. Démontrez que si le slot t est libre dans S^* et est occupé par la tâche J_i dans $\hat{S}$, alors il existe un ordonnancement optimal identique à $\hat{S}$ au moins jusqu'au slot t.

On suppose dans la suite que le slot t est occupé par la tâche J_i dans $\hat{S}$ et par la tâche J_j dans S^*. On note u le slot occupé par J_i dans S^*. On note alors E l'ensemble des tâches exécutées après le slot t et avant le slot u dans S^*.

Question 7 (1/9) — Montrez que $u > t$ et que $d_i \leq d_j$.

Question 8 (1/9) — Soit J_k une tâche de E.
Montrez que J_k n'est ni un ascendant de J_i ni un ascendant de J_j.
En déduire que, sans perte de généralité, on peut supposer que dans S^*, aucun slot n'est libre entre les slots t et u.

Exercice 2 (8 points)

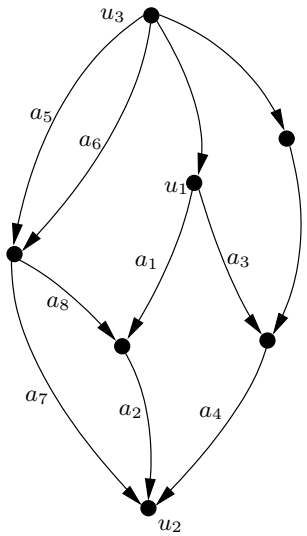
Gestion des domaines skiables

Soit un graphe $G = (V, A)$ où V est l'ensemble des sommets et A l'ensemble des arcs. Si deux arcs a et a' ont même extrémités u et v, on dit que a et a' sont *multiples*. Lorsqu'il n'y a pas ambiguïté, on note uv un arc de A allant du sommet u au sommet v. On appelle *chemin* dans G une suite d'arcs $P = (u_0u_1, u_1u_2, ..., u_{k-1}u_k)$ et pour un des sommets u_i de P, on dit que P "passe" par le sommet u_i. Un *circuit* C dans G est un chemin tel que $u_0 = u_k$. Un graphe est dit *acyclique* s'il ne contient aucun circuit.

Pour un vecteur binaire x indicé sur les arcs de $B \subset A$, on notera $x(B) = \sum_{a \in B} x(a)$.

Soit $G = (V, A)$ un graphe acyclique pouvant posséder des arcs multiples. On peut remarquer que dans un tel graphe, il n'existe pas de chemins passant deux fois par un même sommet. On appelle *double-chemin* entre le sommet u et le sommet v, une paire de deux chemins distincts reliant u à v. On dit qu'un double-chemin est *élémentaire* si les deux chemins n'ont aucun sommet en commun en dehors de u et de v. Soit D un double-chemin de G entre u et v, on notera P_1 et P_2 les deux chemins qui le composent. Si D n'est pas élémentaire, il est possible que ces deux chemins aient des arcs en communs, dans ce cas, ces arcs seront présents deux fois dans D.

La figure ci-contre donne un exemple d'un tel graphe acyclique. On peut remarquer qu'il possède une paire d'arcs multiples a_5 et a_6. Entre les sommets u_1 et u_2, on peut remarquer qu'il existe deux chemins (a_1, a_2) et (a_3, a_4) qui forment donc un double-chemin élémentaire. On peut aussi remarquer que les chemins (a_5, a_7) et (a_6, a_8, a_2) forment un double-chemin entre u_3 et u_2 qui n'est pas élémentaire. Le double chemin formé par (a_5, a_7) et (a_6, a_7) n'est pas élémentaire et l'arc a_7 est compté deux fois.



Etant donnée une fonction $w : A \to \mathbb{R}$ qui associe à tout arc $a \in A$ un poids $w(a)$, le *problème du sous-graphe sans double-chemin* (PSDC) consiste à déterminer un ensemble $B \subset A$ d'arcs tel que le sous-graphe $H = (V, B)$ ne contienne pas de double-chemin et tel que $w(B) = \sum_{a \in B} w(a)$ soit maximum.

Partie 1 : *Modélisation*

On considère le problème suivant intervenant dans la gestion des domaines skiables en haute-montage.
Un domaine skiable est un ensemble de pistes inclinées. Toutes les pistes commencent en altitude (en haut des remontées mécaniques) et se terminent à la sortie du domaine dans la vallée. Les pistes se croisent les unes les autres et peuvent même posséder des tronçons communs. On désire connaître où sont passés les skieurs dans une journée afin de prévoir l'usure des pistes, améliorer la signalisation etc. Pour cela, on munit chaque skieur d'un badge magnétique qui est détectable par des bornes. En plaçant des capteurs sur chaque tronçon du domaine, on peut alors analyser le parcours d'un skieur, c'est-à-dire reconstituer son parcours.
Une solution possible serait de positionner un capteur sur chaque tronçon de pistes du domaine. On désire réduire le nombre de capteurs positionnés tout en conservant la possibilité de reconstituer le parcours de chaque skieur.
Posons $G = (V, A)$ le graphe obtenu en plaçant un sommet à chaque croisement de pistes et un arc pour désigner le tronçon de piste allant d'un croisement à un autre (dans le sens de la descente).

Question 1 (2/8) — Montrer que le problème de gestion du domaine skiable en plaçant un nombre minimum de capteurs est équivalent au problème du sous-graphe sans double-chemin.

Partie 2 : *Formulation*

Soit $G = (V, A)$ un graphe acyclique (pouvant contenir des arcs multiples). On note $\mathcal{DC}$ l'ensemble des doubles-chemins de G.
Question 2 (1/8) — Prouver que le problème du sous-graphe sans double-chemin est équivalent au programme en nombres entiers (P) suivant

$$(P)\begin{cases} Max \quad \displaystyle\sum_{uv\in A} w_{uv}x(uv) & & \\ \qquad x(D) \le |D| - 1, & \text{pour tout double-chemin } D \in \mathcal{DC}, & (1)\\ \qquad 0 \le x(uv), & \forall uv \in A & (2)\\ \qquad x(uv) \ge 1, & \forall uv \in A & (3)\\ \qquad x(uv) \text{ entier}, & \forall uv \in A. & (4) \end{cases}$$

On appelle *contraintes de double-chemin* les contraintes de type (1) et triviales celle de type (2-3).
Question 3 (0, 5/8) — Quels sont les caractéristiques de ce programme en terme de nombres de contraintes et de variables ? Quelles sont les possibilités pour résoudre un tel programme ?

Partie 3 : *Séparation des contraintes de double-chemin*

Soit $D \in \mathcal{DC}$ un double-chemin entre u_0 et v_0, composé des deux chemins P_1 et P_2. On remarque que la contrainte (1) associée à A, i.e. $x(D) \leq |D| - 1$ s'écrit en fait

$$\sum_{uv \in P_1} x(uv) + \sum_{uv \in P_2} x(uv) \leq |P_1| + |P_2| - 1.$$

On considère à présent la classe $\mathcal{C}$ des contraintes (1) associées à des double-chemins élémentaires. Le problème de séparation *Sep* associé à cette classe peut s'énoncer ainsi :

Sep Etant donnée une solution (fractionnaire) x^*, déterminer s'il existe un double-chemin élémentaire D tel que $x^*(D) > |D| - 1$ et, dans le cas où il en existe un, produire un tel double-chemin D.

On associe à chaque arc uv de G la valeur $w^*(uv) = 1 - x^*(uv)$. Etant donnés deux sommets u_0 et v_0, on considère le problème suivant défini à partir de u_0 et v_0.

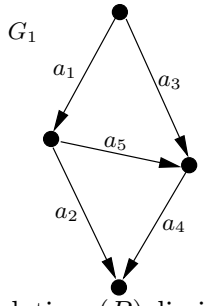
$T(u_0, v_0)$ Déterminer un plus court chemin P_1 de u_0 à v_0 dans G selon le poids w^*. Déterminer ensuite un deuxième chemin P_2 de u_0 à v_0 selon le poids w^* dans le graphe G privé des sommets intérieurs du chemin P_1.

Question 4 (1/8) — Montrer que le problème *Sep* revient à résoudre $n(n-1)$ fois le problème $T(u_0, v_0)$.

Question 5 (0,5/8) — En déduire que la séparation des contraintes de $\mathcal{C}$ est polynomial. Expliquer brièvement comment résoudre le programme (P).

Partie 4 : *Point fractionnaire et coupe*

Considérons le graphe $G_1 = (V_1, A_1)$ de la figure ci-contre. Considérons la solution y^* donnée par $y^*(a_1) = y^*(a_4) = 1$ et $y^*(a_2) = y^*(a_3) = y^*(a_5) = \frac{1}{2}$.



Question 6 (1,5/8) — Enumérez les contraintes de la formulation (P) limitée aux double-chemins élémentaires. Préciser les contraintes qui sont serrées par le point y^*. En admettant l'indépendance linéaire des contraintes obtenues, expliquez brièvement pourquoi y^* est un point extrême fractionnaire de cette formulation.

Question 7 (1/8) — On considère la nouvelle contrainte suivante. Soit deux points u et v. Soit P_1, P_2,... P_k k chemins reliant u à v. On note D^k l'ensemble $P_1, P_2, ..., P_k$ que l'on appelle *un k-chemin* entre u et v. Prouver que la contrainte

$$x(D^k) \leq |D^k| - (k-1) \tag{5}$$

est valide pour $P_{DC}(G)$.

Question 8 (0,5/8) — Montrer que cette contrainte coupe le point y^* dans le cas du graphe G_1. Expliquer pourquoi l'ajout de la contrainte à la formulation est utile à la résolution.

Exercice 3 (6 points)

Unit Commitment Problem (UCP)

On considère le problème de planification de production d'énergie appelé *Unit Commitment Problem* (UCP). On considère ici une version simplifiée pour l'exercice du problème classique.

La période de planification $\{1, \ldots, T\}$ consiste en T *pas de temps* d'une demi-heure chacun pour lesquels on connaît à l'avance la prévision de production électrique consommée : on appelle D_t la demande en kWatt/h à produire sur le pas de temps $t \in \{1, \ldots, T\}$. Le problème consiste à planifier quelles unités de production doivent être en marche à chaque pas de temps de la période.

On considère n unités de production d'énergie électrique (hydro-électriques, centrales thermiques,...). Chaque unité est soit dans un état "on" (en marche), soit dans un état "off" (non en marche). L'objectif du problème est de décider si une unité est "on" ou "off" pour chacun des pas de temps de la période. Chaque centrale $i \in \{1, \ldots, n\}$ est caractérisée par les données suivantes :
- la production g_i d'électricité (en kWatt/h) qui est produite sur un pas de temps par l'unité si elle est "on" (si elle est off, elle ne produit rien sur tout le pas de temps)
- le nombre minimal de pas de temps l_i de repos : c'est-à-dire que si l'unité est on au pas de temps t et que l'on désire qu'elle soit "off" en $t+1$ alors elle doit rester "off" de $t+1$ à $t+l_i$ avant de pouvoir être éventuellement mis "on" en $t+l_i+1$
- un coût de production c_i fixe par pas de temps si elle est on sur un pas de temps.

Le problème UCP consiste à décider l'état on/off de chacune de unités, en minimisant le coût total de production, en respectant la contrainte de repos et en satisfaisant le fait que le total, pour chaque pas de temps, de la production soit supérieure ou égale à la demande.

Question 1 (1/6) — Si $T = 1$, à quel problème très célèbre est équivalent le problème UCP ? En déduire que le problème UCP est NP-difficile.

Pour une unité $i \in \{1, \ldots, n\}$, on considère l'ensemble $\mathcal{P}^i$ des planifications on/off possibles : c'est-à-dire la donnée pour chaque pas de temps du fait que l'unité est on ou off. On note alors $\mathcal{P}$ l'union de tous les ensembles $\mathcal{P}^i$, $i \in \{1, \ldots, n\}$. On associe à chaque planification $p \in \mathcal{P}$ une variable entière 0-1 x_p correspondant au fait que cette planification sera retenue ou non dans la solution.

Question 2 (1/6) — Donner le coût $c(p)$ total pour une planification $p \in \mathcal{P}$.
Question 3 (0,5/6) — Montrer que la formulation PLNE (F) suivante est équivalente

au problème UCP.

$$(F)\begin{cases} Min & \sum_{p\in\mathcal{P}} c(p)x_p & & \\ & \sum_{p\in\mathcal{P}^i} x_p = 1 & \forall i\in\{1,\dots,n\} & (1) \\ \sum_{i=1}^{n} & \sum_{\substack{p\in\mathcal{P}^i \\ \text{t.q. } i \text{ est "on" en } t}} g_i x_p \geq D_t & \forall t\in\{1,\dots,T\} & (2) \\ & x_p \geq 0, & \forall p\in\mathcal{P}, & \\ & x_p \in I\!N & \forall p\in\mathcal{P}. & \end{cases}$$

Question 4 $(0,5/6)$ — Donner, dans le pire des cas, le nombre de variables de (F). Quelle solution algorithmique peut-on envisager pour le résoudre ?

Résoudre la relaxation linéaire de F

On appelle $(\tilde{F})$ la relaxation linéaire de (F). On note λ_i, $i\in\{1,...,n\}$ les variables duales associées aux inégalités (1) de la formulation $(\tilde{F})$ et μ_t, $t\in\{1,\dots,T\}$ les variables duales associées aux inégalités (2).

On admet que le programme linéaire dual $(\tilde{D})$ de $(\tilde{F})$ est le suivant :

$$(\tilde{D})\begin{cases} Max & \sum_{i=1}^{n}\lambda_i + \sum_{t=1}^{T}\mu_t & & \\ & \lambda_i + \sum_{\substack{t=1 \\ \text{t.q. } i \text{ est "on" en } t}}^{n} g_i\mu_t \leq c(p) & \forall i\in\{1,\dots,n\}, \forall p\in\mathcal{P}^i \\ & \lambda_i \text{ libre dans } I\!R & \forall i\in\{1,\dots,n\} & \\ & \mu_t \geq 0 & \forall t\in\{1,\dots,T\}. & \end{cases}$$

On suppose que l'on dispose d'une première solution réalisable pour le problème, c'est-à-dire un ensemble P' de planifications telle que la demande soit satisfaite à chaque pas de temps. On appelle alors $(\tilde{F'})$ le programme linéaire $(\tilde{F})$ restreint aux variables x_p correspondant aux planifications de F' et on note $x' = (x'_p)_{p\in P'}$ la solution optimale de $\tilde{F'}$. On étend alors t' à la solution t pour toute planification $p\in\mathcal{P}$ en posant $x_p = x'_p$ pour les planifications $p\in P'$ et $t_p = 0$ pour toutes les planifications $\mathcal{P}\setminus P'$.

Question 5 $(2/6)$ — En posant λ' la solution optimale du dual $\tilde{D'}$ de $\tilde{F'}$, donner une condition nécessaire et suffisante pour que x soit une solution optimale de $\tilde{F}$.
Question 6 $(1/6)$ — A partir de la question précédente, donnez l'énoncé du problème correspondant à la génération d'une colonne dans la formulation $\tilde{F}$.
Question 7 $(bonus/6)$ — Proposez une méthode polynomiale pour répondre au problème de génération de colonne de la question précédente.