Minimisation du coût maximal

$1 \| - \| \max_i g_i(C_i)$

$J_1, J_2, \ldots, J_n$ tâches / p_i - durée opératoire J_i $i = 1, \ldots, n$
non-preemptif
$\text{Min} \max \{ g_i(C_i), i = 1, \ldots, n \}$ g_i - fonction croissante

1) On pose $P = \sum_{i=1}^{n} p_i$
J_k / $g_k(P) = \min \{ g_i(P), i = 1, 2, \ldots, n \}$

Application numérique (2)

J_k ?

$i = 1$	$g_i(C_i) = 1 + C_i$	$p_i = 2$
$i = 2$	$= 2C_i$	$= \cancel{2}\ 3$
$i = 3$	$= 10$	$= 5$

$P = \sum p_i = 10$

$g_i(P)$? $g_1(10) = 11$ $g_2(10) = 20$ $g_3(10) = 10$

$\boxed{k = 3}$

Montrons qu'il existe 1 ordo. optimal / J_k soit la dernière tâche exécutée.

Supposons que S soit un ordonnancement de coût minimal $g^* = \max \{ g_i(C_i), i = 1, \ldots, n \}$ et que J_k ne soit pas la dernière tâche exécutée.

Critère régulier: S est 1 bloc (pas de temps mort) et démarre à la date 0.

$S = (A, J_k, B, J_j)$ / A et B blocs de tâches (éventuellement vides)

On considère l'ordonnancement
$S' = (A, B, J_j, J_k)$

Montrons que S' est optimal.

On définit C'_i - date de fin d'exécution dans l'ordonnancement S'

- $\forall i \neq k \quad g_i(C'_i) \leq g_i(C_i)$ | g_i - fonction croissante, $C'_i \leq C_i$

- $g_k(C'_k) = g_k(P)$

Or $g_k(P) \leq g_j(P)$ par déf. de k

Ainsi, $g_k(C'_k) \leq g_j(C_j)$

On en déduit que $\max\{g_i(C'_i), i = 1, \dots, n\}$

$\leq \max\{g_i(C_i), i = 1, \dots, n\}$

$\leq g^*$

g^* est minimal

l'ordonnancement S' est optimal.

3) Séquencer les tâches en partant de la fin

Dernière tâche J_k / $g_k(P) = \min\{g_i(P), i = 1, n\}$

On considère

$J_1 = J \setminus \{J_k\}$

et on pose $P_1 = P - p_k$

On réitère ce principe pour J_1 et P_1 jusqu'au séquencement de l'ensemble des tâches.

Algo. exact

(Validité Q1)

$O(n^2)$

$U = J$

$P = p_1 + p_2 + \dots + p_n$

Pour $j = 1$ à n faire

Identifier J_k de U / ^{+}P

$S_k = P - p_k$ // date de début de J_k

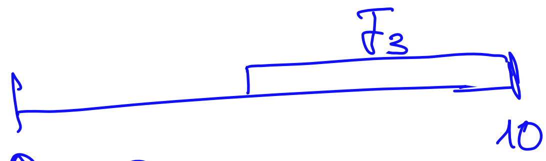
$P = P - p_k$

$U = U \setminus \{J_k\}$

Fin Pour

5) Instance Q2

Itération 1 : $k = 3$



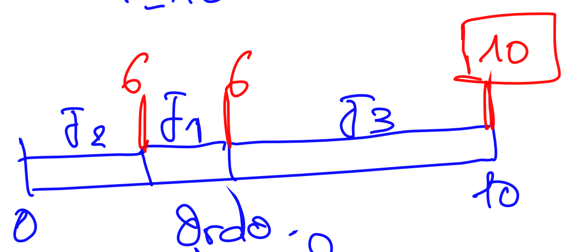
Itération 2 : $P = 10 - 5 = 5$

$k = 1$

$g_1(5) = 6$

$g_2(5) = 10$

$P = 10$



Ordo. optimal

6) $g_i(C_i)$ - retard absolu

$T_i = \max(C_i - d_i, 0)$ - croissante

$\text{Min}(\max T_i)$

$g_k(P) = \min \{ \max(P - d_i, 0), \underbrace{i = 1, \dots, n}_{J_i \in U} \}$

Il suffit de choisir la tâche de + grand d_i.

On retrouve la règle de Jackson (EDD) dans l'ordre des séquences $d_i \nearrow$.

Généralisation $1 \mid prec \mid g_{max}$

Algorithme de Lawler.

$\alpha_i = d^+(J_i)$ - degré extérieur de la tâche J_i (n^{bre} de successeurs)

Sommets du graphe

Ajust. Algo précédent (Mise à jour de α_i)