CAHIER DU LAMSADE

Laboratoire de Management Scientifique et Aide à la Décision
(Université Paris IX Dauphine)

Notes d'après le livre de BAKER
"INTRODUCTION TO SEQUENCING AND SCHEDULING"

Mlle PORTMANN

n°4-1976

Avril 1976

Ce livre* fait une synthèse de ce qui a été publié sur le sujet : ordonnancement d'atelier ou affectation de n travaux à m machines avec réemploi.

Ce résumé est composé de deux parties :

- des schémas qui montrent les différentes hypothèses qui ont été étudiées et sous lesquelles des ordonnancements optimaux ou "sous-optimaux" sont obtenus par des algorithmes ou des heuristiques ;

- de brefs développements qui présentent les algorithmes ou leur principe.

Il est nécessaire en outre de préciser le sens des notations et de donner une propriété générale.

CARACTERISTIQUES DES TRAVAUX

t_j (processing time) durée
r_j (ready time) date au plus tôt
d_j (due date) date au plus tard au-delà de laquelle le travail non achevé est en retard
C_j (completion time) date de fin ou d'achèvement
F_j (flowtime) temps de présence ou d'écoulement $(= C_j - r_j)$
L_j (lateness) avance ou retard $(= C_j - d_j)$
T_j (tardiness) retard $(= \max(0, L_j))$

CRITERES

M (makespan) durée totale

$$\bar{F} = \frac{1}{n} \sum_{j=1}^{n} F_j \qquad F_{max} = \max_{1 \leqslant j \leqslant n} \{F_j\}$$

$$\bar{L} = \frac{1}{n} \sum_{j=1}^{n} L_j \qquad L_{max} = \max_{1 \leqslant j \leqslant n} \{L_j\}$$

$$\bar{T} = \frac{1}{n} \sum_{j=1}^{n} T_j \qquad T_{max} = \max_{1 \leqslant j \leqslant n} \{T_j\}$$

Les moyennes pouvant être pondérées :

$$\text{ex : } \bar{F}_w = \frac{\sum_{j=1}^{n} w_j F_j}{\sum_{j=1}^{n} w_j}$$

$\bar{J}$ nombre de travaux présents en moyenne entre 0 et M
$\bar{V}$ cf. $\bar{J}$ mais les travaux sont pondérés
N_T nombre de travaux en retard

remarque : tous ces critères sont des fonctions des C_j.

PROPRIETE GENERALE

Un critère est dit régulier quand sa valeur décroit si et seulement si l'un au moins des C_j décroit (les autres restant fixes).

Un sous-ensemble est dit dominant pour un critère s'il contient au moins un optimum relatif à ce critère.

Se limiter aux sous-ensembles dominants permet de restreindre l'ensemble des ordonnancements à examiner pour trouver un optimum (ensembles qu'il est plus facile de découvrir lorsque le critère est régulier).

En particulier :

pour le problème de base dans le cas de une machine :

- les ordonnancements sans travaux interrompus sont dominants.
- les ordonnancements ne laissant jamais la machine inoccupée sont dominants.

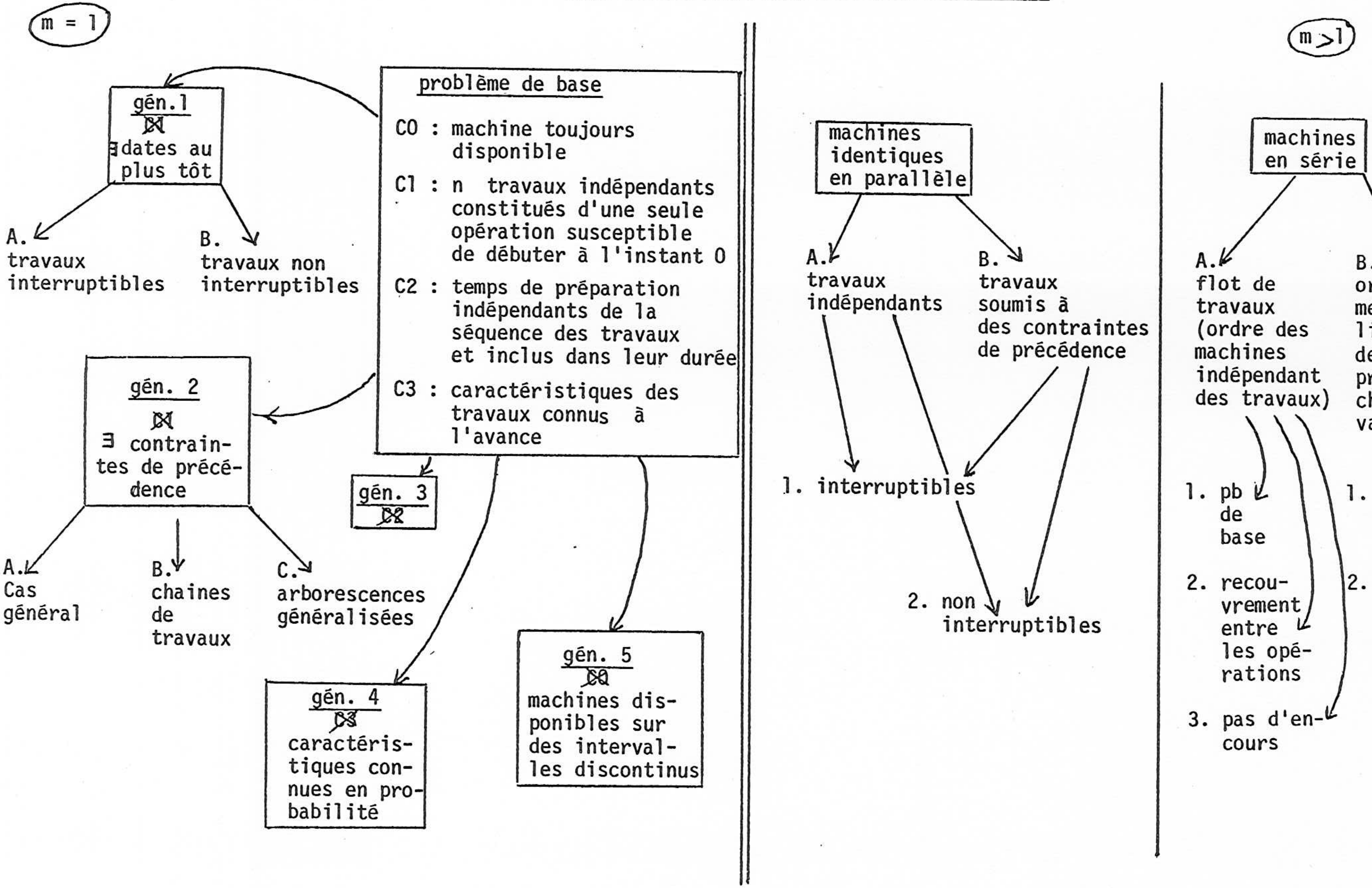
pour les flots de travaux :

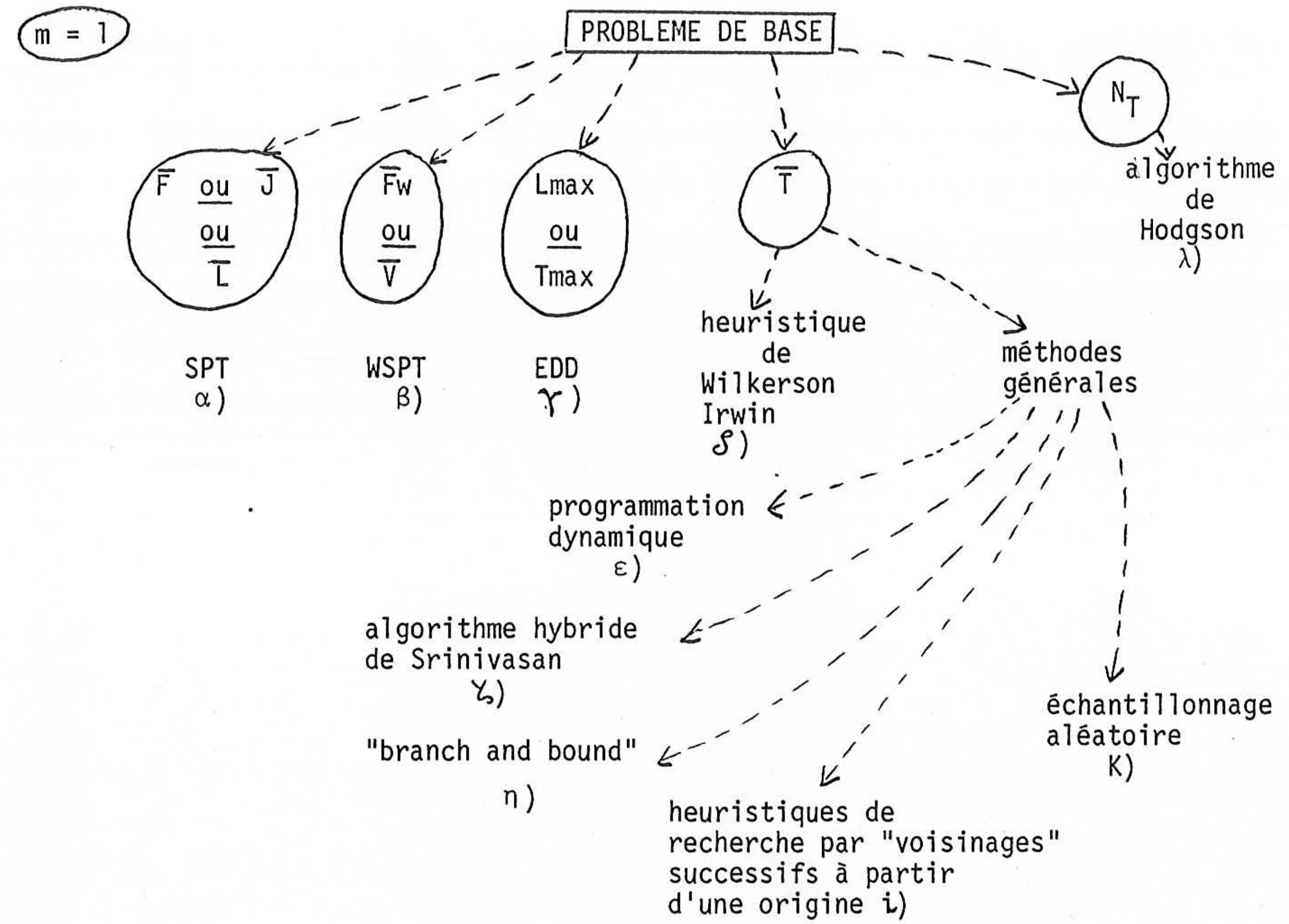
- si le critère est régulier, les ordonnancements qui ont la même séquence de travaux sur les machines 1 et 2 sont dominants.
- si le critère est M , les ordonnancements qui ont la même séquence de travaux sur les machines m et $m - 1$ sont dominants.

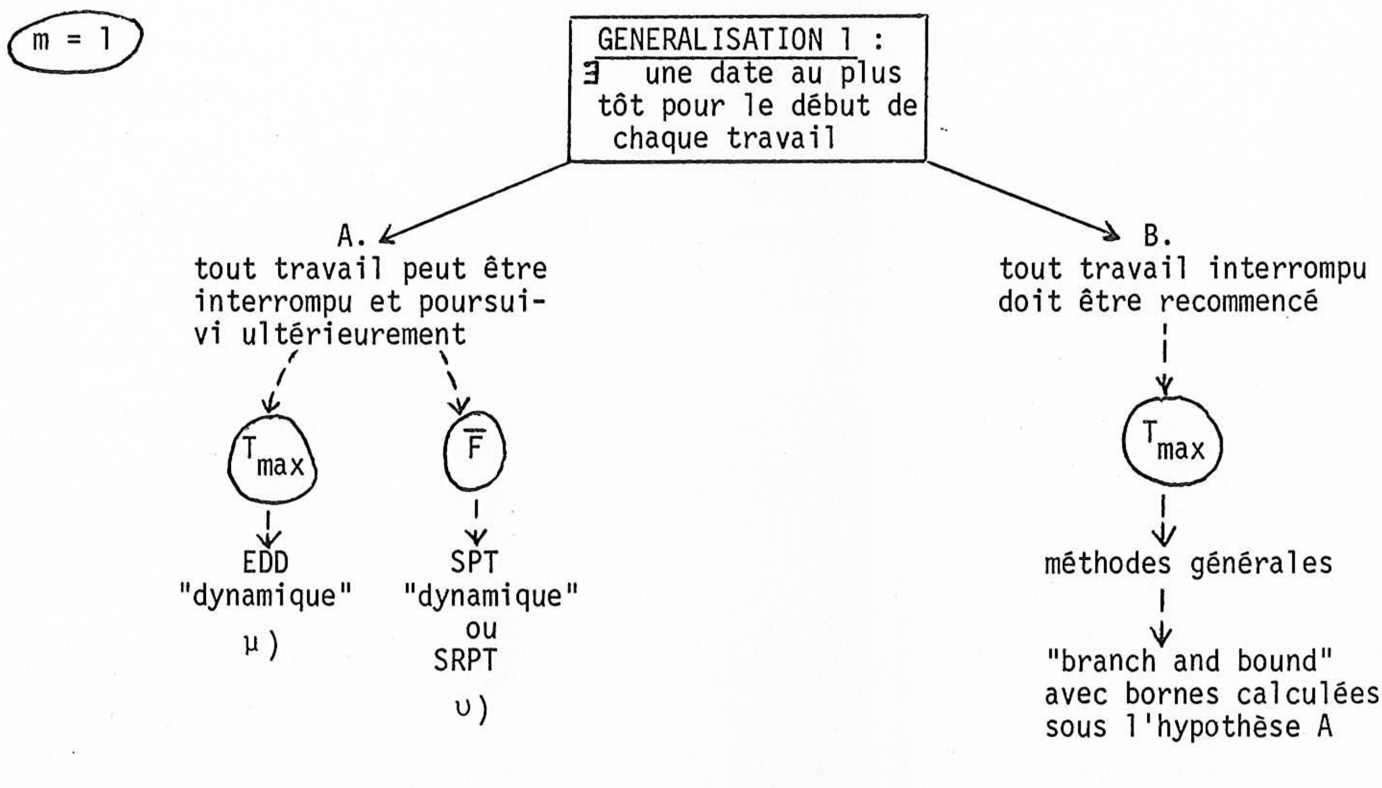
pour le cas le plus général (machines en série et critère M)

- les ordonnancements "semi-actifs" sont dominants.
- il en est de même pour les "actifs" mais pas pour ceux "sans délai".

AFFECTATION DE n TRAVAUX A m MACHINES AVEC REEMPLOI




m = 1
PROBLEME DE BASE
F̄ ou J̄ ou L̄
F̄w ou V̄
Lmax ou Tmax
T̄
N_T
SPT α)
WSPT β)
EDD γ)
heuristique de Wilkerson Irwin δ)
méthodes générales
algorithme de Hodgson λ)
programmation dynamique ε)
algorithme hybride de Srinivasan ζ)
"branch and bound" η)
heuristiques de recherche par "voisinages" successifs à partir d'une origine ι)
échantillonnage aléatoire K)


m = 1
GENERALISATION 1 :
∃ une date au plus tôt pour le début de chaque travail
A.
tout travail peut être interrompu et poursuivi ultérieurement
B.
tout travail interrompu doit être recommencé
T_max
F̄
T_max
EDD "dynamique" μ)
SPT "dynamique" ou SRPT ν)
méthodes générales
"branch and bound" avec bornes calculées sous l'hypothèse A

m = 1

GENERALISATION 2 :
∃ des contraintes de précédence entre les travaux

A.
Cas général

T_{max}
ou
L_{max}

EDD'
ξ)

B.
Chaînes de travaux :
chaînes indépendantes
ordre des travaux imposé dans chaque chaîne

$\overline{F}$

SPT'
o)

C.
Arborescences généralisées

$\overline{F}_w$

algorithme de Sidney
π)

m = 1

GENERALISATION 3 :
temps de préparation dépend de la séquence

M

méthodes générales pour résoudre le problème du voyageur de commerce

programmation dynamique

"branch and bound" algorithme de Little

heuristique de la ville non visitée la plus proche

programmation en nombres entiers

$m = 1$

GENERALISATION 4 :
On connait seulement l'espérance mathématique des durées et des dates au plus tard

$E(\bar{F}_w)$

WSP"E(T)"

ρ)

$\max\limits_j P_r[Lj \geqslant 0]$

E"E(DD)"

σ)

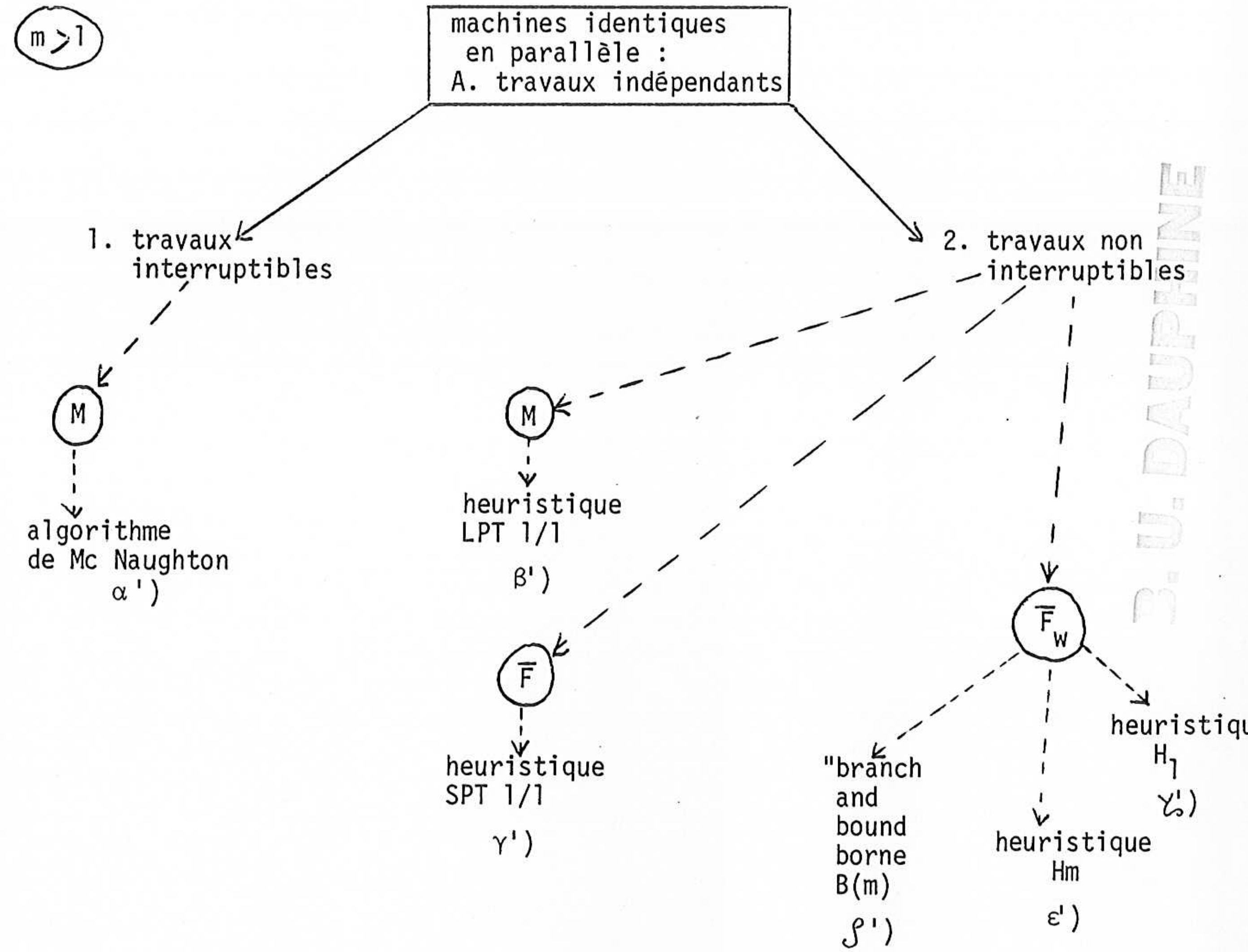
$m = 1$

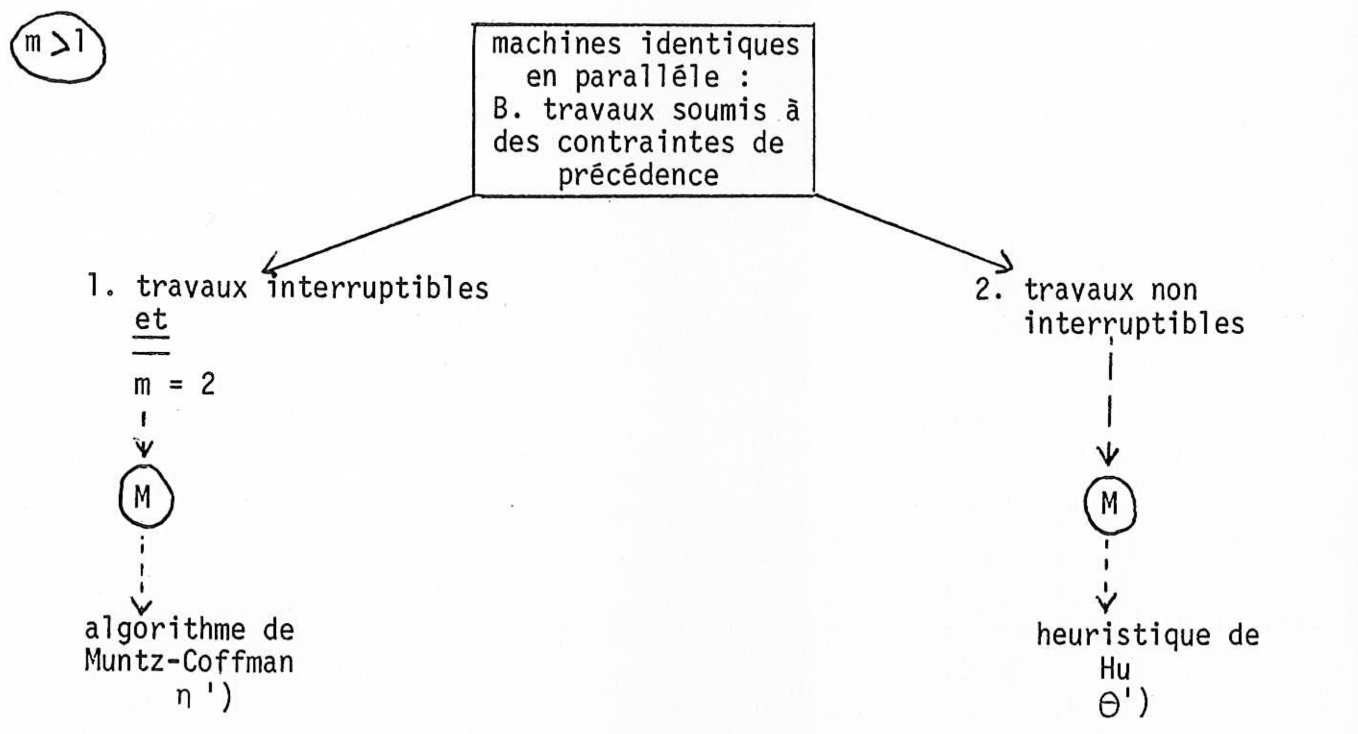
GENERALISATION 5 :
machine disponible sur des intervalles discontinus comprenant des heures de travail normales et des heures supplémentaires au coût horaire b.

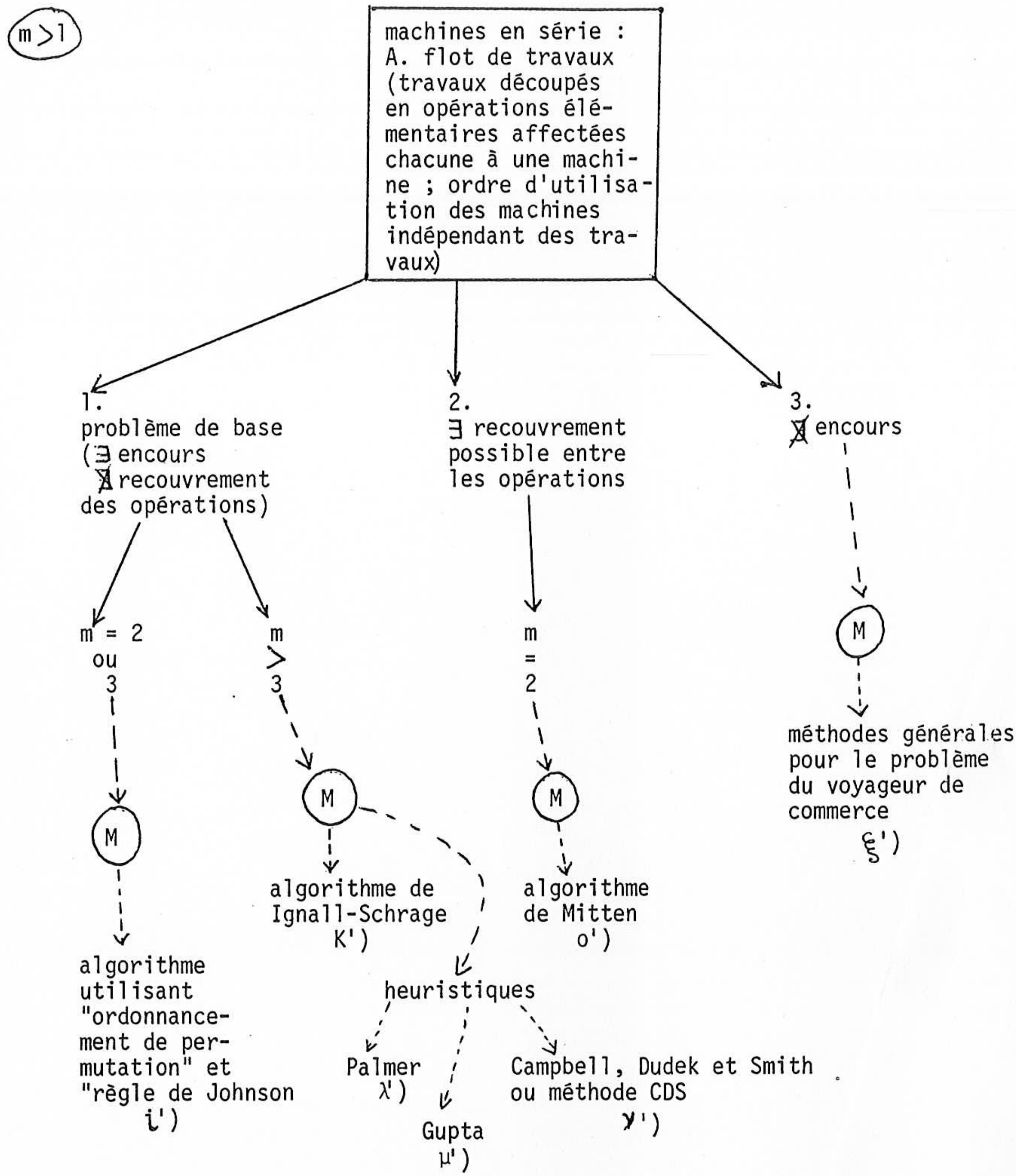
$b.\sum\limits_k x_k + \bar{T}_{w1} + \bar{F}_{w2}$

"branch and bound"
bornes proposées par Gelders et Kleindorfer

τ)


m > 1
machines identiques
en parallèle :
A. travaux indépendants
1. travaux
interruptibles
M
algorithme
de Mc Naughton
α')
2. travaux non
interruptibles
M
heuristique
LPT 1/1
β')
F
heuristique
SPT 1/1
γ')
F_w
"branch
and
bound
borne
B(m)
ρ')
heuristique
Hm
ε')
heuristiqu
H_1
γ₀')


m > 1
machines identiques
en parallèle :
B. travaux soumis à
des contraintes de
précédence
1. travaux interruptibles
et
m = 2
M
algorithme de
Muntz-Coffman
η')
2. travaux non
interruptibles
M
heuristique de
Hu
θ')


m > 1
machines en série : A. flot de travaux (travaux découpés en opérations élémentaires affectées chacune à une machine ; ordre d'utilisation des machines indépendant des travaux)
1. problème de base (∃ encours ∄ recouvrement des opérations)
2. ∃ recouvrement possible entre les opérations
3. ∄ encours
m = 2 ou 3
m > 3
m = 2
M
M
M
M
algorithme utilisant "ordonnancement de permutation" et "règle de Johnson ι')
algorithme de Ignall-Schrage K')
heuristiques
Palmer λ')
Gupta μ')
Campbell, Dudek et Smith ou méthode CDS ν')
algorithme de Mitten o')
méthodes générales pour le problème du voyageur de commerce ξ')

m > 1

machines en série :
B. ordonnancement d'ateliers (travaux découpés en opérations, ordre des machines propre à chaque travail)

1. une machine par opération

∀critère

emploi de générateurs d'ordonnancements (réalisables, semi-actifs, actifs, sans délai)

π')

"branch and bound"

dispatching sur règles de priorité validées par des études de simulation

ρ')

2. machines en parallèle par opération

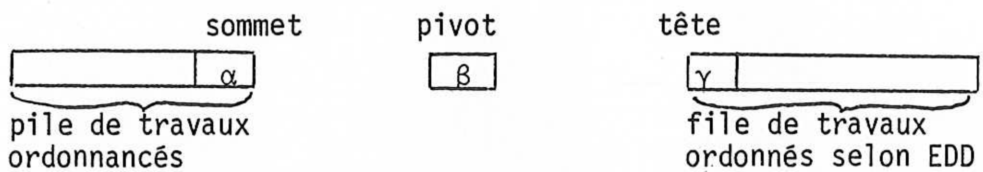
cf.1
avec générateurs plus complexes et bornes du "branch and bound" qui tiennent compte des machines en parallèle

α) SPT (shortest processing time)
les travaux sont ordonnancés dans l'ordre non décroissant des durées

β) WSPT (weighted shortest processing time)
les travaux sont ordonnancés dans l'ordre non décroissant des quantités: durée divisée par poids.

γ) EDD (earliest due date)
les travaux sont ordonnancés dans l'ordre non décroissant des dates au plus tard.

δ) heuristique de Wilkerson-Irwin
principe de la méthode :



initialisation
on ordonne tous les travaux selon EDD :

- pile = premier travail
- pivot = second travail
- file = restant des travaux

tant qu'il existe des travaux non ordonnancés faire
application de règles (cf. Baker pages 29 à 35) qui permettent de choisir une des trois actions a) b) ou c)
a) β empilé ; pivot = γ ; γ oté de la file
b) γ empilé ; γ oté de la file
c) α dépilé ; α remis dans la file dans l'ordre EDD
fintantque

ε) programmation dynamique
critère de structure additive : $Z = \sum_{j=1}^{n} g_j(C_j)$

principe : dans une séquence optimale, toute sous-séquence finale J est optimale et ne dépend que de la somme q_J des durées des travaux qui la précèdent, d'où la formule de récurrence :

$$G(\emptyset) = 0$$

$$\text{et} \quad G(J) = \min \left[g_j(q_J + t_j) + G(J - \{j\}) \right]$$

remarque : en fait, énumération implicite de tous les ordonnancements.

ζ) algorithme hybride de Srinivasan (cf. Baker, pages 49 à 55)
on utilise la programmation dynamique en construisant simultanément à chaque itération des sous-séquences optimales finales et des sous-séquences optimales initiales en utilisant des tests de dominance qui donnent l'ordre dans toute séquence optimale pour certaines paires de travaux.

η) "branch and bound" (pour mémoire) nécessite

a) la définition d'une propriété séparatrice :
ex : si P_σ est l'ensemble des ordonnancements ayant σ comme sous-séquence finale :

$$P_\sigma = \bigcup_{i \notin \sigma} P_{i\sigma}$$

b) la définition de bornes :
ex : pour $\overline{T}$: $v_\sigma = \sum_{j \in \sigma} w_j T_j$
ou mieux $v_\sigma + \min_{j \notin \sigma} \left(w_j \max \left\{0, \sum_{j' \notin \sigma} t_{j'} - d_j\right\}\right)$

c) la définition d'une stratégie :
jumptracking ou backtracking

ι) heuristiques de recherche par "voisinages" successifs

pour différentes origines faire
    point courant = origine
    tant que il existe un voisin x de point courant qui augmente strictement le critère faire point courant = x
    fintantque
    (x est optimum local)
finpour
conserver le meilleur des optima locaux

κ) échantillonnage aléatoire
exemple : on développe un "branch and bound" sans borne et le prochain sous-problème à partitionner est tiré au hasard parmi les fils du précédent.
Toute feuille ainsi obtenue est un élément de l'échantillon.

λ) algorithme de Hodgson

initialisation :

E = file des travaux dans l'ordre EDD

calculer les C_j correspondants

$L = \emptyset$

tant qu'il y a des travaux en retard dans E faire

k = indice du premier travail en retard

j = indice du (premier) plus long travail entre 1 et k

oter j de E et le mettre dans L

recalculer les C_j

fintantque

une solution optimale = E suivi de L (dans un ordre quelconque)

μ) EDD "dynamique"

La machine traite toujours un des travaux accessibles qui a la date au plus tard la plus tôt.

ν) SPT "dynamique" ou SRPT (shortest remaining processing time)

La machine traite toujours un des travaux accessibles dont la durée restante est la plus courte.

ξ) EDD'

On calcule de nouvelles dates au plus tard d'_j de la manière suivante:

soit K l'ensemble des successeurs de j dans le graphe potentiel tâche :

$$d'_j = \min \left(d_j, \min_{k \in K} \{d'_k\}\right)$$

On ordonnance dans l'ordre non décroissant des d'_j.

o) SPT'

Pour chaque chaine on calcule :

$$t'_j = \frac{\sum_{k \in \text{chaine } j} t_k}{n_j}$$

où n_j est le nombre de travaux dans la chaîne. On ordonnance les chaines dans l'ordre non décroissant des t'_j.

π) <u>algorithme de Sidney</u>

a) cas d'une arborescence :

<u>tant qu</u>'il reste des travaux à ordonnancer <u>faire</u>

K = travaux sans "prédecesseurs non placés"

<u>pour</u> tout k de K <u>faire</u>

S_k = ensemble des chemins issus de k

$$s_k = \min_{c_k \in S_k} \frac{\sum_{i \in c_k} t_i}{\sum_{i \in c_k} w_i}$$

<u>finpour</u>

placer un des travaux qui ont le plus petit s_k

<u>fintantque</u>

b) cas d'une anti-arborescence :

processus analogue :

remplacer "prédécesseur" par "successeur" et "chemin issu" par "chemin ayant pour extrémité" ; placer les travaux en partant de la fin.

c) arborescence généralisée :

découper le graphe en arborescences et anti-arborescences et appliquer à chaque tronçon les processus a) ou b).

ρ) <u>WSP"E(T)"</u>

Les travaux sont ordonnancés dans l'ordre non décroissant des quantités : espérance de la durée divisée par le poids du travail.

σ) <u>E"E(DD)"</u>

Les travaux sont ordonnancés dans l'ordre non décroissant des espérances des dates au plus tard.

τ) <u>cf. l'article</u>

"Coordinating aggregate and detailed scheduling decisions in the one machine job-shop: I Theory". Gelders-Kleindorfer, International Institute of Management, D.1000 Berlin 33, Griegstraße 5 I/73.15 (mars 1973).

α') <u>algorithme de Mc Naughton</u>

Calculer $M^* = \max\left\{\frac{1}{m} \sum_{j=1}^{n} t_j, \max_j\{t_j\}\right\}$

Mettre tous les travaux en séquence.

Couper cette séquence en tranches de durée M^*

β') heuristique LPT 1/1

Mettre les travaux dans l'ordre LPT.

Les affecter 1 par 1, dans cet ordre, toujours à la machine la moins occupée.

γ') heuristique SPT 1/1

Cf. β') en remplaçant LPT par SPT.

δ') bornes

B(1) = valeur de $\overline{F_w}$ si $m = 1$

B(n) = valeur de $\overline{F_w}$ si $m = n$

$$B(m) = \frac{1}{2m}\left[(m - 1)\ B(1) + 2B(n)\right]$$

ε') heuristique H_m

Mettre les travaux dans un ordre R.

Les affecter m à m aux machines (les plus longs aux machines les moins occupée

Ordonnancer sur chaque machine dans l'ordre WSPT.

ζ') heuristique H_1

Cf. ε') en remplaçant m à m par 1 à 1.

η') algorithme de Muntz-Coffman (graphe potentiel- tâche sans circuit)

Procédure d'étiquetage :

donner l'étiquette 0 aux travaux sans successeur; pour tout travail j non étiqueté dont les successeurs K sont étiquetés, lui donner l'étiquette :

$$\alpha_j = \max_{k \in K}\{\alpha_k\} + t'_j$$

(où t'_j est la durée restante pour les travaux interrompus)

Phase d'affectation :

tantque tous les travaux ne sont pas entièrement placés faire

- étiquetage
- affecter tous les travaux de plus grandes étiquettes (si leur nombre k est $>$ m : leur accorder une portion de machine m/k et allonger leur durée)
- déterminer l'instant t où il faut remettre en cause cette affectation.

fintantque

Phase finale :

dans chaque intervalle où l'affectation est constante, réorganiser pour ne plus partager les machines entre les travaux.

θ') <u>heuristique de Hu</u> (généralisé au graphe sans circuit)

Phase d'étiquetage (cf. η')

Phase d'ordonnancement :

affecter les travaux sans prédécesseur au plus m par m dans l'ordre décroissant des étiquettes (tout travail affecté étant ôté du graphe).

ι') "<u>ordonnancements de permutation</u>"

Les travaux passent dans le même ordre sur chaque machine (dominants pour m = 2 ou 3 et critère M).

"<u>règle de Johnson</u>" (m = 2; m = 3 cf. Baker page 146)

i précède j dans la séquence si :

$$\min\{t_{i1}, t_{j2}\} \leqslant \min\{t_{i2}, t_{j1}\}$$

K') <u>algorithme de Ignall-Schrage</u> (cf. Baker, pages 149 à 157)

"Branch and bound" où borne = $\max(b_1, b_2)$, b_1 étant orienté machine et b_2 travaux et où l'exploration est tronquée par une propriété dominante :

soit $q_k^{(i)}$ la fin de la séquence $\sigma^{(i)}$ sur la machine k.
Si deux séquences partielles $\sigma^{(1)}$ et $\sigma^{(2)}$ contiennent les mêmes travaux et si $q_2^{(1)} \leqslant q_2^{(2)}$ et $q_3^{(1)} \leqslant q_3^{(2)}$ alors $\sigma^{(1)}$ domine $\sigma^{(2)}$)

En outre on place les travaux consécutivement sur la machine dominante (comportant la somme des durées la plus grande) et l'on remonte et descend à partir d'elle.

λ') <u>heuristique de Palmer</u> :

Calculer $s_j = (m - 1)tj_m + (m-3)\, tj_{m-1} + \ldots - (m - 3)\, tj_2 - (m - 1)\, tj_1$

On ordonnance dans l'ordre non croissant des s_j.

μ') <u>heuristique de Gupta</u>

Cf λ' avec $s_j = \dfrac{e_j}{\min\limits_{1 \leqslant k \leqslant m-1} \{tj_k + tj_{k+1}\}}$

où $e_j = 1$ si $t_{j1} < t_{jm}$ et $- 1$ sinon

ν') <u>méthode CDS</u>

Itération de la règle de Johnson en regroupant les premières et les dernières opérations par 1, puis 2, puis 3, ...

O') <u>algorithme de Mitten</u> (règle de Johnson étendue)

Soit :

a_j : durée nécessaire d'avancement de j_1 avant le début de j_2.

b_j : durée nécessaire avant la fin de j_2 après la fin de j_1.

$U = \{j \mid tj_1 < tj_2\}$ et $V = \{j / tj_1 \geqslant tj_2\}$

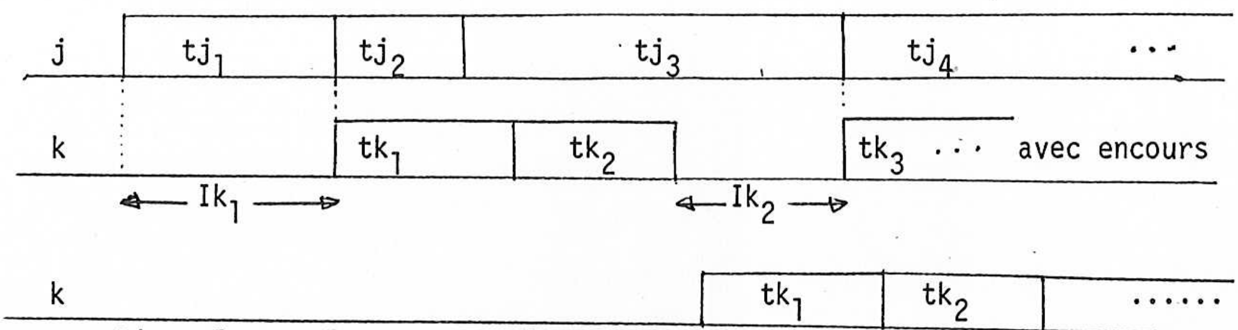
$y_j = \max(aj - tj_1, bj - tj_2)$

U ordonné par $tj_1 + yj$ non décroissant

V ordonné par $tj_2 + yj$ non décroissant

Optimum : U suivi de V.

ξ') <u>pas d'encours : principe</u>



$Dj_k = I_{k1} + I_{k2} + \dots + I_{km}$

donc D_{ij} = décalage entre le début de i et j si i et j se suivent dans un ordonnancement de permutation. Les D_{ij} constituent les distances du problème du voyageur de commerce.

π ') <u>Ordonnancement d'atelier : générateurs</u>.

Notation : ϕ_j : date au plus tôt pour l'achèvement d'une opération j de St.

σ_j date au plus tôt de début de j

a) ordonnancements actifs

1. $t = 0$; $PS_t = \emptyset$; $S_t = \{$opérations j sans prédécesseurs$\}$
2. $\phi^* = \min_{j \in S_t} \{\phi_j\}$

   m^* machine correspondante
3. pour tout $j \in S_t$ et tel que $\sigma_j < \phi^*$ (machine m^* requise) créer un $PS_{t+1} = PS_t \cup \{j\}$ et un $S_{t+1} = S_t - \{j\}$ $\cup \{$successeurs de $j\}$

   retourner à 2. tant que tous les $S_{t'}$ ne sont pas vides.

b) ordonnancements sans délai

1. $t = 0$; $PS_t = \emptyset$; $S_t = \{$opérations j sans prédécesseurs$\}$
2. $\sigma^* = \min_{j \in S_t} \{\sigma_j\}$

   m^* machine correspondante
3. pour tout $j \in S_t$ et tel que $\sigma_j = \sigma^*$ (machine m^* requise) créer un $PS_{t+1} = PS_t \cup \{j\}$ et un $S_{t+1} = S_t - \{j\} \cup \{$successeurs de $j\}$

   retourner à 2. tant que tous les $S_{t'}$ ne sont pas vides.

remarque importante : les ordonnancements sans délai ne constituent pas un ensemble dominant.

ρ') Ordonnancement d'atelier : règles de priorité

SPT : opération de plus courte durée

FCFS : première arrivée, première servie

MWKR : opération qui appartient au travail de durée restante la plus longue

MOPNR : opération qui a le plus grand nombre d'opération derrière

LWKR : contraire de MWKR

RANDQM : au hasard

AWINQ : opération dont le successeur viendra dans la plus courte file d'attente

FOFO : opération qui peut se terminer la première (éventuellement insertion de temps libre)

FASFS : cf. FCFS pour les travaux
TWORK : cf. SPT pour les travaux
EDD : cf. SPT pour les travaux
} même priorité pour les opérations d'un même travail

MST : minimum de détente $(d_j - C_j)$ au niveau du travail

OPNDD : des dates au plus tard sont associées aux opérations : cf. EDD

S/OPN : des dates au plus tard sont associées aux opérations : cf. MST

TSPT : SPT avec exception pour les opérations ayant attendu plus que W unités de temps alors pour elles : FCFS.