OPTIMISATION ET RECHERCHE OPÉRATIONELLE *Janvier 2018*

Examen Final

Nom: Prenom: Numéro étudiant:

— Durée de l'examen : 1h30
— Document autorisé : une feuille de notes A4 recto-verso
— Cet examen comporte trois exercices indépendants pouvant être traités dans n'importe quel ordre. Les deux premiers sont assez courts tandis que le dernier est un problème comportant trois parties indépendantes elles-aussi.
— L'énoncé est à rendre avec la copie, n'oublier pas d'indiquer vos noms et prénoms sur toutes les feuilles.
— Toutes les réponses doivent être justifiées.
— Le barème donné est indicatif.

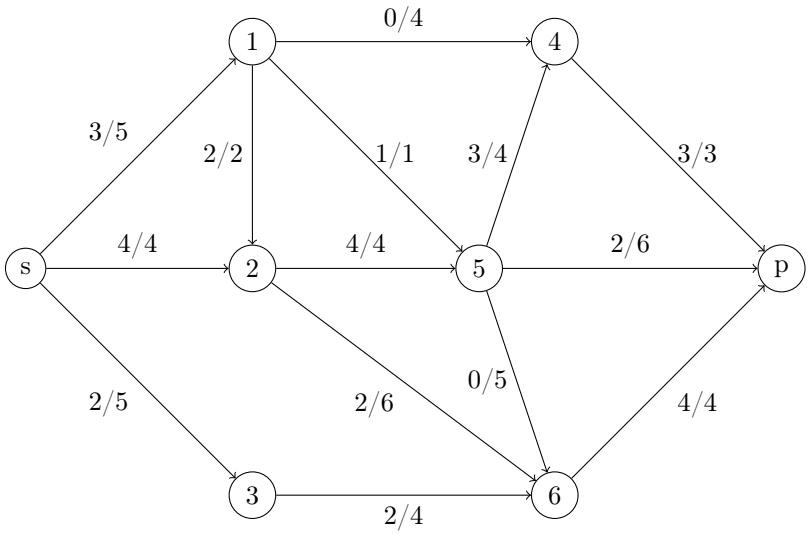
Exercice 1 : Question de cours (2 pts)

Donner le nom de trois algorithmes métaheuristiques vus en cours. Parmi ceux-ci, lesquels sont probabilistes ? déterministes ? Justifier.

Exercice 2 : Flots (4.5 pts)

Question 1. (2 pts) Le flot suivant est-il complet ? Est-il maximal ?

Justifiez vos réponses et si le flot n'est pas maximal, augmentez-le jusqu'à obtenir un flot maximal, et donnez sa valeur. Vous pouvez décrire le flot maximal sur la feuille directement (qui sera à rendre), mais votre réponse devra dans tous les cas être justifiée.



Question 2. (1 pt) Donner une coupe minimale.

Question 3. (1,5 pt) On souhaite désormais augmenter le flot total, et pour cela, nous avons la possibilité d'augmenter la capacité de l'arc $(2, 5)$. Cet arc est-il actuellement limitant (càd qu'une augmentation de sa capacité permettrait d'augmenter le flot total) ? Si oui, quel est la plus petite valeur de capacité entière qu'on peut donner à l'arc $(2, 5)$ pour qu'il ne soit plus limitant ? Dans ce cas, donnez la nouvelle valeur du flot. Justifiez toutes vos réponses.

Exercice 3 : (15.5 pts) *Les différentes parties de ce problème sont indépendantes et peuvent être traitées dans n'importe quel ordre. Vous pouvez aussi admettre une question pour répondre à la suivante.*

L'entreprise Regeot fabrique des voitures. Pour fabriquer les différents pièces, elle se pose plusieurs problèmes d'ordonnancement.

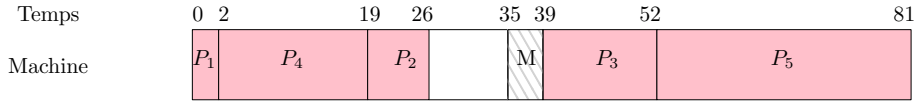
Partie I : Une machine avec un temps de maintenance (4.5 pts)

Dans un premier temps, on considère une seule machine qui doit fabriquer n pièces différentes dont la durée de fabrication varie. On notera d_i la durée de fabrication de la pièce i. Une opération de maintenance est prévue à une date fixe t_M et pendant une durée d_M. Pendant la maintenance, aucune pièce ne peut être fabriquée et l'on ne peut pas interrompre la fabrication d'une pièce au moment de la maintenance. On cherche à minimiser le temps final où toutes les pièces seront fabriquées. On considèrera que la fabrication des pièces commence à la date $t_0 = 0$.

Pour mieux comprendre le problème, voilà un exemple avec $n = 5$, $t_M = 35$ et $d_M = 4$. Les pièces à usiner ont les durées suivantes :

Pièces	1	2	3	4	5
Durée d_i	2	7	13	17	29

Un ordonnancement possible avec comme temps final 81 est le suivant (sur le dessin, M désigne la période de maintenance) :



Question 1. (1,5 pt) Modéliser ce problème sous forme d'un PLNE. On pourra remarquer qu'il suffit de décider pour chaque pièce si elle est effectuée avant ou après la maintenance.

On propose de résoudre ce problème avec le principe de programmation dynamique. Plutôt que de minimiser le temps final d'éxécution on va résoudre le problème équivalent consistant à maximer le temps de tâches effectuées avant la pause de maintenance. On pose donc $f(i, k)$ le temps maximal de travail qui peut être fait pendant une durée inférieure à k en fabriquant des pièces parmi les pièces 1 à i.

Le tableau suivant donne quelques valeurs de $f(i, k)$ pour les données de l'exemple précédent. Par exemple, $f(4, 18) = 17$ signifie que le temps maximal de travail qui peut être fait avec les tâches 1 à 4 sans dépasser 18 unités de temps est au maximum 17 (en l'occurence, atteint avec la tâche 4 seule).

$i\backslash k$	0	1	2	3	4	5	6	...	18	...	29	30	31	32	33	34	35
1 ($d_1 = 2$)	0	0	2	2	2	2	2	...	2	...	2	2	2	2	2	2	2
2 ($d_2 = 7$)	0	0	2	2	2	2	2	...	9	...	9	9	9	9	9	9	9
3 ($d_3 = 13$)	0	0	2	2	2	2	2	...	15	...	22	22	22	22	22	22	22
4 ($d_4 = 17$)	0	0	2	2	2	2	2	...	17	...	26	30	30	32	32	32	32
5 ($d_5 = 29$)	0	0	2	2	2	2	2	...	17	...							

Question 2. (0,5 pt) Pour quelles valeurs de i et k, le calcul de $f(i, k)$ résoud notre problème ?

Question 3. (1 pt) Donner une relation de récurrence pour $f(i, k)$ utilisant $f(i-1, k)$ et $f(i-1, k-d_i)$.

Question 4. (1,5 pt) Compléter le tableau (pour $i = 5$ et $29 \leq k \leq 35$) et en déduire une solution optimale pour le problème initial.

Partie II : Usinage de pièces sur m machines avec des précédences (7 pts)

Dans cette partie, l'entreprise doit toujours fabriquer n pièces mais dispose cette fois de m machines identiques. Les pièces doivent être réalisées en une seule fois sur une seule machine mais peuvent être réalisées indifféremment sur l'une ou l'autre des machines. On cherche encore à minimiser le temps nécessaire pour que toutes les pièces soient fabriquées. Pour chaque pièce P, on note $d(P)$ la durée de fabrication de la pièce P. On notera $\mathcal{P}$ l'ensemble des pièces.

Question 5. (1 pt) Rappeler la complexité algorithmique de ce problème. Existe-t-il un algorithme d'approximation, si oui, avec quel facteur ?

En fait, certaines pièces ont besoin que d'autres pièces soient finies pour pouvoir être fabriquées (mais pas nécessairement sur la même machine). Dans la suite de cette partie, on considèrera l'exemple suivant sur trois machines :

Pièce	Durée	Précédences
A	3	
B	2	
C	3	
D	4	
E	7	A
F	2	D
G	3	A,D
H	3	B,C,D
I	2	G,H
J	4	D,G,H

Question 6. (1 pt) Donner l'ordonnancement obtenu sur 3 machines en suivant l'algorithme de liste avec la liste $\{A, B, C, D, ...\}$. Pour rappel, cela consiste à éxécuter au temps t (en partant à $t = 0$, la plus petite tâche disponible selon l'ordre donné par la liste.

Nous allons montrer que n'importe quel algorithme de liste fournit un algorithme d'approximation avec un facteur $2 - 1/m$ de l'optimal. Soit I une instance quelconque de notre problème. Soit $opt(I)$ le temps minimal que l'on peut obtenir parmi tous les ordonnancements possibles et $sol(I)$ la solution donnée par un algorithme de liste quelconque. Soit P_ℓ la pièce dont la fabrication se termine en dernière. On note $t(P_\ell)$ la date à laquelle la pièce a commencée à être fabriquée et $d(P_\ell)$ sa durée.

Question 7. (0,5 pt) Exprimer $sol(I)$ en fonction de $t(P_\ell)$ et $d(P_\ell)$.

Soit $P_{\ell-1}$ la pièce qui est terminée en dernière parmi toutes les pièces qui doivent être faites avant P_ℓ. De la même manière, on définit $P_{\ell-2}$,...,P_s avec P_i qui est la pièce terminée en dernière parmi toutes les pièces qui doivent être faites avant P_{i+1} et P_s qui n'a aucune pièce qui doit être finie avant elle. Il y a ainsi une chaîne de dépendances $P_s \to P_{s+1} ... \to P_\ell$.

Question 8. (0,5 pt) Sur l'ordonnancement obtenu à la question 6, donner les noms des pièces correspondant à P_ℓ, $P_{\ell-1}$,...,P_s.

Question 9. (1 pt) Montrer que $\sum_{i=s}^{\ell} d(P_i) \leq opt(I)$ et $\sum_{P\in\mathcal{P}} d(P) \leq m \times opt(I)$.

Question 10. (1 pt) Expliquer pourquoi entre les dates $t(P_i)+d(P_i)$ et $t(P_{i+1})$ (pour $i \in \{s, ..., \ell-1\}$), ainsi qu'avant la date $t(P_s)$ toutes les machines sont occupées, et occupées à d'autres tâches que les tâches de la chaîne $P_s, ..., P_\ell$. En déduire l'inégalité suivante :

$$m \times \left(t(P_s) + \sum_{i=s}^{\ell-1} [t(P_{i+1}) - (t(P_i) + d(P_i))] \right) \leq \sum_{P\in\mathcal{P}} d(P) - \sum_{i=s}^{\ell} d(P_i)$$

Question 11. (1 pt) En utilisant les questions 7,9 et 10, en déduire que l'on a bien une $(2 - 1/m)$-approximation, c'est-à-dire que $sol(I) \leq (2 - 1/m) \times opt(I)$.

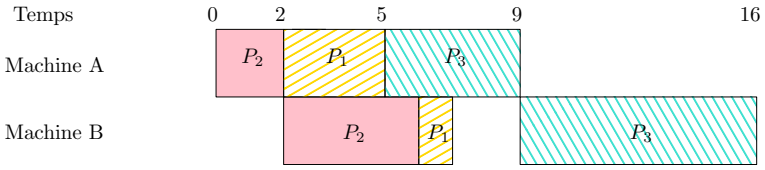
Question 12. (1 pt) Donner un exemple général sur m machines où le facteur $2 - 1/m$ est atteint.

Partie III : Ordonnancement sur 2 machines de tâches séquentielles `(4 pts)`

Dans cette partie, l'entreprise doit usiner n pièces qui doivent être fabriquée par deux machines, A et B séquentiellement : d'abord sur la machine M_A puis sur la machine M_B. Une fois l'ordre choisi sur la machine A, **les pièces doivent être réalisées suivant le même ordre sur la machine** B. On dispose comme données des durées de fabrication $d_A(P_i)$ et $d_B(P_i)$ de chaque pièce P_i sur les deux machines. Encore une fois, le but est de minimiser le temps final où toutes les pièces sont éxécutées.

Voilà un exemple de données. En usinant les pièces dans l'ordre 2,1,3 on obtient un ordonnancement qui termine à la date 16.

Pièces	1	2	3
Durée d_A	3	2	4
Durée d_B	1	4	7



Question 13. `(1 pt)` Cette solution est-elle optimale ? Justifiez pourquoi.

L'algorithme de Johnson fournit un algorithme polynomial pour ce problème. Il procède ainsi :

1. Soit S l'ensemble des pièces P_i telles que $d_A(P_i) \leq d_B(P_i)$.
2. Soit T les autres pièces.
3. Ordonnancer les pièces dans l'ordre suivant : d'abord les pièces de S par d_A croissante, puis les pièces de T par d_B décroissante.

Question 14. `(1.5 pt)` Appliquer cet algorithme sur l'instance suivante. On donnera les ensembles S et T et l'ordonnancement final obtenu.

Pièces	1	2	3	4	5
Durée d_A	4	1	3	5	2
Durée d_B	1	3	2	3	3

Pour justifier son algorithme, Johnson a prouvé que pour obtenir l'ordonnancement optimal, il suffit de mettre la pièce P_i avant la pièce P_j si l'inégalité suivante est vérifiée

$$\min(d_A(P_i), d_B(P_j)) \leq \min(d_B(P_i), d_A(P_j)) \tag{1}$$

Question 15. `(1.5 pts)` Montrer que, pour une instance quelconque, l'ordonnancement obtenu avec l'algorithme de Johnson vérifie bien cette propriété. C'est-à-dire que pour toutes pièces P_i et P_j l'inégalité (1) est vérifiée. On pourra séparer les cas suivant si P_i et P_j sont tous les deux dans S ou bien tous les deux dans T ou bien un dans chaque ensemble.