

Ordonnancement sur une machine

machine fait qu'une chose à la fois.

tâche $i \rightarrow$ deadline d_i

\rightarrow durée p_i

si la tâche i se termine à T_i , son retard $T_i = \max(T_i - d_i, 0)$

On veut choisir un ordre de passage sur la machine

pour minimiser $\sum_{i=1}^n T_i^2$ plus le retard est grand plus c'est pénalisant.

Les tâches doivent s'enchaîner dans la machine.

	1	2	3	4	5	6
p_i	3	4	2	5	3	1
d_i	5	5	7	8	10	10

Problème de minimisation:

solution, on ordonne en fonction de la deadline \nearrow

Calcul du majorant:	3	7	3	14	17	18
	1	2	3	4	5	6
	↓	↓	↓	↓	↓	↓

$$4 + 4 + 36 + 49 +$$

Propriété de dominance:

Pour tout ordonnancement optimal la date de fin sera: $\sum_{i=1}^n p_i$ (18 donc)
ceci est mieux.

A	B	
---	---	--

Le mieux est que la machine ne s'arrête jamais, comme ici.

que ceci:

A		B
---	--	---

il y a une tâche j qui se termine à 18
on peut calculer son retard, & minorant. Parmi les deadlines, le max des d_i est 10, donc:

cette tâche a un retard de $18 - \underline{10} = 8$

10 dans notre tableau

$$T_j^2 = 8^2 = 64$$

Le retard est entre: $64 \leq T_i^2 \leq 157$

Noeud S est défini par une séquence $J(S)$ qui termine tous les ordonnancements du noeud S .

$$S \rightarrow J(S) = \{3, 6\} \xrightarrow{\text{choix aléatoire}} \begin{array}{|c|c|} \hline & 3 & 6 \\ \hline \end{array}$$

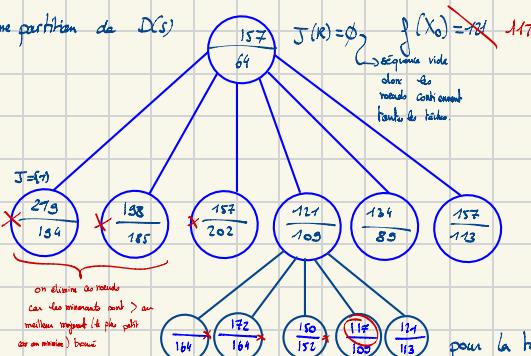
$$R \rightarrow J(R) = \emptyset$$

• Séparation d'un noeud S

S a auant de fils qu'il n'y a de tâches : $\notin J(S)$

$$\text{fils : } S_i \quad J(S_i) = \{i, J(S)\}$$

$D(S)$ fournit une partition de $D(S)$



[S - 4]

Evaluation d'un noeud S

Majorant \rightarrow ordonner les autres tâches par $d_i \nearrow$ ayant $J(S)$

	$J(S)$
--	--------

minorant \rightarrow $\max_{i \in J(S)} d_i$

	J	$J(S)$
	$\sum_{i=1}^n p_i - \sum_{i \notin J(S)} p_i$	

$$T_j \geq \sum_{i \notin J(S)} p_i - \max_{i \in J(S)} d_i$$

$$\sum_{i \in J(S)} t_i^2 + \left(\max_{i \in J(S)} \left(\sum_{i \notin J(S)} p_i - \max_{i \in J(S)} d_i \right) \right)^2$$

1	2	3	5	6	13	78
\downarrow						

$$\text{pour la tâche } 1, J = \{1\} \text{ donc } 1 \text{ est majorant à } j_1$$

$$\text{pour la tâche } 6, J = \{1, 2, 3, 4, 5\} \text{ donc } 6 \text{ est minorant à } j_6$$

4	6	11	14	15	18
\downarrow	\downarrow	\downarrow	\downarrow	\downarrow	\downarrow
2	3	4	5	6	7
\downarrow	\downarrow	\downarrow	\downarrow	\downarrow	\downarrow
3	6	10	15	25	
\downarrow	\downarrow	\downarrow	\downarrow	\downarrow	\downarrow
3	16	25			

Pour le majorant :
on fait la diff entre
un di et sa durée p_i

$$\text{majorant } 3+16+25+165 = 219$$

$$\text{minorant } 25 + (18-d_i)$$

$$25 + 165 = 194$$

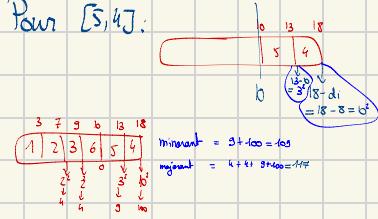
on met nulle si = la fin.

jeu de minorant on calcule (\exists) - sa durée + le plus grand di sans compter celui de nœud tâche où il n'est plus dans la séquence, on prend en compte

que les bouches dans la séquence, donc pas dans J . Ce qui est pas dans J est dans la séquence.

Si le minorant est supérieur à 121 ($J \neq \emptyset$) on élimine les nœuds entier.

Pour $[5, 14]$:



Exercice 2.4 :

question 1:

Contraintes

1) Stocker une fois

$$x_{i1} + y_{i1} \leq 1 \text{ pour tout } i \in \{1, \dots, n\}$$

2) Capacité des disques

$$\begin{aligned} \sum_{i=1}^n x_{i1} l_i &\leq M \\ \sum_{i=1}^n y_{i1} l_i &\leq M \end{aligned}$$

3) Variables binaires

$$x_{i1} \in \{0, 1\}$$

$$y_{i1} \in \{0, 1\}$$

Objectif

$$\max \sum_{i=1}^n x_{i1} + y_{i1}$$

question 2:

trier les fichiers par taille

les mettre un à un jusqu'à ce qu'aucun ne rentre.

question 3

Au nœud S

$F(S)$ $R(S)$ $D_1(S)$ et $D_2(S)$ \Rightarrow fichier déjà sur les disques.

Equivaut entre deux critères

$$\max \sum_{i=1}^n x_{i1} + y_{i1}$$

$$\bullet x_{i1} + y_{i1} \leq 1 \quad i \in F(S)$$

$$\bullet \text{si } i \in R(S) \quad x_{i1} = y_{i1} = 0$$

$$\bullet \text{si } i \in D_1(S) \quad x_{i1} = 1 \text{ et } y_{i1} = 0$$

$$\bullet \text{si } i \in D_2(S) \quad x_{i1} = 0 \text{ et } y_{i1} = 1$$

$$\max \sum_{i \in F(S)} x_{i1} + y_{i1} + |D_1(S)| + |D_2(S)|$$

$$\bullet x_{i1} + y_{i1} \leq 1 \quad i \in F(S)$$

$$\bullet \sum_{i \in F(S)} x_{i1} \leq M_1(S)$$

$$\bullet \sum_{i \in F(S)} y_{i1} \leq M_2(S)$$

Exercice 2.4 Stockage de fichiers

On considère un ensemble de fichiers $\mathcal{F} = \{1, \dots, n\}$ de tailles respectives (en octets) l_1, \dots, l_n . On dispose de deux disques durs de taille M (en octets).

On cherche à stocker un nombre maximum de fichiers, sachant qu'un fichier ne peut pas être découpé entre les deux disques, et qu'il est possible qu'il ne soit pas faisable de les stocker tous.

Les applications numériques se feront sur l'exemple suivant : $n = 5$, $M = 4$, $l_2 = l_5 = 1$, $l_1 = l_4 = 2$, $l_3 = 3$.

Question 1 : En notant x_i la variable de décision qui vaut 1 si le fichier est stocké sur le premier disque, 0 sinon, et y_i la variable de décision qui vaut 1 si le fichier est stocké sur le second disque, et 0 sinon, modéliser le problème à l'aide d'un programme linéaire à variables bivalentes.

On s'intéresse maintenant à la définition d'une méthode arborescente pour résoudre ce problème.

Question 2 : On considère le problème suivant : étant donné un seul disque de taille C , quel est le nombre maximum de fichiers de \mathcal{F} pouvant être stockés sur ce disque ? Indiquer un algorithme permettant de résoudre ce problème.

Question 3 : A chaque nœud S de l'arborescence seront associés quatre sous-ensembles de fichiers disjoints : $R(S)$ l'ensemble des fichiers rejettés, $D_1(S)$ l'ensemble des fichiers affectés au disque 1, $D_2(S)$ l'ensemble des fichiers affectés au disque 2, et $F(S)$ l'ensemble des autres fichiers : $F(S) = \mathcal{F} \setminus (R(S) \cup D_1(S) \cup D_2(S))$. On supposera que $M_1(S) = M - \sum_{i \in D_1(S)} l_i \geq 0$ et $M_2(S) = M - \sum_{i \in D_2(S)} l_i \geq 0$.

Notons $X(S)$ le nombre maximum de fichiers parmi $F(S)$ pouvant être stockés sur un disque de taille $M_1(S) + M_2(S)$.

Montrer que $H(S) = X(S) + |D_1(S) \cup D_2(S)|$ est une évaluation par excès de la fonction objectif au nœud S .

Prouver la récursion.

Comment faire la relaxation?

On combine les deux contraintes, on fait ainsi une relaxation du précédent programme linéaire.

$$\begin{aligned} \text{Max } & \sum z_i g_i + |D_1(S)| + |D_2(S)| \\ & \sum_{i \in F(S)} x_i + y_i \leq 1 \quad i \in F(S) \\ & \sum_{i \in F(S)} b_i (x_i + y_i) \leq \Gamma_1(S) + \Gamma_2(S) \\ & x_i, y_i \in \{0,1\} \end{aligned}$$

on fait min
un changement de variable
 $\underline{z_i = x_i + y_i}$

$$\begin{aligned} \text{Max } & \sum z_i + |D_1(S)| + |D_2(S)| \\ & \sum_{i \in F(S)} z_i \leq \Gamma_1(S) + \Gamma_2(S) \quad (\Rightarrow) \\ & x_i, y_i \in \{0,1\} \end{aligned}$$

$$\begin{aligned} \text{Max } & \sum_{i \in F(S)} z_i + |D_1(S)| + |D_2(S)| \\ & \sum_{i \in F(S)} z_i \leq \Gamma_1(S) + \Gamma_2(S) \\ & z_i \in \{0,1\} \end{aligned}$$

on peut résoudre ça avec la question 2.

On vient de prouver que ceci est une relaxation.

question 4:

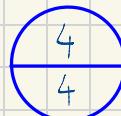
Ici on maximise, donc l'évaluation par défaut se fait sur le minimum, qui est une solution particulière.

$$n = 5, M = 4$$

$$\begin{matrix} i & 1 & 2 & 3 & 4 & 5 \\ f_i & 2 & 1 & 3 & 2 & 1 \end{matrix}$$

2 disques de taille 4.

	2	4	6	
max	2	5	1	4
min	2	5	1	
D1	2	5	1	
D2	4			



Question 4 : On considère l'algorithme suivant, permettant de calculer une solution associée au noeud S :

- classer les fichiers de $F(S)$ par ordre de taille croissante.
- Affecter les premiers au disque 1, jusqu'à hauteur de $M_1(S)$ octets maximum.
- Affecter les suivants au disque 2 jusqu'à hauteur de $M_2(S)$ maximum.
- Calculer pour l'exemple numérique la solution ci-dessus et l'évaluation par excès de la question précédente associée à la racine de l'arbre (noeud correspondant à l'ensemble de toutes les solutions).

Question 5 : La séparation d'un noeud S consiste à choisir un fichier $j \in F(S)$, et à générer au plus trois fils S', S'', S''' avec :

- $R(S') = R(S) \cup \{j\}$, $D_1(S') = D_1(S)$, $D_2(S') = D_1(S)$, $F(S') = F(S) \setminus \{j\}$
- lorsque $l_j \leq M_1(S)$: $R(S'') = R(S)$, $D_1(S'') = D_1(S) \cup \{j\}$, $D_2(S'') = D_1(S)$, $F(S'') = F(S) \setminus \{j\}$
- lorsque $l_j \leq M_2(S)$: $R(S''') = R(S)$, $D_1(S''') = D_1(S)$, $D_2(S''') = D_1(S) \cup \{j\}$, $F(S''') = F(S) \setminus \{j\}$

Appliquer la méthode arborescente ainsi définie à l'exemple, en précisant les règles de parcours de l'arborescence choisies, les évaluations des noeuds, et les troncatures effectuées.