Algorithmique - ROB3
TD3 - Algorithmes gloutons

Exercice 1 (Le coût de la panne sèche) *Le professeur Midas conduit une voiture entre Amsterdam et Lisbonne sur l'européenne E10. Son réservoir d'essence plein lui permet de parcourir n kilomètres, et sa carte lui donne les distances entre les m stations-service sur la route. Le professeur souhaite faire le moins d'arrêts possible pendant le voyage.*

Question 1.1 *Donner une méthode efficace permettant au professeur Midas de déterminer les stations-service où il peut s'arrêter.*

Question 1.2 *Montrer que cette méthode aboutit à une solution optimale.*

Question 1.3 *Quelle est la complexité de cette méthode ?*

Exercice 2 (Propagation de rumeurs) *On considère un groupe de n personnes, chacune d'entre elles au courant d'une rumeur différente. Ces personnes souhaitent partager l'ensemble des rumeurs les unes avec les autres en s'envoyant des messages électroniques. On suppose qu'un expéditeur peut inclure toutes les rumeurs dont il/elle est au courant dans un message, mais qu'un message ne peut avoir qu'*un unique destinataire. *Le but est de propager toutes les rumeurs en un nombre minimum de messages.*

Question 2.1 *Donner une méthode efficace permettant de minimiser le nombre de messages.*

Question 2.2 *Montrer que cette méthode aboutit à une solution optimale, sans chercher à démontrer la propriété du choix glouton ni la propriété de sous-structure optimale.*

Question 2.3 *Quelle est la complexité de cette méthode en nombre d'envois de messages ?*

Exercice 3 (Deviner la valeur d'une carte) *On cherche à concevoir une stratégie pour minimiser l'espérance du nombre de questions posées dans le jeu suivant. On dispose d'un paquet de 45 cartes composé d'une carte de valeur 1, de deux cartes de valeur 2, de trois cartes 3, et ainsi de suite jusqu'à neuf 9. Quelqu'un tire une carte au hasard dans le paquet mélangé, et on cherche à identifier la valeur de la carte en posant des questions dont la réponse est oui ou non.*

Question 3.1 *Proposer une méthode optimale, et indiquer l'espérance du nombre de questions posées.*

Exercice 4 (Ordonnancement de tâches sur une machine) *On s'intéresse au problème d'ordonnancement de tâches sur une machine qui consiste à définir les dates de début d'exécution d'un ensemble de n tâches J_i, $i = 1, \cdots, n$ caractérisées par une durée opératoire p_i et une date d'échéance d_i à laquelle la tâche devrait idéalement être terminée. La machine ne peut exécuter qu'une tâche à la fois. Si l'on note s_i la date de début d'exécution de la tâche J_i, le retard de J_i sera défini par $l_i = \max\{0, s_i + p_i - d_i\}$. Si $l_i > 0$, la tâche J_i est en retard.*

Le but de l'exercice est de déterminer un ordonnancement minimisant la valeur du retard maximal $L = \max_i l_i$ pour l'ensemble des tâches.

Question 4.1 *Déterminer la valeur de L pour un ensemble de $n = 6$ tâches pour les données suivantes : $p = (3, 2, 1, 4, 3, 2)$, $d = (6, 8, 9, 9, 14, 15)$ et $s = (5, 1, 0, 11, 8, 3)$.*

Question 4.2 *Différents algorithmes gloutons peuvent être décrits selon le choix de l'ordre fixé sur les tâches pour leur exécution. Proposer un contre-exemple pour montrer que l'ordre des durées opératoires croissantes n'est pas optimal.*

Question 4.3 *Les dates d'échéance doivent donc être prises en compte pour définir un ordre sur les tâches. On définit l'urgence de la tâche J_i comme étant la valeur $d_i - p_i$ et on s'intéresse à l'algorithme glouton qui ordonnance les tâches selon l'ordre des urgences croissantes. Cet algorithme est-il optimal ?*

Question 4.4 *Montrer que les ordonnancements ne contenant aucune période d'inactivité pour la machine sont dominants.*

On considère l'algorithme glouton suivant que l'on appellera EDD :

Ré-indexer les tâches de façon à ce que $d_1 \leq d_2 \leq \cdots d_n$
$t \leftarrow 0$
pour $i = 1$ ***à*** n ***faire***
  Affecter la tâche J_i à l'intervalle $[t, t + p_i]$
  $s_i \leftarrow t$
  $C_i \leftarrow s_i + p_i$
  $t \leftarrow C_i$
fin pour
pour $i = 1$ ***à*** n ***faire***
  Afficher les intervalles $[s_i, C_i]$
fin pour

Question 4.5 *Dérouler l'algorithme EDD sur l'exemple de la question 1 et calculer la valeur du retard maximal pour la solution trouvée.*

On suppose dans la suite que les tâches sont indexées dans l'ordre de leurs dates d'échéance croissantes. Etant donné un ordonnancement, une inversion est une paire de tâches J_i et J_j pour laquelle $d_i < d_j$ et J_j est ordonnancée avant J_i.

Question 4.6 *Montrons que la permutation d'une paire de tâches inversées consécutives dans un ordonnancement n'entraîne pas l'augmentation de son retard maximal.*

Question 4.7 *Montrer qu'il existe un ordonnancement optimal ne possédant ni inversion, ni temps d'inactivité.*

Question 4.8 *Que peut-on dire de la valeur de retard maximal pour les ordonnancements ne possédant pas d'inversions ni de temps d'inactivité ?*

Question 4.9 *Montrer que l'ordonnancement renvoyé par l'algorithme EDD est optimal.*

TD 3

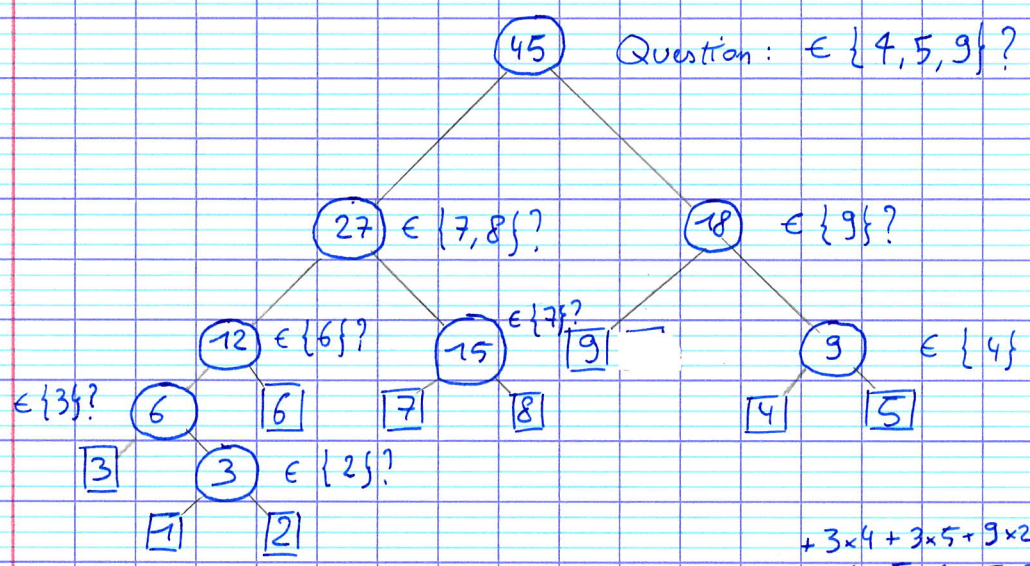
Exercice 3

Le paquet contient : $1 \times \boxed{1} + 2 \times \boxed{2} + \dots + 9 \times \boxed{9}$

$= \sum_{i=1}^{9} i = \frac{9 \times 10}{2} = 45$ cartes.

On veut minimiser : $\sum_{i=1}^{9} \frac{i}{45} Q(i)$ où $Q(i)$ est le nombre de questions nécessaires pour identifier la carte i.

On construit une arborescence (cf Huffman).

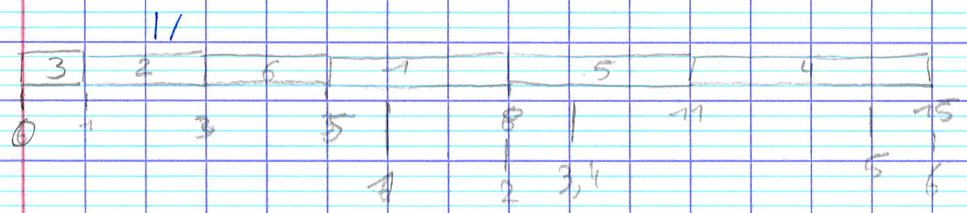


Espérance du nombre de question : $\frac{+3\times4+3\times5+9\times2}{3\times6+5\times1+5\times2+4\times3+3\times7+3\times8}$

$= 3$

Exercice 4

1/



$l = (2, 0, 0, 6, 0, 0) \Rightarrow L = \max_i (l_i) = 6$

2/

Prenons : $p = (1, 5)$ $d = (7, 5)$ ~~$s = 17$~~

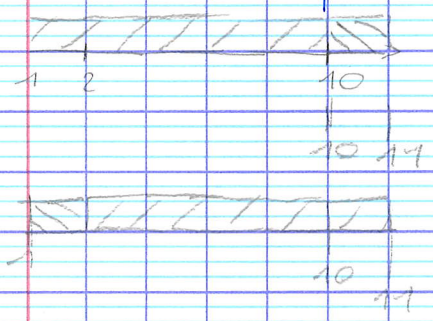
Si on ordonne par durée croissante : $l = (0, 1)$ $L = 1$

L'inverse donne $\ell = (0,0) \rightarrow L = 0$.

3/

Prenons $p = (1, 10)$ $d = (10, 11)$

$d - p = (9, 1)$



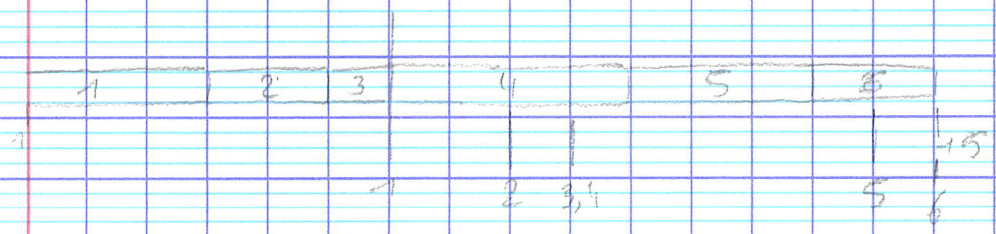
$\ell = (1, 0) \rightarrow L = 1$

$\ell = (0, 0) \rightarrow L = 0$

4/

Tout ordonnancement peut être "calé à gauche" pour obtenir un ordonnancement sans temps d'inactivité dont le retard max est au moins aussi bon.

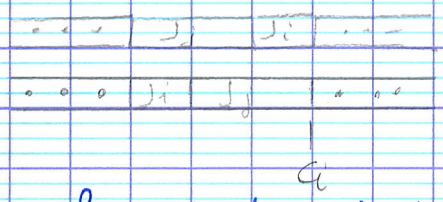
5/



$\ell = (0, 0, 0, 1, 0, 0) \rightarrow L = 1$

6/

$d_j > d_i$



Après l'échange, les retards de toutes les tâches autres que i et j restent inchangés.

$\ell_i = \max\{0, C_i - d_i\} \geq \ell_i$

$l_j' = \max\{1, C_i - d_j\}$ or $d_j > d_i \Rightarrow C_i - d_j < C_i - d_i$

Donc $l_i \geq l_j'$

Donc $\max\{l_i, l_j\} \geq \max\{l_i', l_j'\}$ car $l_i' \leq l_i$ et $l_j' \leq l_i$

7/

Si il y a une inversion dans un ordre, alors il y a nécessairement une inversion entre éléments consécutifs. Soit O un ordonnancement optimal. En calant à gauche les tâches, on peut obtenir un ordonnancement sans temps d'inactivité. En itérant des échanges de tâches entre tâches inversées consécutives, on obtient un ordonnancement optimal sans inversion ni temps d'inactivité.

8/

Si plusieurs tâches partagent la m̂ date de fin d'échéance, alors il n'y a pas un unique ordonnancement sans inversions des temps d'inactivité.

Néanmoins, dans un tel ordonnancement, parmi les tâches partageant la même date d'échéance, c'est la dernière tâche exécutée qui a le retard max. Ce retard max reste inchangé quel que soit l'ordre d'exécution des tâches de même date d'échéances (elles sont toutes consécutives au m̂ endroit dans l'ordonnancement).

Donc toutes les ordonnancements sans inversion ni temps d'inactivité ont le même retard max qui est optimal.

9/

L'algorithme EDD retourne un ordonnancement sans temps d'inactivité ni inversion donc un ordonnancement optimal.