Paris *le 09/01/2009*

Ordonnancement avec sélection et coûts de setup: les problèmes de base

Jie Meng-Gérard *Philippe Chrétienne*

Introduction

On s'intéresse à des problèmes d'ordonnancements classiques

- avec n jobs J_i,
- m machines parallèles identiques,
- une ligne de temps constituée de T slots successifs $[t-1, t]$,
- etc...

sauf qu'on ajoute 2 possibilités supplémentaires:

1) de la sélection pour les jobs

2) des coûts de setup pour les slots de temps

Introduction: la sélection

La sélection, ça veut dire que:

on n'est pas obligé d'exécuter tous les jobs J_i

mais on doit sélectionner ceux qui vont l'être.

Selon quel critère sélectionner?

on associe à chaque job J_i un gain g_i

et on gagnera g_i si J_i est exécuté et 0 sinon.

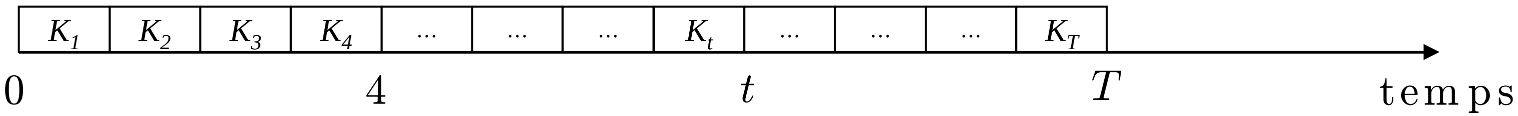
Le problème de la sélection revient à décider pour chaque job J_i s'il va être exécuté ou non (variable booléenne $e_i = 1$ ou $e_i = 0$).

Le gain total d'un ordonnancement est la somme $\sum_{i=1}^{n} e_i g_i$ qu'il faudra essayer de maximiser.

Introduction: les coûts de setup

Les coûts de setup, ce sont:

- des coûts K_t associés à chaque slot de temps t.



- Si au moins une machine est utilisée sur le slot t (variable booléenne $u_t = 1$), il faut payer K_t.
  Si aucune machine n'est utilisée sur le slot t (variable booléenne $u_t = 0$), il ne faut rien payer.

- On paie le même coût de setup K_t, une seule fois, qu'on utilise 1 machine ou m machines sur le slot t.

Le coût total d'un ordonnancement est la somme $\sum_{t=1}^{T} u_t K_t$ qu'il faudra essayer de minimiser.

Introduction: quel critère d'optimisation?

Dans les problèmes coûts de setup/sélection,

ce n'est plus le temps qu'on optimise:

- on ne cherche pas l'ordonnancement le plus rapide,
- ni celui qui minimise le plus les retards,
- ni celui qui est le plus juste-à-temps,
- ...
- on cherche l'ordonnancement le plus rentable!

On cherche l'ordonnancement qui parvient à la fois

à minimiser l'ensemble des coûts de setup $\sum_{t=1}^{T} u_t K_t$ et

à maximiser les gains d'exécution $\sum_{i=1}^{n} e_i g_i$.

Nous verrons que plusieurs fonctions objectif sont possibles.

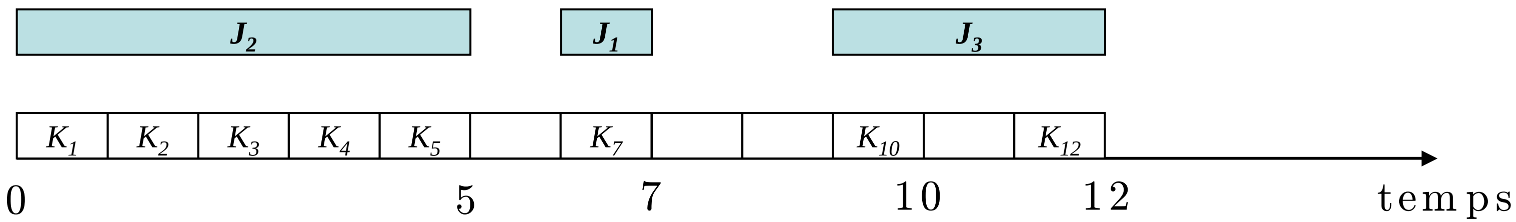
Plan de la présentation

I Coûts de setup seuls: $1|K_t|\sum u_t K_t$

$1|K_t|\sum u_t K_t$ est défini par:

- n jobs $J_1, \ldots, J_n$ de durée $p_1, \ldots, p_n$,
- T slots $t \in \{1, \ldots, T\}$ de coût de setup K_t avec $T \geq \sum_{i=1}^{n} p_i$

Exemple avec $n = 3$, $p_1 = 1$, $p_2 = 5$, $p_3 = 3$, $T = 12$



L'ordonnancement ci-dessus coûte

$$\sum_{t=1}^{12} u_t K_t = (K_1 + \ldots + K_5) + (K_7) + (K_{10} + K_{11} + K_{12}).$$

I Coûts de setup seuls: $1|K_t|\sum u_t K_t$

Propriété 1. Le problème $1|K_t|\sum u_t K_t$ est $\mathcal{NP}$-Complet.

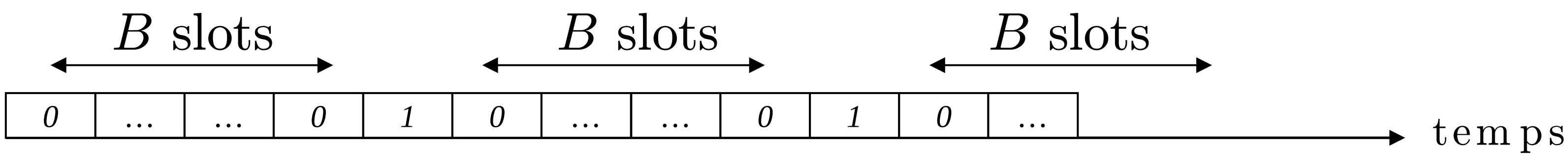
Preuve. On réduit 3-Partition à $1|K_t|\sum u_t K_t$.

L'instance de 3-Partition définie par:

- $3n$ éléments de poids $a_1, \ldots, a_{3n}$
- $\sum_{i=1}^{3n} a_i = nB$ et $B/4 < a_i < B/2$ pour $1 \le i \le 3n$

se réduit à l'instance de $1|K_t|\sum u_t K_t$ définie par:

- $3n$ jobs J_i de durée $p_i = a_i$
- $n(B+1)$ slots dont les K_t sont définis comme suit:



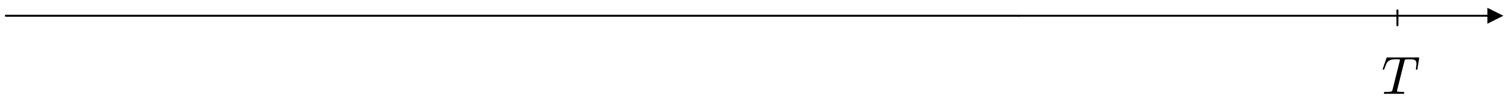
Il y a une 3-Partition *ssi* il y a un ordonnancement de coût nul.

I Coûts de setup seuls: $1|K_t; chain|\sum u_t K_t$

Propriété 2. Le problème $1|K_t; chain|\sum u_t K_t$ est en $O(nT)$.

Preuve. On suppose que les jobs doivent être exécutés dans l'ordre $J_1 \to J_2 \to \ldots \to J_n$.

$$J_1 \to J_2 \to \ldots \to J_n$$



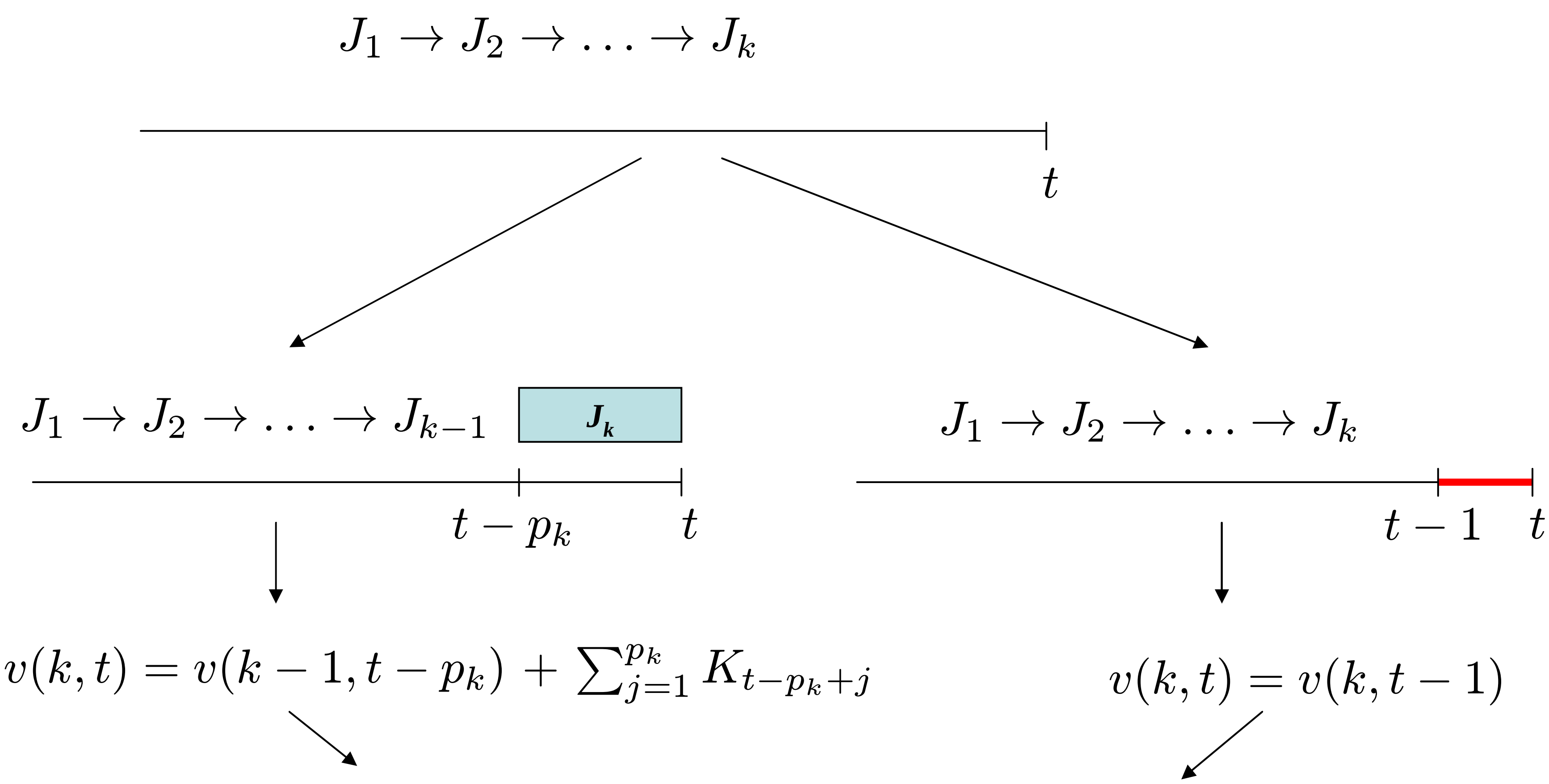
On note $P(k,t)$ le ss-pb ds lequel $J_1, \ldots, J_k$ doivent être exécutés avant t.

On note $v(k,t)$ la valeur optimale de $P(k,t)$.

$$J_1 \to J_2 \to \ldots \to J_k \to J_{k+1} \to \ldots \to J_n$$

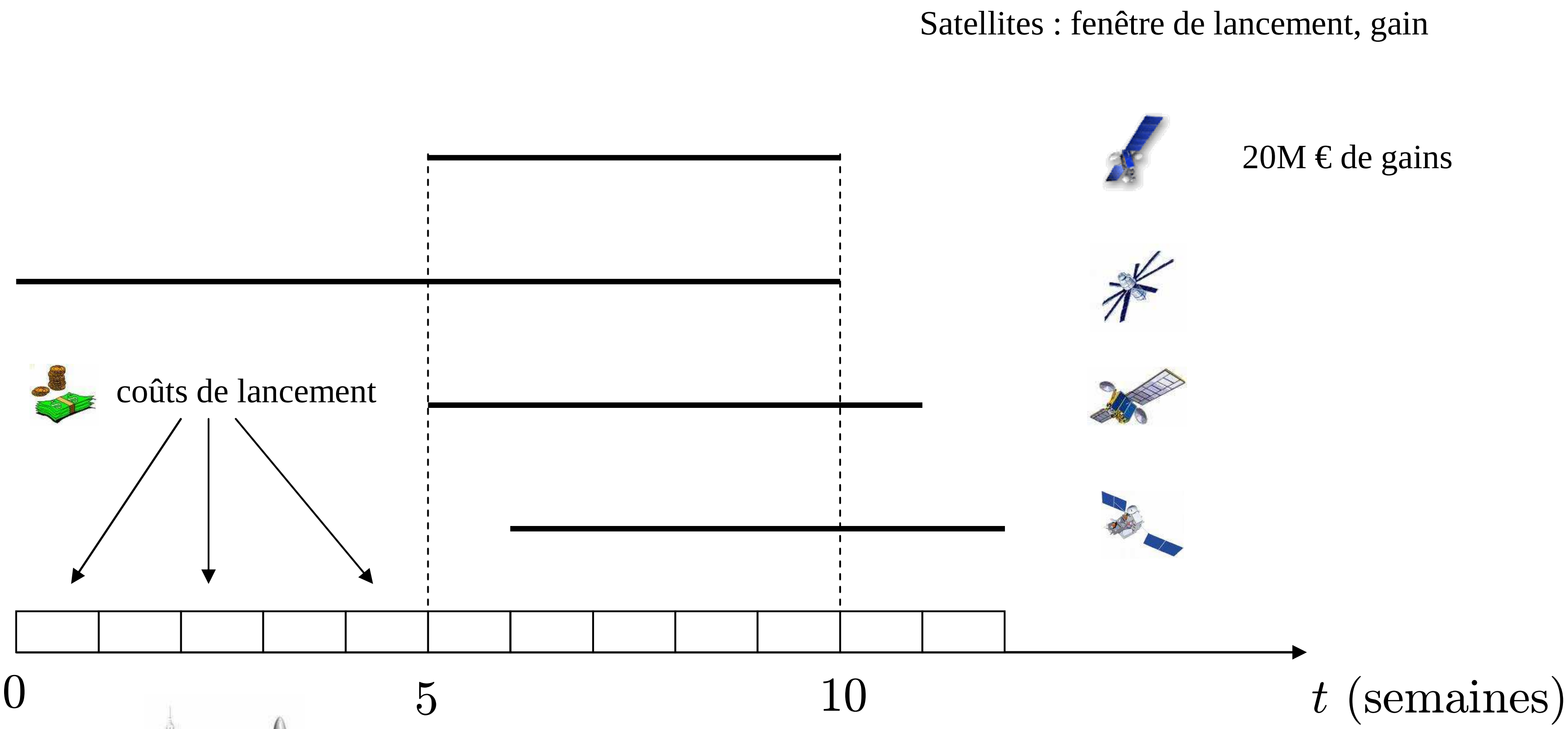
On note $P(k,t)$ le ss-pb ds lequel $J_1,\ldots,J_k$ doivent être exécutés avant t.

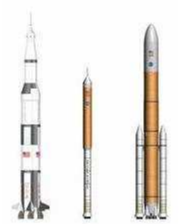
On note $v(k,t)$ la valeur optimale de $P(k,t)$.



$$v(k,t) = \min \begin{cases} v(k-1, t-p_k) + \sum_{j=1}^{p_k} K_{t-p_k+j} \\ v(k, t-1) \end{cases}$$

I Coûts de setup et sélection: le problème de départ





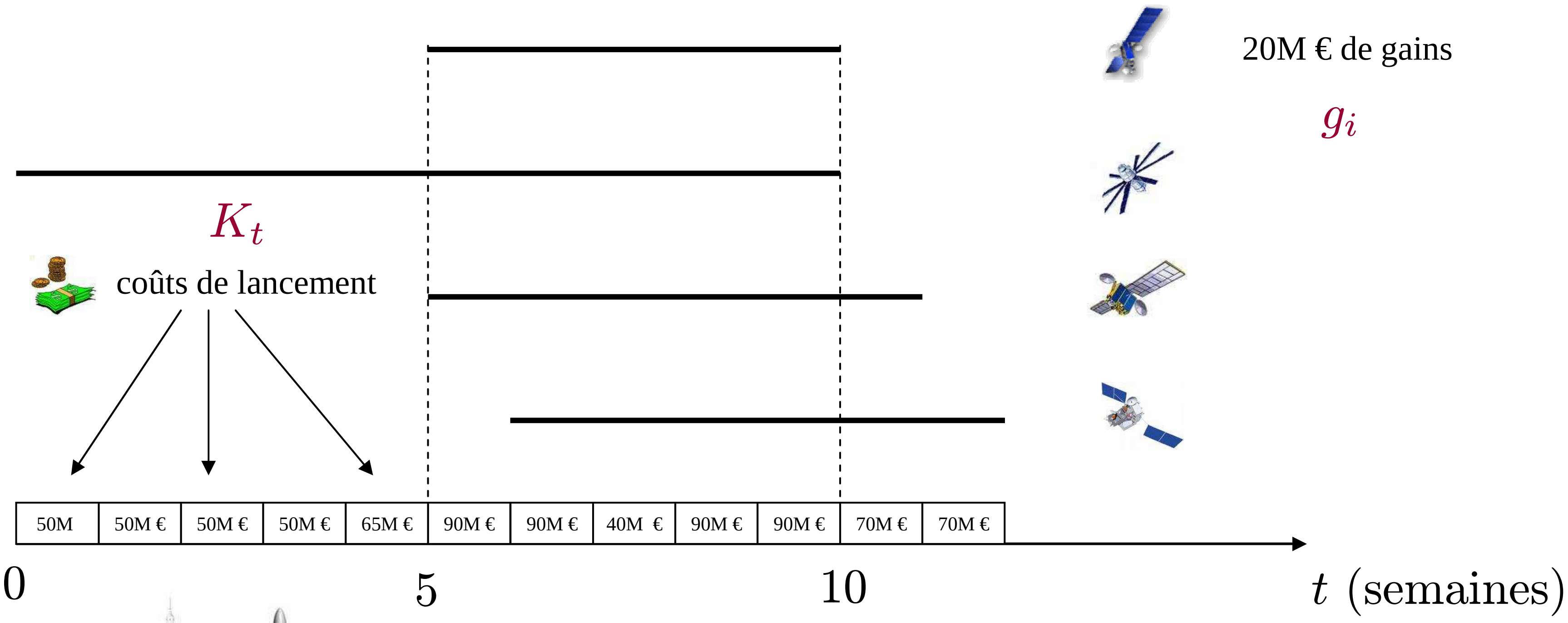
Capacité de chargement limitée: 2 satellites/semaine

$\Rightarrow$ Choisir quels satellites lancer et quand de façon à maximiser les gains

I Coûts de setup et sélection: le problème de départ

Le problème de satellite correspond au problème

$$P|p_i = 1; r_i; d_i; K_t; g_i|\sum u_t K_t - \sum e_i g_i$$



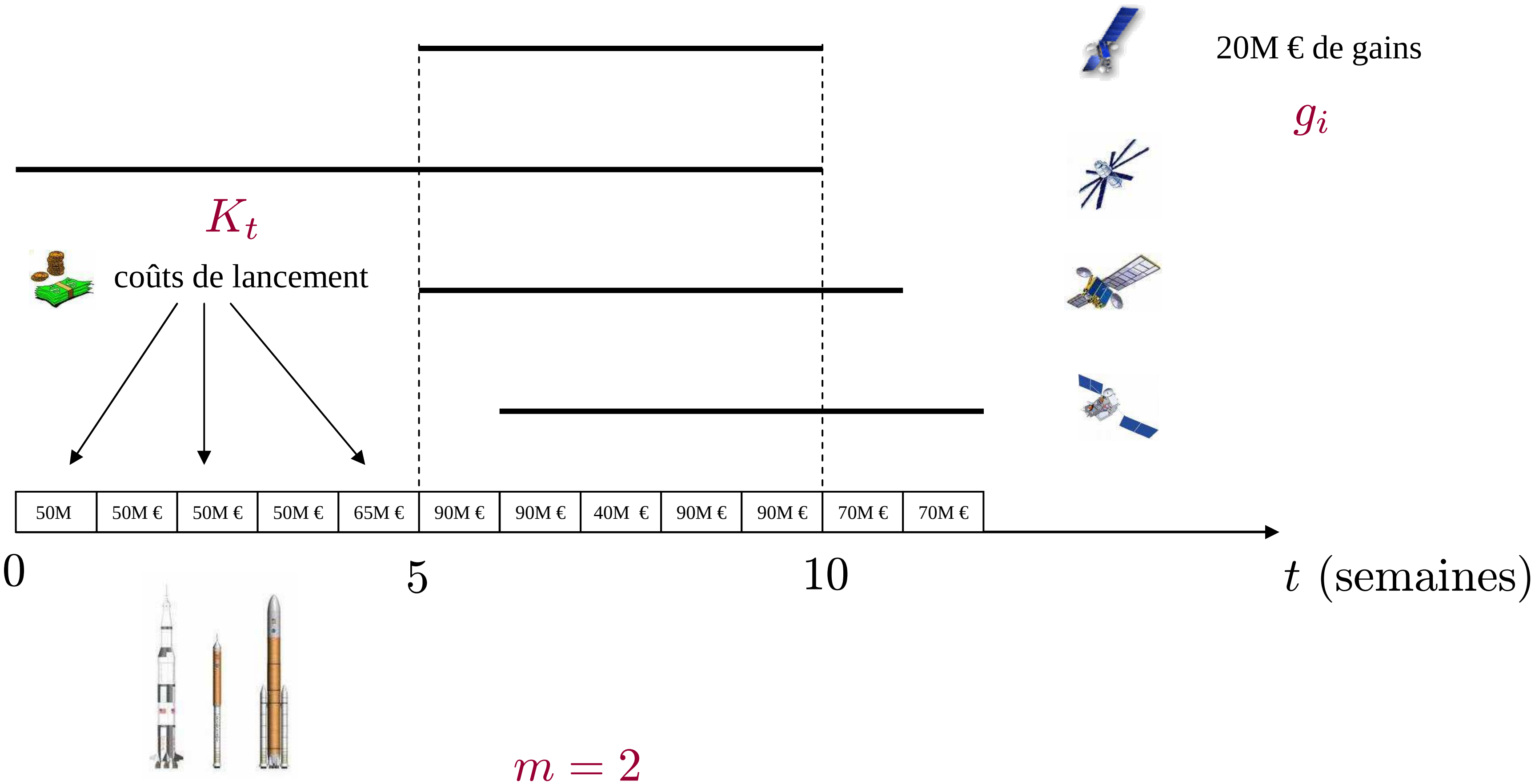
Problème résolu en $O(n^5)$ prog. dyn.

$m = 2$

Capacité de chargement limitée: 2 satellites/semaine

I Coûts de setup et sélection: suite problème satellites (problème ouvert)

$$P2|p_i = 1; r_i; d_i; K_t; g_i; w_i + w_j \leq W|\sum u_t K_t - \sum e_i g_i$$



Capacité de chargement limitée: 2 satellites/semaine

III Coûts de setup et sélection: gains dépendants du temps

Propriété. Le problème $P|p_i = 1; K_t; g_{it}| \sum u_t K_t - \sum e_i g_{it}$ est NP-Complet au sens fort.

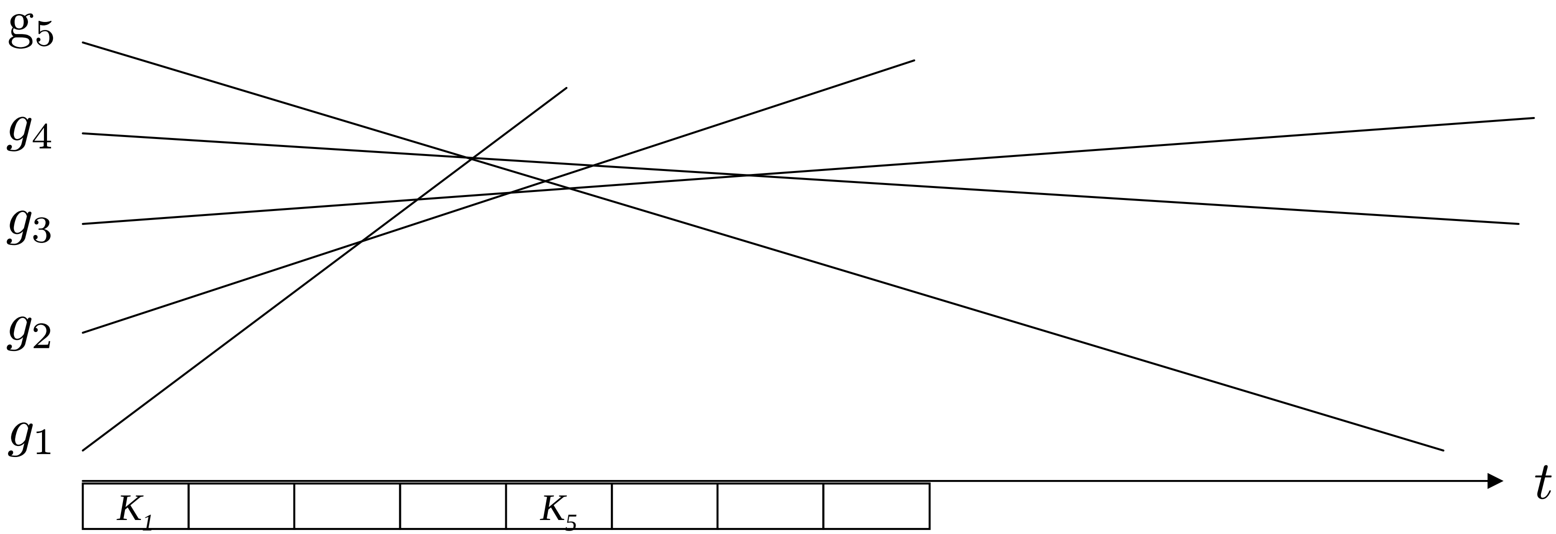
Preuve. On réduit SET COVER à $P_\infty|p_i = 1; g_{it}| \sum u_t \epsilon - \sum e_i g_{it}$.

	S_1	S_2	S_3	S_4	S_5
J_1	0	0	1	0	1
J_2	1	1	0	1	0
J_3	1	0	1	1	0
J_4	1	0	0	0	1

Couverture de taille i *ssi*

ordo. de coût $i\epsilon - n$

Propriété. Le problème $P|p_i = 1; K_t; g_{it} = \alpha_i t + \beta_i| \sum u_t K_t - \sum e_i g_{it}$ est de complexité $O(mnT)$.



Preuve. Algorithme de programmation dynamique basé sur la relation de dominance:

$$\alpha_i \geq \alpha_j \Rightarrow C_i \leq C_j$$

I Coûts de setup et sélection par contrainte

Dans le cas où on a une contrainte budgétaire,
on peut mettre la sélection au niveau d'une contrainte
au lieu de faire une sélection par la fonction objectif.

$$P|r_i, d_i, p_i = 1; g_i; \sum u_t K_t \leq L| - \sum e_i g_i$$

ex: campagne de pub

$$P|r_i, d_i, p_i = 1; g_i; \sum u_t \leq L| - \sum e_i$$

ss-pb le plus simple
(couverture L cliques graphe intervalle)