

# Modélisation et résolution des problèmes d'ordonnancement

## Formalisation

Mohamed Ali ALOULOU

Masters ID et MODO  
Université Paris Dauphine  
E-mail : [aloulou@lamsade.dauphine.fr](mailto:aloulou@lamsade.dauphine.fr)

October 25, 2006

## Exemple

### Critères d'évaluation

### Classification des problèmes d'ordonnancement

- Schémas de classification

- Les environnements machines

- Les tâches et leurs caractéristiques

### Formalisation des problèmes d'ordonnancement

- Modélisation mathématique

- Modélisation à l'aide d'un graphe

### Problème central de l'ordonnancement

# Plan du cours

## Exemple

## Critères d'évaluation

## Classification des problèmes d'ordonnancement

- Schémas de classification

- Les environnements machines

- Les tâches et leurs caractéristiques

## Formalisation des problèmes d'ordonnancement

- Modélisation mathématique

- Modélisation à l'aide d'un graphe

## Problème central de l'ordonnancement

# Enoncé

LEKIN - flexible job-shop scheduling system - [Job Pool - shifting (Job Shop)]

Workspace File Schedule Job Operation Sort Tools Window Help

ID	Wght	Fls	Due	Pt.tm.	Stat.	Bgn
<b>J001</b>	1	0	0	22		4
W001				10	A	4
W002				8	A	15
W003				4	A	28
<b>J002</b>	1	0	0	22		0
W002				8	A	0
W001				3	A	14
W004				5	A	17
W003				6	A	22
<b>J003</b>	1	0	0	14		0
W001				4	A	0
W002				7	A	8
W004				3	A	22

# Description du problème

On 3 jobs et 4 machines

- ▶ Gamme de fabrication de  $J_1 : W_1 \rightarrow W_2 \rightarrow W_3$
- ▶ Gamme de fabrication de  $J_2 : W_2 \rightarrow W_1 \rightarrow W_4 \rightarrow W_3$
- ▶ Gamme de fabrication de  $J_3 : W_1 \rightarrow W_2 \rightarrow W_4$

Deux types de contraintes

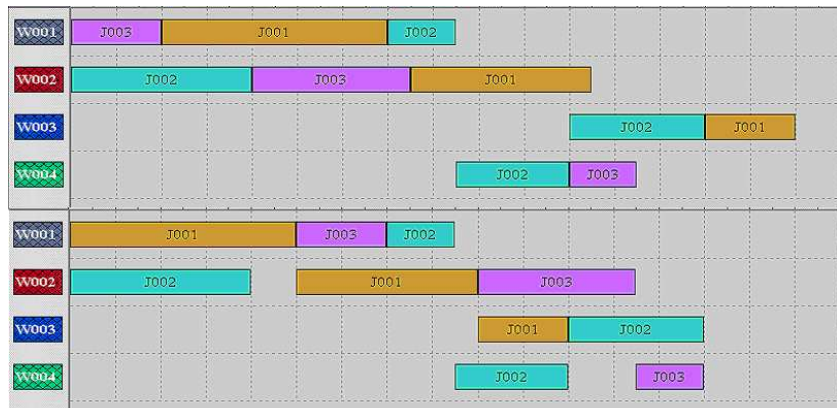
1. Contraintes de succession entre deux opérations d'un même job
2. Contraintes de ressources : une ressource ne peut effectuer qu'une seule tâche à la fois

# C'est quoi un ordonnancement ?

- ▶ Un ordonnancement est complètement caractérisé par les dates de début des opérations de chaque job
- ▶ Un ordonnancement est dit **admissible** s'il satisfait les contraintes du problème

# Représentation d'un ordonnancement

## Diagramme de Gantt orienté machines



## Diagramme de Gantt orienté jobs ...

# Plan du cours

Exemple

Critères d'évaluation

Classification des problèmes d'ordonnancement

Schémas de classification

Les environnements machines

Les tâches et leurs caractéristiques

Formalisation des problèmes d'ordonnancement

Modélisation mathématique

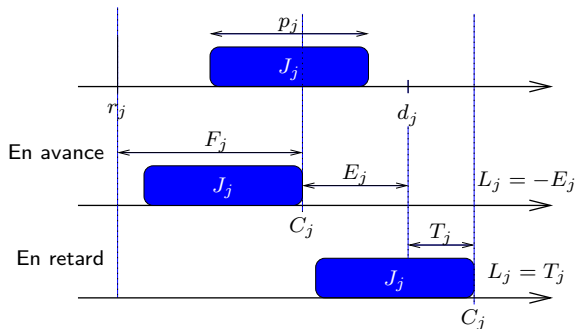
Modélisation à l'aide d'un graphe

Problème central de l'ordonnancement

# Comment évaluer un ordonnancement ?

- ▶ Pour évaluer un ordonnancement on a besoin de définir une (ou plusieurs) fonction(s) objectif(s) ou critère(s)
- ▶ Les critères d'optimisation s'expriment généralement en fonction des dates de fin des tâches (ou jobs)  $C_j$  (**completion times**).
- ▶ Les critères sont généralement exprimés en fonction des mesures suivantes
  - ▶ le retard algébrique  $L_j = C_j - d_j$  (**lateness**)
  - ▶ le retard absolu  $T_j = \max(0, C_j - d_j)$  (**tardiness**)
  - ▶ l'avance  $E_j = \max(0, d_j - C_j)$  (**earliness**)
  - ▶ le pénalité unitaire de retard  $U_j = 0$  si  $C_j \leq d_j$ ,  $U_j = 1$  sinon
  - ▶ le durée de séjour dans l'atelier  $F_j = C_j - r_j$
  - ▶ une fonction générique  $f_j(t)$  donnant le coût induit si on termine  $J_j$  à  $t$

# Illustration



# Les critères d'optimisation

On cherche alors à minimiser un ou plusieurs critères

$$F(C_1, \dots, C_n)$$

- ▶ une fonction générique de **coût maximum**

$$f_{\max} = \max_j \{f_j(C_j)\}$$

- ▶ la durée totale de l'ordonnancement  $C_{\max} = \max_j \{C_j\}$
- ▶ le retard algébrique maximum  $L_{\max} = \max_j L_j$
- ▶ le retard maximum  $T_{\max} = \max_j T_j$

- ▶ une fonction générique de **coût total**  $f_{\Sigma} = \sum_j \{f_j(C_j)\}$

- ▶ la somme (pondérée) des dates de fin  $\sum_j (w_j)C_j$
- ▶ la somme (pondérée) des retards  $\sum_j (w_j)T_j$
- ▶ le nombre (pondéré) des tâches (ou jobs) en retard  $\sum_j (w_j)U_j$
- ▶ la durée moyenne de séjour  $\sum_j F_j$
- ▶ la somme (pondérée) des avances et des retards  $\sum_j \alpha_j E_j + \beta_j T_j$

# Critères réguliers

## Définition

Un **critère**  $F(C_1, \dots, C_n)$  est dit **régulier** si et seulement si  $F$  est une fonction croissante des dates de fin des tâches (ou jobs)

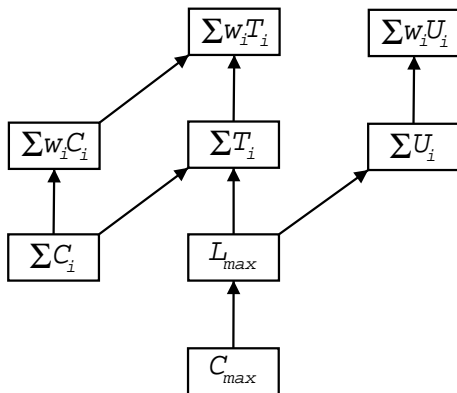
## Corollaire

- ▶ Les critères  $C_{\max}$ ,  $L_{\max}$ ,  $T_{\max}$ ,  $\sum_j (w_j)C_j$ ,  $\sum_j (w_j)T_j$ ,  $\sum_j (w_j)U_j$ ,  $\sum_j (w_j)F_j$  sont réguliers
- ▶ Le critère  $\sum_j \alpha_j E_j + \beta_j T_j$  n'est pas régulier

## Théorème

Les ordonnancements **actifs** sont dominants pour les critères réguliers

## Arbre de réduction en fonction des critères



# Plan du cours

Exemple

Critères d'évaluation

Classification des problèmes d'ordonnancement

- Schémas de classification

- Les environnements machines

- Les tâches et leurs caractéristiques

Formalisation des problèmes d'ordonnancement

- Modélisation mathématique

- Modélisation à l'aide d'un graphe

Problème central de l'ordonnancement

# Schémas de classification

Nous suivons les schémas de classification proposés par (Graham et al, 1979).

Classification à trois champs  $\alpha|\beta|\gamma$

- ▶  $\alpha$  : environnement machine
- ▶  $\beta$  : les caractéristiques des tâches
- ▶  $\gamma$  : le (ou les) critère(s) à optimiser

## D'une façon générale ...

On doit exécuter  $n$  tâches ou  $n$  travaux (*jobs*).

Le champ  $\alpha$  est décomposé en deux sous-champs  $\alpha_1$  et  $\alpha_2$ .

Selon les valeurs prises par  $\alpha_1$ , on distingue :

- ▶ Les problèmes à une machine
- ▶ Les problèmes à machines parallèles
- ▶ Les problèmes d'ateliers
  - ▶ Ateliers à cheminement unique (*flowshop*)
  - ▶ Ateliers à cheminements multiples (*jobshop*)
  - ▶ Ateliers à cheminements libres (*openshop*)
  - ▶ ...
- ▶ L'ordonnancement de projet sous contraintes de ressources

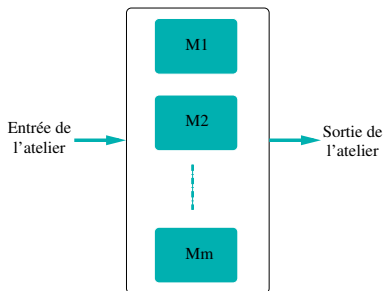
# Problèmes à une machine

Toute tâche  $J_j, j \in \{1, \dots, n\}$  de durée  $p_j$  (*processing time*) s'exécute sur une machine qui ne peut traiter plus qu'une tâche à la fois.

Le champ  $\alpha_1$  est absent et  $\alpha_2 = 1$ .

# Problèmes à machines parallèles (1)

Toute tâche  $J_j, j \in \{1, \dots, n\}$  peut être exécutée indifféremment sur une des  $m$  machines mises en parallèle.



## Problèmes à machines parallèles (2)

$p_{i,j}$  est la durée d'exécution de  $J_j$  sur la machine  $M_i$ ,  $i = 1, \dots, m$ .

- ▶ si  $\alpha_1 = P$  alors **machines identiques**  $\Rightarrow \forall i, p_{i,j} = p_j$
- ▶ si  $\alpha_1 = Q$  alors **machines uniformes**  $\Rightarrow \forall i, p_{i,j} = p_j/s_i$  où  $s_i$  est la vitesse de traitement  $M_i$
- ▶ si  $\alpha_1 = R$  alors **machines indépendantes**  $\Rightarrow \forall i, p_{i,j} = p_j/s_{i,j}$  où  $s_{i,j}$  est la vitesse de traitement de la tâche  $J_j$  par la machine  $M_i$
  
- ▶ si  $\alpha_2$  est un entier positif, le nombre de machines est supposé constant. Si  $\alpha_2$  est absent alors ce nombre est supposé arbitraire.

# Problèmes d'ateliers

- ▶  $m$  machines différentes  $M_i, i \in \{1, \dots, m\}$
- ▶  $n$  travaux (jobs)  $J_j, j \in \{1, \dots, n\}$ .
  - ▶ Chaque job  $J_j$  est décrit par  $n_j$  tâches ou opérations  $O_{i,j}, i \in \{1, \dots, n_j\}$
  - ▶ La durée d'une opération  $O_{i,j}$  est  $p_{i,j}$
  - ▶ La machine qui exécute l'opération  $O_{i,j}$  du job est notée  $M(O_{i,j})$  ou  $M_{i,j}$ .
  - ▶ Les opérations d'un même job ne peuvent pas être exécutées simultanément
- ▶  $\alpha_2$  (voir machines parallèles)

# Ateliers à cheminement unique : Flowshop



- ▶ Chaque Job est constitué de  $m$  opérations et l'ordre de passage sur les différentes machines est le même pour tous les jobs  $J_j : O_{1,j} \rightarrow O_{2,j} \rightarrow, \dots, \rightarrow O_{m,j}$  et  $M_{i,j} = M_i$
- ▶  $\alpha_1 = F$

# Ateliers à cheminements quelconques : Jobshop

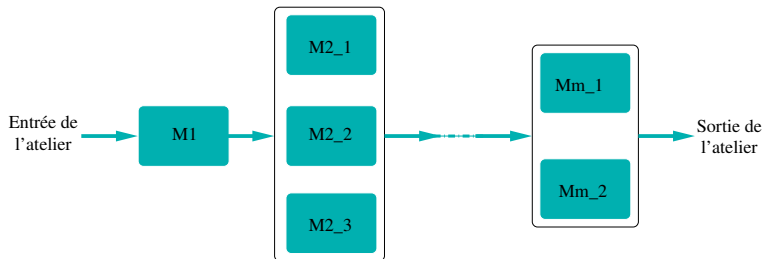
- ▶ Le nombre d'opérations n'est pas forcément le même pour tous les jobs
- ▶ Chaque job a son propre ordre de passage sur les machines
- ▶  $\alpha_1 = J$
- ▶ Exemple

$J_j$	$J_1$			$J_2$			$J_3$	
$O_{i,j}$	$O_{1,1}$	$O_{2,1}$	$O_{3,1}$	$O_{1,2}$	$O_{2,2}$	$O_{3,2}$	$O_{1,3}$	$O_{2,3}$
$M_{i,j}$	$M_1$	$M_2$	$M_3$	$M_2$	$M_1$	$M_3$	$M_3$	$M_2$
$p_{i,j}$	3	2	5	4	2	2	2	3

# Ateliers à cheminements libres : Openshop

- ▶ Le nombre d'opérations n'est pas forcément le même pour tous les jobs
- ▶ L'ordre de passage sur les machines est totalement libre
- ▶  $\alpha_1 = O$

## Autres ateliers : Flowshop hybride



# Les tâches et leurs caractéristiques

Chaque tâche  $J_j$  ou chaque job  $J_j$  peut être caractérisé par

- ▶ une date de début au plus tôt  $r_j$  (**release date**)
- ▶ une durée  $p_j$ , pour la tâche, ou  $p_{i,j}$ , pour l'opération  $i$  du job (**processing time**)
- ▶ une date de fin souhaitée  $d_j$  (**due date**)
- ▶ une date de fin obligatoire  $\tilde{d}_j$  (**deadline**)
- ▶ un poids relatif  $w_j$  (**weight**)  $\Rightarrow$  importance ou poids
- ▶ Il peut y avoir des contraintes de précédence entre les tâches. Ces contraintes sont représentées par un graphe  $G = (\mathcal{J}, A)$ .

# Les tâches et leurs caractéristiques

$\beta = \beta_1\beta_2\beta_3\beta_4\dots$  dans  $\alpha|\beta|\gamma$

- ▶  $\beta_1 = pmtn$  si la **préemption** des tâches est autorisée, sinon  $\beta_1$  est absent
- ▶ S'il y a des **contraintes de précedence** entre les tâches  $\beta_2 \in \{prec, chain, in - tree, out - tree\}$ , sinon  $\beta_2$  est vide
- ▶  $\beta_3 = r_j$  si les **dates de début au plus tôt**  $r_j$  (ou dates de disponibilité) des tâches ne sont pas forcément identiques, sinon  $(\forall j, r_j = 0)$   $\beta_3$  est absent
- ▶  $\beta_4 = \tilde{d}_j$  si la tâche ou le job possède une **date de fin obligatoire**

# Plan du cours

Exemple

Critères d'évaluation

Classification des problèmes d'ordonnancement

Schémas de classification

Les environnements machines

Les tâches et leurs caractéristiques

Formalisation des problèmes d'ordonnancement

Modélisation mathématique

Modélisation à l'aide d'un graphe

Problème central de l'ordonnancement

# Avantages et inconvénients

## Avantages

- ▶ Grande expressivité
- ▶ Formalisme mathématique
- ▶ Possibilité d'utiliser les méthodes de génération de colonnes et de relaxation lagrangienne

## Inconvénients

- ▶ Difficile à résoudre d'une façon générale

# Formulation basée sur les dates de début des tâches

**Variables de décision** : soit les dates de fin des tâches  $C_j$  ou leur dates de début  $S_j$

## Les contraintes

- ▶ Contraintes conjonctives (de potentiels)

forme générale :  $S_j \geq S_i + a_{ij}$

succession simple :  $S_j \geq S_i + p_i$

- ▶ Contraintes disjonctives

- ▶ Forme disjunctive :  $S_j \geq S_i + p_i$  ou  $S_i \geq S_j + p_j$

- ▶ Forme conjunctive

$$Mx_{ij} + S_i \geq S_j + p_j$$

$$M(1 - x_{ij}) + S_j \geq S_i + p_i$$

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

**Objectif** : à vous de compléter

## Autres formulations

1.  $x_{jt} = 1$  si  $j$  commence (ou finit) à l'instant  $t$ , sinon  $x_{jt} = 0$ ,  
 $0 \leq t \leq T$ ,
2.  $\delta_{ij} = 1$  si  $i$  précède  $j$  et 0 sinon,
3.  $x_{(ik)j} = 1$  si  $j$  est placée en  $k$ ème position sur la machine  $i$



# Plan du cours

Exemple

Critères d'évaluation

Classification des problèmes d'ordonnancement

Schémas de classification

Les environnements machines

Les tâches et leurs caractéristiques

Formalisation des problèmes d'ordonnancement

Modélisation mathématique

Modélisation à l'aide d'un graphe

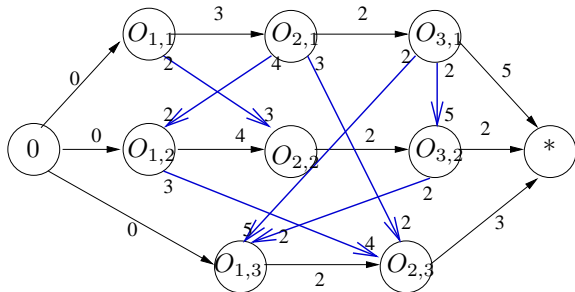
**Problème central de l'ordonnancement**

# Problème central de l'ordonnancement

- ▶ Le problème central de l'ordonnancement est le problème où les ressources sont supposées illimitées
- ▶ illimitées ne veut pas dire infinies
- ▶ illimitées peut vouloir dire que tous les conflits sont arbitrés

## Quel est l'importance du problème central ?

Si on arbitre les conflits, on peut calculer les dates de début au plus tôt des opérations



**Question** : Si le problème était  $J|r_j|L_{\max}$ , quelles modifications doit on apporter au graphe pour que ça marche ?