Ordonnancement en temps réel d'atelier avec prise en compte des temps de préparation des ressources

Christian Artigues

2 novembre 1998

à Hélène.

Avant-Propos

Le travail présenté dans ce mémoire a été effectué au Laboratoire d'Analyse et d'Architecture des Systèmes du C.N.R.S., dans le cadre d'une convention C.I.F.R.E.(Conventions Industrielles de Formation par la Recherche) avec la société CABINET VILLAUMIE SA dont je fais partie. Je remercie Messieurs Alain Costes et Jean-Claude Laprie de m'avoir accueilli au sein du L.A.A.S. dont ils ont assuré successivement la direction.

Je remercie également Monsieur Jean-Claude Hennet, directeur de recherche au C.N.R.S. et responsable du groupe "Décision et Conduite dans les Systèmes de Production" du LAAS, de m'avoir accueilli dans son équipe.

Je tiens à exprimer toute ma gratitude et mon amitié à François Roubellat, Directeur de recherche au CNRS, qui m'a encadré pour cette thèse. Sa rigueur, son réalisme, son expérience et son exigence ont été les piliers de ce travail qui constitue une étape dans l'élaboration de la méthode O.R.A.B.A.I.D. qu'il a créée. Sur le plan personnel, j'ai particulièrement apprécié ses encouragements constants et les fréquents "Cool, jeune homme !" qu'il a su me lancer pendant la phase finale et difficile de la rédaction du mémoire.

Je remercie par ailleurs Jacques Erschler, Professeur à l'INSA et Directeur du Département de Génie Electrique et Informatique, dont les travaux en analyse sous contraintes sont aussi à l'origine de cette approche, de m'avoir donné l'opportunité d'enseigner dans son département.

Je remercie Messieurs Michel Villaumié et Benoît Vervandier, respectivement Président Directeur Général et Directeur Général de CABINET VILLAUMIE SA de m'avoir accueilli au sein de cette société, et de m'avoir fait confiance pour le développement du coeur de la nouvelle version du logiciel ORDO. Je remercie en outre Benoît Vervandier pour sa participation au jury de thèse.

Je remercie Messieurs Pierre Baptiste, Professeur à L'ENSMM, Christian Proust Professeur et directeur de l'E3I, Marino Widmer, Professeur à l'Université de Fribourg, pour l'intérêt et l'attention qu'ils ont accordé à cette étude et d'avoir accepté d'en être les rapporteurs.

Je remercie Marie-Claude Portmann, Professeur à l'Ecole des Mines de Nancy, pour sa lecture attentive du mémoire, et d'avoir accepté de participer à ce jury.

Je remercie, Marc Courvoisier, Professeur à l'U.P.S. et directeur de l'INSA de Toulouse, d'avoir accepté de participer à ce jury et de l'intérêt qu'il a manifesté suite à la présentation de ces travaux.

Je remercie particulièrement Jean-Claude De Cargouet, Directeur Informatique et Comptable à la société PUBLINIC, d'avoir accepté de participer à ce jury, en y apportant le point de vue d'un utilisateur d'ORDO.

Un grand merci à Jean-Charles Billaut, Maître de Conférences à l'E3I, à qui j'ai succédé sur la méthode ORABAID et qui s'est énormément investi pour me rendre la tâche plus facile, aussi bien pour son aide et ses conseils en début de thèse que pour avoir observé le mémoire à la loupe, décortiqué les algorithmes et relevé les nombreuses coquilles. Merci Jean-Charles, mais comme tu dis si bien : "Dans la vie, on n'est jamais trop aidé" !

Je tiens à remercier chaleureusement les collègues du Groupe ex-S.P., actuellement D.C.S.P. et futur O.C.S.D. qui m'ont aidé dans différentes phases de ce travail: Jean Lasserre, qui m'a encouragé et aidé à structurer de façon cohérente le troisième chapitre, Robert Valette, pour ses conseils sur les réseaux de Petri Colorés, Ronan Champagnat, pour la même raison et pour les correction qu'il a apportées à "mes" réseaux, Eric Zamai pour son aide lors de la préparation de la soutenance, François Giraud, Nabil Ben Khalifa et Adel Benzina, pour l'entraide entre thésards en fin de rédaction, William Roux pour, entre autres, la méthode "tabou", Patrick Esquirol pour les discussions en ordonnancement et en algorithmique, mais aussi pour sa bonne humeur, Pierre Lopez, pour sa lecture d'une partie du mémoire et pour les discussions sur les thèmes de l'ordonnancement et de la techno, qui m'ont beaucoup apporté dans les deux domaines, et Philippe Torres pour l'algorithme que je lui ai piqué.

Hors du L.A.A.S. Je tiens à remercier Stéphane Dauzère-Pérès de l'Ecole des Mines de Nantes pour ses conseils, Clarisse Flipo de Grenoble qui a gentiment relu et corrigé l'état de l'art sur les temps de préparation.

Je remercie toute l'équipe du Cabinet Villaumié et plus particulièrement Odile, Didier, Isabelle, Lydie,... qui m'ont toujours bien accueilli lors de mes séjours à Besançon. A ce sujet, je remercie même Anne, malgré sa facheuse habitude de remplir systèmatiquement mon sac de confettis.

Mes remerciements vont également à l'ensemble du personnel des services informatique, administratif et de reproduction du LAAS, en particulier à Michelle Powell, Christian Berty et Eliane Dufour.

Pour finir : spéciale dédicace à mes parents, mes grand-parents et ma soeur Corinne et à tous mes copains Philippe "Toto", Gilles, William, Marco, Hélène D., Jérome "La tchatche", Isabelle "LA phrase p. 35", J.P. "Jurançon et micro-informatique", Cédric "Sea, Sex and Sun", Stéphanie "Lucette", Didier "Lo Bernat", les "petits" Blanquet, Yannick "Guédos", Marc "La Darre", Vincent "La Vince", Yannick "La Lesple", Pierre "Saturnin", Jean-Marc "Tachos", Richard "Riri", Francis "Braquos", Laurent "Chouchi", Anne "Le Poulpe", Olivier "Castipoil", Eric "Zille", Sandrine, Valérie, Marie-Pierre, Steph le Lorrain, Laurent le Troyen, P.Y.M. le Breton, Pierre "Yvette" l'Alsacien, Alex G., Christophe "Crichou" le Ch'ti., et les autres...

Table des matières

Glossaire

Chapitre II

$\mathcal{R}$: ensemble des ressources de l'atelier

M : nombre de ressources de l'atelier

E_k : nombre de lignes de charge d'une ressource k

N : nombre d'ordres de fabrication

i : indice relatif aux ordres de fabrication

r_i : date de début au plus tôt de i

d_i : date de livraison de l'ordre de i

h_i : nombre d'opérations de i

g_i : gamme de i

$G_i = X_i, U_i$: graphe potentiels-tâches des relations de précédence entre les opérations de i

(i, j) : opération j de l'ordre de fabrication i

m_{ij} : nombre de modes étendus d'exécution de (i, j)

m : indice relatif aux modes étendus d'exécution

M_{ij}^m : nombre de pools de (i, j)

$\mathcal{P}_{ij}^m$: ensemble des pools de (i, j)

P_{ij}^{mr} : $r^{\text{ième}}$ pool de (i, j) dans le mode étendu m

e_{ij}^k : nombre de ressources utilisées par (i, j) sur la ressource k

e_{ij-}^{mk} : nombre minimal de lignes de charge utilisables par (i, j) sur la ressource k dans le mode étendu m

e_{ij+}^{mk} : nombre maximal de lignes de charge utilisables par (i, j) sur la ressource k dans le mode étendu m

l_{ij} : occupation de (i, j) sur l'ensemble des ressources $\mathcal{R}$

$\mathcal{R}_{ij}$: sous-ensemble des ressources de $\mathcal{R}$ utilisées par (i, j) et définies par l_{ij}

e_{ij}^{kl} : occupation de (i, j) sur la ligne de charge l de la ressource k.

Pg_{ij}^m : pool guidant de l'opération

p_{ij} : durée de l'opération

p_{ij}^{mge} : durée de l'opération si (i,j) est exécutée sur e lignes de charge de la ressource guidante g.

$prec_{ij}^{i}$: ensemble des opérations précédentes de (i,j) dans g_i

$succ_{ij}^{i}$: ensemble des opérations suivantes de (i,j) dans g_i

$\mathcal{O}$: ensemble des opérations d'exécution

$\mathcal{R}b$: ensemble des ressources principales de l'atelier

M_b : nombre de ressources principales dans l'atelier

$\mathcal{R}_c$: ensemble des ressources complémentaires de l'atelier

M_c : nombre de ressources complémentaires dans l'atelier

$\mathcal{P}b_{ij}^{m}$: ensemble des pools principaux de (i,j) dans le mode m

Mb_{ij}^{m} : nombre de pools principaux de (i,j)

$\mathcal{P}c_{ij}^{m}$: ensemble des pools complémentaires de (i,j) dans le mode m

Mc_{ij}^{m} : nombre de pools complémentaires de (i,j)

lb_{ij} : occupation des ressources principales par (i,j)

lc_{ij} : occupation des ressources complémentaires par (i,j)

$\mathcal{R}b_{ij}$: ensemble des ressources principales utilisées par (i,j) ou association

$\mathcal{R}c_{ij}$: ensemble des ressources complémentaires utilisées par (i,j)

$\mathcal{T}$: ensemble des types d'opérations d'exécution

RT_{ij} : type de l'opération (i,j)

sm $(sd,\ sct)$: opération de montage (démontage, changement de type)

μ_{sm} $(\delta_{sd},\ \tau_{sct})$: nombre de modes étendus de montage (démontage, changement de type)

μ $(\delta,\ \tau)$: indices relatifs aux modes étendus de montage (démontage, changement de type)

$\mathcal{P}b_{sm}^{\mu}$ $(\mathcal{P}b_{sd}^{\delta},\ \mathcal{P}b_{sct}^{\tau})$: ensemble des pools principaux de l'opération de montage (démontage, changement de type) dans le mode étendu μ $(\delta,\ \tau)$

Mb_{sm}^{μ} $(Mb_{sd}^{\delta},\ Mb_{sct}^{\tau})$: nombre de pools principaux de montage (démontage, changement de type) dans le mode étendu μ $(\delta,\ \tau)$

$Pb_{sm}^{\mu r}$ $(Pb_{sd}^{\delta r},\ Pb_{sct}^{\tau r})$: $r^{\text{ième}}$ pool principal dans le mode étendu de montage μ (démontage δ, changement de type τ)

$\mathcal{A}^{\mu}$ $(\mathcal{A}^{\delta},\ \mathcal{A}^{\tau})$: ensemble des associations définies par le mode étendu de montage μ (démontage δ, changement de type τ)

$\mathcal{P}c_{sm}^{\mu}$ $(\mathcal{P}c_{sd}^{\delta},\ \mathcal{P}c_{sct}^{\tau})$: ensemble des pools complémentaires de montage, (démontage, changement de type) dans le mode étendu μ $(\delta,\ \tau)$

Mc_{sm}^{μ} $(Mc_{sd}^{\delta},\ Mc_{sct}^{\tau})$: nombre de pools complémentaires de montage, (démontage, changement de type) dans le mode étendu μ $(\delta,\ \tau)$

$Pc_{sm}^{\mu r}$ $(Pc_{sd}^{\delta r},\ Pc_{sct}^{\tau r})$: $r^{\text{ième}}$ pool complémentaire dans le mode étendu de montage μ, (démontage δ, changement de type τ)

$e_{sm-}^{\mu k}$ $(e_{sd-}^{\delta k},\ e_{sct-}^{\tau k})$: nombre minimal de lignes de charge utilisables par sm $(sd,\ sct)$ sur la ressource k dans le mode étendu de montage μ, (démontage δ, changement de type τ)

$e^{\mu k}_{sm+}$ ($e^{\delta k}_{sd+}$, $e^{\tau k}_{sct+}$) : nombre maximal de lignes de charge utilisables par sm (sd, sct) sur la ressource k dans le mode étendu de montage μ (démontage δ, changement de type τ)

$p^{\mu ge}_{sm}$ ($p^{\delta ge}_{sd}$) : durée de sm (sd) dans le mode étendu de montage μ (démontage δ) si sm (sd) utilise e lignes de charge de la ressource guidante de montage (démontage) g

$p^{\tau XY ge}_{sct}$: durée de sct dams le mode étendu de changement de type τ pour passer du type X au type Y si sct utilise e lignes de charge de la ressource guidante de changement de type g

Pg^{μ}_{sm}, (Pg^{δ}_{sd}, Pg^{τ}_{sct}) : pool guidant de montage (démontage, changement de type) du mode étendu μ (δ, τ)

$\mathcal{S}m$ ($\mathcal{S}d$, $\mathcal{S}ct$) : ensemble des opérations de montage nécessaires

$\mu(a)$ ($\delta(a)$, $\tau(a)$) : un des modes étendus de montage (démontage, changement de type) correspondant à l'association a

$\tau(k)$: un des modes étendus de changement de types correspondant à la ressource k

$\mathcal{S}$: ensemble des opérations de préparation nécessaires

sct_{ij} : opération de changement de type précédant (i,j) sur l'ensemble des ressources associées $\mathcal{R}b_{ij}$

$p(k)$:opération précédente de (i,j) sur la ressource k

$sd_{p(k)}$: opération de démontage suivant $p(k)$ sur l'ensemble des ressources principales $\mathcal{R}b_{p(k)}$

sct^{k}_{ij} : opération de changement de type de la ressource isolée k suivant $sd_{p(k)}$ et précédant sm_{ij}

sm_{ij} : opération de montage précédant (i,j) sur l'ensemble des ressources principales $\mathcal{R}b_{ij}$

RT^{init}_{k} : type correspondant à l'état initial de la ressource principale k

$\mathcal{R}b^{init}_{k}$: association à laquelle appartient la ressource k à l'état initial

G_{ij} : groupe d'opérations d'exécution permutables contenant (i,j)

G_s : groupe contitué de l'opération de préparation s

r_{ij} : date de début au plus tôt de l'opération d'exécution (i,j)

d_{ij} : date de fin au plus tard de l'opération d'exècution (i,j)

r_s : date de début au plus tôt de l'opération de préparation s

d_s : date de fin au plus tard de l'opération de préparation s

$\mathcal{G} = \{X, U\}$: graphe potentiels tâches pour le calcul des dates de début au plus tôt

O : sommet fictif origine des temps

H : sommet fictif mesurant le plus grand retard

(i, h_{i+1}) : opération fictive ajoutée à la fin de chaque ordre de fabrication i

Chapitre III

$\mathcal{PI}$: position d'insertion dans le graphe $\mathcal{G}$

$\mathcal{G}^*$: graphe résultant de l'insertion de (i,j) dans le graphe $\mathcal{G}$

$(k, l, p(k,l), f(k,l))$: 4-uplet constituant élémentaire d'une position d'insertion $\mathcal{PI}$ où k est une ressource de $\mathcal{R}$, l une ligne de charge de k et $p(k,l)$ et $f(k,l)$ sont des opérations consécutives sur la ligne de charge l

r^*_{ij} : date de début au plus tôt de (i, j) après insertion dans $\mathcal{G}$

d^*_{ij} : date de fin au plus tard de (i, j) après insertion dans $\mathcal{G}$, si l'insertion est admissible

$\Delta d_{max}(i, j, \mathcal{PI})$: impact sur l'admissibilité de l'insertion de (i, j) à la position $\mathcal{PI}$

$\mathcal{PI}(i, j, \mathcal{G})$: ensemble des positions d'insertion valides possibles dans $\mathcal{G}$ pour (i, j)

$\mathcal{PI}_{opt}$: position valide de (i, j) dans $\mathcal{G}$ d'augmentation minimale du plus grand retard

u_α : arc de type ressource de $\mathcal{G}$

$orig_\alpha$: opération origine de l'arc u_α

$dest_\alpha$: opération destination de l'arc u_α

l_α : occupation des ressources de l'arc de type ressource u_α

e^k_α : nombre de lignes de charge occupées par u_α sur k

l^k_α : occupation des lignes de charge de l'arc de type ressource u_α sur la ressource k

e^{kl}_α : variable en 0-1 égale à 1 si u_α occupe la ligne de charge l de la ressource k, égale à 0 sinon

I_α : intervalle associé à u_α

r_α : date de début au plus tôt associée à u_α

d_α : date de fin au plus tard associée à u_α

W_α : largeur de l'intervalle I_α

$\mathcal{U}_{entr}(o)$: ensemble des arcs entrant dans l'opération o

$\mathcal{U}_{sort}(o)$: ensemble des arcs sortant de l'opération o

$\mathcal{G}u_{\mathcal{R}s}$: graphe de compatibilité des arcs de type ressource occupant une partie des lignes de charge de l'ensemble de ressources $\mathcal{R}s$

$\mathcal{C}$: clique de $\mathcal{G}u_{\mathcal{R}s}$

$l_\mathcal{C}$: occupation des ressources de la clique $\mathcal{C}$

e^k_α : nombre de lignes de charge occupées par la clique $\mathcal{C}$ sur k

$l^k_\mathcal{C}$: occupation des lignes de charge de la clique $\mathcal{C}$ sur la ressource k

$e^{kl}_\mathcal{C}$: variable en 0-1 égale à 1 si la clique $\mathcal{C}$ occupe la ligne de charge l de la ressource k, égale à 0 sinon

$r_\mathcal{C}$: date de début au plus tôt associée à la clique $\mathcal{C}$

$d_\mathcal{C}$: date de fin au plus tard associée à la clique $\mathcal{C}$

$W_\mathcal{C}$: largeur de la clique $\mathcal{C}$

$\mathcal{PI}(\mathcal{C})$: position d'insertion caractérisée par la clique $\mathcal{C}$

$dd_o(\mathcal{C})$: clique obtenue par déplacement vers la droite de la clique $\mathcal{C}$

$\mathcal{U}^{\mathcal{R}_{ij}}_{entr}(o)$: sous ensemble des arcs entrant de o occupant une partie des ressources $\mathcal{R}_{ij}$

$\mathcal{U}^{\mathcal{R}_{ij}}_{sort}(o)$: sous ensemble des arcs sortant de o occupant une partie des ressources $\mathcal{R}_{ij}$

$dg_o(\mathcal{C})$: clique obtenue par déplacement vers la gauche de la clique $\mathcal{C}$

$\mathcal{C}^{\mathcal{R}_{ij}}_{init}$: clique maximale initiale sur l'ensemble de ressources $\mathcal{R}_{ij}$

$\mathcal{C}^{\mathcal{R}_{ij}}_{fin}$: clique maximale finale sur l'ensemble de ressources $\mathcal{R}_{ij}$

$\mathcal{U}_{\max}$: ensemble des arcs d'une clique de plus grande date de début au plus tôt

$\mathcal{U}_{\min}$: ensemble des arcs d'une clique de plus petite date de fin au plus tard

$\mathcal{L}_{ij}^{m}$: ensemble des occupations des ressources l_{ij} possibles pour l'opération (i,j) dans le mode étendu m

$p_{ij}(l_{ij})$: durée de (i,j) pour l'occupation l_{ij}

l_{ij+} : borne supérieure de l'occupation de (i,j)

l_{ij-} : borne inférieure de l'occupation de (i,j)

$\mathcal{R}_{ij}^{m+}$: ensemble de toutes les ressources de chaque pool de (i,j) dans un mode étendu m

$\mathcal{C}^{+}$: clique maximale sur l'ensemble de ressources $\mathcal{R}_{ij}^{m+}$

$\mathcal{PI}(b)$: position d'insertion sur les ressources principales

$\mathcal{PI}(c)$: position d'insertion sur les ressources complémentaires

r_{ij}^{*c} : date de début au plus tôt de (i,j) imposée par la position d'insertion sur les ressources principales

d_{ij}^{*c} : date de fin au plus tard de (i,j) imposée par la position d'insertion sur les ressources principales

$\mathcal{G}_{ij}^{*}$: graphe représentant la séquence d'opérations de préparation, et (i,j) à insérer à une position d'insertion $\mathcal{PI}(b)$

$\tilde{\mathcal{G}}_{ij}$: graphe $\mathcal{G}_{ij}^{*}$ privé de l'opération (i,j)

$\tilde{\mathcal{G}}$: graphe issu de l'insertion à une position d'insertion $\mathcal{PI}(b)$ du graphe $\tilde{\mathcal{G}}_{ij}$

$\tilde{u}(i,j)$: arc de $\tilde{\mathcal{G}}_{ij}$ équivalent à la position d'insertion de (i,j)

$\mathcal{C}(c)$: clique suffisante sur les ressources complémentaires de (i,j)

Chapitre IV

$\mathcal{G}'$: graphe issue de la suppression d'une opération d'exécution (i,j)

T_{max} : plus grand retard vrai associé à $\mathcal{G}$

T'_{max} : plus grand retard vrai associé à $\mathcal{G}'$

Introduction

Pour les entreprises qui travaillent à la commande, le contexte actuel impose des délais de plus en plus serrés alors que la demande de plus en plus irrégulière ne permet plus d'anticiper la production. Dans cet environnement incertain, l'intérêt d'un système informatique d'ordonnancement des ateliers est évident pour organiser le travail prévu sur les ressources disponibles, intégrer au mieux dans cette organisation les commandes imprévues et urgentes ainsi que pour maîtriser les conséquences des aléas internes sur des programmes de fabrication de plus en plus tendus. Pour qu'un tel système soit efficacement utilisé, il doit gagner la confiance des différents acteurs de la production dans l'atelier. Cette confiance passe par des caractéristiques essentielles qui sont : la prise en compte des contraintes réelles de l'atelier, la cohérence des solutions proposées, la robustesse face aux aléas, la flexibilité du système pour permettre l'intervention des hommes dans la conduite de l'atelier et donc sa capacité à proposer une aide à la décision. C'est pour permettre la prise en compte de ces multiples caractéristiques que le LAAS développe depuis plusieurs années une approche originale des problèmes d'ordonnancements, la méthode ORABAID (Ordonnancement d'Atelier Basé sur une AIde à la Décision).

La prise en compte des contraintes réelles de l'atelier passe par l'utilisation de modules intégrant des caractéristiques telles que les ressources disjonctives et cumulatives (opérateurs par exemple), ressources multiples pour réaliser une opération, calendrier de disponibilité des ressources, gammes non linéaires, préparation des ressources, etc... Cette prise en compte permet au système d'ordonnancement d'établir des plans de fabrication réalistes, conduisant à des prévisions fiables des dates de livraison des commandes à réaliser. La prise en compte simultanée de ces multiples caractéristiques est peu fréquente dans les travaux de recherche en ordonnancement.

Pour assurer la cohérence des solutions proposées, une faible perturbation dans les contraintes ne doit pas entraîner une forte pertubation dans les séquences proposées. Les approches d'ordonnancement pratiques uniquement basées sur des règles de priorité ne permettent pas d'assurer cette cohérence.

Pour assurer la robustesse face aux perturbations, la méthode ORABAID s'appuie sur l'élaboration de plusieurs ordonnancements compatibles avec les contraintes au lieu d'un ordonnancement unique, ce qui dégage de la flexibilité pour faire face aux aléas internes de l'atelier.

Pour permettre l'intervention des responsables et des opérateurs de l'atelier dans sa conduite, la méthode ORABAID ne se limite pas à proposer un ensemble d'ordonnancements, elle accompagne aussi le décideur dans l'exploitation de cette proposition par une aide à la décision. Elle fournit des indicateurs pour exploiter au mieux cet ensemble d'ordonnancements. Elle permet une actualisation permanente de cet ensemble en fonction de la situation réellement constatée dans l'atelier, donc en particulier une précision fiable des délais de commande à réaliser. En présence d'aléas, la méthode ORABAID permet si nécessaire de proposer des actions correctives pour tenir les délais. Enfin, en présence ou non d'aléas, elle permet au décideur de tester les conséquences de certaines décisions qu'il envisage, contribuant ainsi à une meilleure rationalisation du processus décisionnel.

Pour réaliser l'ensemble des fonctionnalités énumérées ci-dessus, la méthode ORABAID s'appuie sur la caractérisation d'un ensemble d'ordonnancements sous la forme d'une séquence de groupes d'opérations permutables telle que les opérations au sein d'un même groupe utilisent simultanément exactement les mêmes ressources et sont totalement permutables vis-à-vis du respect de l'ensemble des contraintes retenues. Un système interactif d'aide à la décision pour l'ordonnancement en temps réel est ensuite utilisé pour exploiter cette séquence de groupes et l'adapter en fonction des aléas survenus.

Les études successives effectuées pour développer cette méthode [Demmou, 1977], [Thomas, 1980], [Le Gall, 1989], [Billaut, 1993], ont progressivement intégré dans le modèle de l'atelier des contraintes de plus en plus complexes (ensembles de ressources utilisables indifféremment pour exécuter la même opération, ressources multiples simultanément nécessaires à la réalisation d'une opération, ressources cumulatives pouvant exécuter plusieurs opérations simultanément, gammes non linéaires...). Ces travaux, ainsi que le logiciel associé ont servi de base à partir de 1985, au développement du logiciel ORDO^R d'ordonnancement en temps réel d'atelier, avec la société SYSECA SA tout d'abord, puis par la suite avec la Société CABINET VILLAUMIE SA.

L'objectif de l'étude présentée dans cette thèse est de prendre en compte les activités de préparation des ressources qui peuvent être nécessaires avant d'exécuter une opération d'une gamme. Ce problème est considéré dans certains travaux sur l'ordonnancement [Gavett, 1965] , [Proust *et al.*, 1991] [Guinet, 1991] [Monma et Potts, 1993] [Schutten, 1996] [Ovacik et Uzsoy, 1994b]... Toutefois, il apparait utile de rechercher une modélisation de cette activité de préparation qui corresponde à la réalité des ateliers, en relation avec le contexte multi-ressources. L'ensemble des contraintes prises en compte devenant très complexe, l'accent a été mis dans ce travail sur la maîtrise de l'explosion combinatoire des procédures, en particulier pour la prise en compte du contexte cumulatif multi-ressources.

Dans le premier chapitre, la fonction ordonnancement est positionnée au sein du système de gestion de la production. Les principaux problèmes d'ordonnancement sont présentés. Les différentes approches de résolution de ces problèmes sont décrites, ce qui permet de situer celle retenue dans cette étude : la méthode ORABAID. Les principes et les évolutions de cette méthode d'ordonnancement basée sur une aide à la décision sont rappelés. Le contexte de l'étude présentée dans ce mémoire est défini.

Dans le second chapitre, un état de l'art présente la prise en compte de temps de préparation dans les problèmes d'ordonnancement à travers différents modèles et différentes approches de résolution rencontrés dans la littérature. Le modèle retenu dans cette étude est présenté et justifié. Le concept d'activité de préparation, mettant en jeu les notions de ressources principales et complémentaires, est défini. La notion de groupe d'opérations permutables est étendue à ce contexte et la caractérisation d'une séquence de groupes au moyen d'un graphe potentiels-tâches est présentée.

Dans le troisième chapitre une approche pour la résolution du problème d'insertion d'une opération d'exécution dans un ordonnancement est proposée. L'objectif est la minimisation de l'impact de l'insertion sur l'admissibilité. Un algorithme polynomial et optimal basé sur des règles de dominance est présenté dans le cas cumulatif multi-ressources sans préparation. L'algorithme optimal est étendu successivement aux problèmes comportant des possibilités multiples d'affectation, aux ressources comportant une activité de préparation et enfin à la combinaison de ces deux caractéristiques. Une heuristique pour l'insertion est présentée dans le cas où les opérations d'exécution et l'activité de préparation nécessitent des ressources complémentaires.

Dans le quatrième chapitre, la validation expérimentale de la procédure d'insertion du chapitre précédent est effectuée par l'intermédiaire de son intégration dans une méthode de voisinage pour l'amélioration d'un ordonnancement. Une méthode de type tabou est présentée dans un contexte sans préparation. Cette méthode est testée sur des problèmes connus d'ordonnancement de projet à mode simple et à modes multiples. La méthode tabou est ensuite étendue au contexte général avec une activité de préparation. Des résultats expérimentaux sont présentés sur des problèmes générés aléatoirement.

Dans le cinquième et dernier chapitre, l'extension de la méthode ORABAID à la prise en compte des activités de préparation est présentée et les procédures définies dans les études précédentes en l'absence de temps de préparation sont améliorées par l'utilisation des algorithmes polynomiaux d'insertion présentés au chapitre III. L'extension concerne la procédure de génération d'une séquence de groupes d'opérations d'exécution permutables, la procédure de recherche de l'admissibilité de la séquence de groupes et le système interactif d'aide à la décision pour l'exploitation en temps réel de cette séquence. En particulier, de nouveaux états, événements et décisions liés aux entités composant l'atelier sont définis. Enfin, la nouvelle version du logiciel ORDO^R intégrant les activités de préparation est présentée.

Première partie

Approche du problème d'ordonnancement

Chapitre I

Approche pour l'ordonnancement

Résumé

La fonction ordonnancement est positionnée au sein du système de gestion de la production. Les principaux problèmes d'ordonnancement sont présentés. Les différentes approches de résolution de ces problèmes sont décrites, ce qui permet de situer celle retenue dans cette étude : la méthode ORABAID. Les principes et les évolutions de cette méthode d'ordonnancement basée sur une aide à la décision sont rappelés. Le contexte de l'étude présentée dans ce mémoire est défini.

I.1 Présentation générale des problèmes d'ordonnancement

I.1.1 La fonction ordonnancement au sein du système de gestion de la production

Dans une entreprise productrice de biens ou de services, on peut distinguer le système de production du système de gestion. Le rôle du système de gestion est de "commander" le système de production en respectant un ensemble de contraintes en vue d'atteindre des objectifs définis. Il s'agit dans ce paragraphe de définir le type de systèmes de production concerné par cette étude et de positionner la fonction ordonnancement au sein du système de gestion. Ce paragraphe s'appuie sur [Giard, 1988], [Legait, 1992], [Hayes et Wheelwright, 1979], [Le Moal et Tarondeau, 1979] et [Hétreux, 1996].

I.1.1.a) Le système de production

Le système de production rassemble l'ensemble des moyens qui permettent la transformation de ressources en produits ou en services. Deux classifications des systèmes de production trouvées dans la littérature [Giard, 1988] permettent de situer précisément le contexte du travail présenté ici.

La première typologie sépare les systèmes fonctionnant à la commande de ceux basés sur une production pour stock.

- Dans les *systèmes basés sur une production pour stock*, la fabrication est déclenchée par anticipation d'une demande solvable. Pour pouvoir produire pour stock, il est nécessaire d'une part que l'éventail des produits finis visés soit restreint et d'autre part que la demande de chaque produit soit suffisamment importante et prévisible. En outre, le choix de ce type de production n'est intéressant que si le cycle de production est long par rapport au délai séparant la prise de commande de la livraison ou en cas de forte demande saisonnière.

- Dans les *systèmes basés sur une production à la commande*, tout ou partie de la fabrication (et/ou assemblage) est déclenchée par la commande ferme d'un client. Ils concernent les entreprises proposant une grande variété de produits dont la demande est aléatoire ou celles qui ne définissent leurs produits qu'à partir de demandes précises de clients. C'est généralement le cas des entreprises sous-traitantes.

Une deuxième classification est centrée sur la nature et le volume des produits fabriqués.

- *Les systèmes à production continue* ou *process*:
  Dans ces systèmes la matière circule en flux continu. L'attente entre deux ressources est exclue ou très limitée. Ce type de systèmes concerne surtout les industries dont la production nécessite la manipulation de matières liquides ou gazeuses.

- *Les systèmes de production unitaire* ou *projet*:
  La taille du produit ou la demande imposent une production de très faible quantité et la tâche principale consiste à réunir les moyens nécessaires au bon moment et au bon endroit. Dans ce type de structure, le problème majeur est celui d'un arbitrage entre une recherche de coût compétitif et le respect des délais. Des techniques d'ordonnancement de projet (voir paragraphe I.1.2.a)) sont utilisées.

- *Les systèmes de production en grande série* ou *masse*:
  Dans le cas où le nombre de produits similaires à fabriquer dans les mêmes délais est important ou si l'éventail de produits différents est restreint, il peut être rentable de constituer des chaînes de fabrication. L'ordre de passage des produits sur les ressources étant toujours le même, celles-ci peuvent être placées dans un ordre fixe dépendant du produit à fabriquer. Le principal problème réside en l'équilibrage de la chaîne. Si celui-ci est correctement réalisé, de tels systèmes se caractérisent par une très bonne utilisation des ressources et un faible pourcentage de temps perdu en attente pour les produits en fabrication.

- *Les systèmes de production en petite ou moyenne série* ou *atelier*:
  Contrairement à la catégorie précédente, il s'agit d'ateliers dans lesquels la diversité des produits et le faible volume des demandes ne permettent pas une spécialisation

des moyens de production. Les différents produits suivent leur propre chemin sur des ressources communes, souvent regroupées par fonctionnalité équivalente. Cette organisation est celle de la plupart des PME manufacturières et de sous-traitance.

Dans la suite de ce mémoire on s'intéresse principalement aux systèmes de production **à la commande** et en **petite ou moyenne série**.

Un exemple de ce type d'entreprises peut donner une idée des enjeux d'un système de gestion de production adapté à la conjoncture actuelle. La figure I.1 montre l'évolution de l'horizon des commandes selon Rémi Laurent, PDG de la société SIOBRA, spécialisée en injection du Zamak à Arbois (Jura) [Laurent, 1997]. Pour cette société, les délais de livraison se chiffraient autrefois en semaines, et la demande régulière permettait d'anticiper la production par l'intermédiaire de commandes prévisionnelles (partie gauche de la figure). Aujourd'hui la réduction des délais et l'incertitude liée au contexte économique imposent de gérer les commandes au jour le jour sans possibilité réelle d'anticipation (partie droite de la figure). Le maître mot est la réactivité.

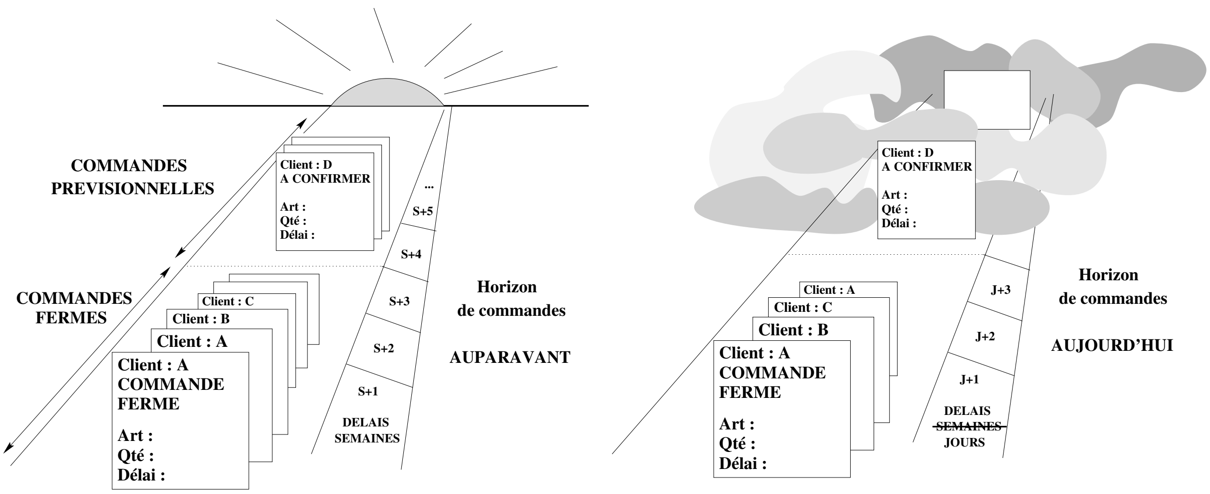


FIG. I.1 – *Evolution de l'horizon des commandes d'après [Laurent, 1997]*

I.1.1.b) Le système de gestion de production

Le rôle de ce système est la commande du système de production par l'intermédiaire de décisions qu'on classe habituellement en trois catégories :

- Les *décisions stratégiques* se traduisent par la formulation de la politique à long terme de l'entreprise. Elles concernent ses objectifs principaux. Ce sont par exemple des décisions d'investissement ou d'embauche.

- Les *décisions tactiques* correspondent à un ensemble de décisions à moyen terme, parmi lesquelles la planification de la production en fonction de la demande ou de la stratégie de gestion des stocks. Elles sont contraintes par les décisions stratégiques. Elles définissent habituellement le volume à produire et les délais d'obtention souhaités ainsi que l'organisation des moyens.

- Les *décisions opérationnelles* assurent l'exécution de la planification retenue et la flexibilité quotidienne nécessaire pour faire face aux aléas. Ces décisions concernent essentiellement l'ordonnancement qui sera présenté dans le paragraphe I.1.2.

Comme on peut le constater, ces décisions sont organisées de manière hiérarchique en trois niveaux. La décision d'un niveau donné devient une contrainte pour le niveau inférieur. L'impossibilité de respecter ces contraintes à un niveau entraîne la remise en cause des décisions du niveau supérieur, éventuellement par un processus de négociation de contraintes [Huguet, 1994].

On peut remarquer que cette structure classique à trois niveaux est à reconsidérer pour le cas des entreprises travaillant à la commande en environnement incertain. La planification de la production (correspondant au niveau tactique) est alors difficilement réalisable faute d'information suffisante sur les commandes prévisionnelles. Les délais de livraison resserrés imposent la prise en compte de chaque commande nouvelle directement au niveau opérationnel.

Dans ce mémoire, nous nous intéressons uniquement au **niveau opérationnel**, et plus particulièrement à **l'ordonnancement à court terme d'un atelier de production** à la commande et en petite ou moyenne série.

I.1.2 Les problèmes d'ordonnancement

L'objectif de ce paragraphe est de présenter les différents problèmes d'ordonnancement et les approches classiques de résolution, afin de mieux situer la nôtre.

I.1.2.a) Présentation des différents problèmes d'ordonnancement

Dans le système de production, le problème d'ordonnancement consiste à organiser dans le temps l'exécution d'opérations interdépendantes à l'aide de ressources disponibles en quantités limitées pour réaliser un plan de production [Erschler *et al.*, 1992].

On peut également trouver des problèmes d'ordonnancement dans d'autres systèmes (informatique, gestion de projets). Aussi, il est préférable de définir le problème d'ordonnancement hors de tout contexte d'application en se basant sur les concepts de tâche, de ressource, de contrainte et d'objectif :

> "Ordonnancer un ensemble de tâches, c'est programmer leur exécution en leur allouant les ressources requises et en fixant leur date de début".[GOThA, 1993]

Les tâches sont soumises à des *contraintes* et un ordonnancement, c'est-à-dire une solution au problème ainsi défini, est évalué vis-à-vis d'un ou plusieurs *objectif(s)* à satisfaire.

Notation adoptée Pour décrire les différents problèmes d'ordonnancement, plusieurs notations ont été introduites. Nous avons retenu la notation en trois champs $\alpha|\beta|\gamma$ très utilisée dans la littérature [Lawler *et al*, 1989], [Blazewicz *et al.*, 1996], [Pinedo, 1995]. Le champ α décrit les caractéristiques des ressources. Le champ β décrit les contraintes et les caractéristiques des tâches. Le champ γ décrit le ou les objectifs sous la forme d'un critère à optimiser.

Ordres de fabrication, opérations Par référence au système de production, on utilise dans la suite le terme **opération** à la place des termes *tâche* ou *activité* rencontrés dans la littérature. De même on utilise le terme d'**ordre de fabrication** à la place des termes *travail* ou *projet*. Un ordre de fabrication peut correspondre à un *lot* de produits à fabriquer.

A un ordre de fabrication est associé un ensemble d'opérations à exécuter à l'aide de ressources. Les opérations d'un ordre de fabrication sont liées par des relations de précédence pouvant être modélisées par un graphe potentiels-tâches. Lorsque ce graphe est systématiquement réduit à une seule opération, on parle de problème *mono-opération*. Les termes ordre de fabrication et opération peuvent alors être employés indifféremment. Dans les autres cas, on parle de problèmes *multi-opérations*. Un ordre de fabrication i peut posséder une *date de début au plus tôt* r_i, une date de fin au plus tard souhaitée (*date de livraison*) d_i, une date de fin au plus tard impérative (*deadline*) $\tilde{d}_i$, une priorité w_i représentant l'urgence relative de i.

Une opération (i, j) est caractérisée par sa durée p_{ij} encore appelée temps opératoire où temps d'exécution. Ce temps correspond au temps d'exécution de l'ensemble des éléments du lot. Additionnellement, des temps de préparation précédant et/ou suivant l'exécution peuvent être nécessaires. Ces temps correspondent par exemple au montage et au démontage d'outils, au nettoyage des machines. Ils seront décrits en détail dans le paragraphe II.1.

Ressources On distingue trois catégories de ressources [Blazewicz *et al.*, 1996] :

– Les ressources *renouvelables* deviennent à nouveau utilisables après avoir exécuté une opération. Parmi ces ressources, on distingue les ressources *disjonctives* qui ne peuvent exécuter qu'une opération à un instant donné (par exemple : une machine), des ressources *cumulatives* qui peuvent exécuter un nombre limité d'opérations simultanément (par exemple : une équipe d'opérateurs). Le nombre maximum d'opérations exécutables par une ressource cumulative est appelé *capacité* où *quantité disponible* de la ressource.

– Les ressources *consommables* ne peuvent être réutilisées après l'exécution d'une opération (par exemple matière première, budget).

– Les ressources *doublement limitées* combinent les contraintes liées aux deux catégories précédentes. Par exemple, une pompe à carburant est à la fois une ressource disjonctive (la pompe) et une ressource consommable (le carburant).

Les deux derniers types de ressource ne seront pas étudiés dans ce mémoire.

D'autre part, on peut associer à une ressource des caractéristiques liées à sa préparation en vue de l'exécution d'une opération, décrites précisément dans le paragraphe II.1.

Contraintes Pour une description des principales contraintes intervenant dans les problèmes d'ordonnancement, on se réfère à [Erschler, 1976]. Dans la notation adoptée, ces contraintes sont indiquées dans le champ β.

Les *contraintes internes* auxquelles sont soumises les opérations, parfois appelées *contraintes absolues*, sont locales au problème d'ordonnancement et sont de deux types :

i) contraintes liées à l'interdépendance des opérations : on distingue plusieurs catégories de contraintes de ce type :

- le respect des contraintes de précédence entre opérations d'un même ordre de fabrication (obligatoire pour tous les problèmes multi-opérations). Ces problèmes sont notés $prec$ si le graphe des précédences entre opérations d'un même ordre de fabrication est quelconque (gamme quelconque), $in-tree$ si le graphe de précédence est tel qu'une opération ne peut avoir qu'une suivante (gamme convergente), $out-tree$ si l'opération ne peut avoir qu'une précédente (gamme divergente), $chain$ si une opération ne peut avoir qu'une suivante et une précédente (gamme linéaire).

- le respect des contraintes de précédence éventuelles entre opérations d'ordres de fabrication différents (problèmes notés $j-prec$, $j-in-tree$, etc...). Ces contraintes ne se différencient des précédentes que pour les problèmes multi-opérations.

- les décalages temporels minimaux entre opérations successives d'un même ordre de fabrication (problèmes notés a_{ijil} : décalage temporel minimal entre la date de fin de (i, j) et la date de début de (i, l)). Si $a_{ijil} > 0$, on modélise par exemple les temps de transport d'un lot d'une machine à une autre. Si $a_{ijil} < 0$, on modélise le chevauchement d'opérations, c'est-à-dire la possibilité de commencer une opération avant la fin de l'opération précédente dans la gamme. Ce cas arrive fréquemment lorsqu'un sous-lot peut être transféré vers la machine suivante avant la fin du lot complet. Dès l'arrivée du premier sous-lot, l'opération suivante peut commencer.

- les décalages temporels maximaux entre opérations successives d'un même ordre de fabrication (problèmes notés A_{ijil} : décalage temporel maximal entre la date de fin de (i, j) et la date de début de (i, l)). Si $A_{ijil} = 0$, on a le cas particulier d'impossibilité d'attente entre deux opérations successives d'un même ordre de fabrication (problèmes noté $no - wait$). Ces contraintes correspondent par exemple aux systèmes de production de type process (voir paragraphe I.1.1.a)).

- la possibilité d'interrompre une opération (problèmes notés $pmtn$). Les opérations peuvent être exécutées par morceaux, les morceaux ne pouvant pas être réalisés simultanément, ce qui arrive fréquemment dans les systèmes informatiques. On parle de problèmes préemptifs ou non-préemptifs.

- la possibilité de découpage en sous-lots (*lot streaming*, problèmes notés $split$). Un ordre de fabrication peut à tout moment être découpé en sous ordres de fabrication de gamme identique et suivant leur propre chemin dans l'atelier.

ii) contraintes liées au partage des ressources disponibles en quantités limitées.

- Les *contraintes disjonctives* sont liées à l'utilisation par plusieurs opérations d'une même ressource disjonctive. Deux opérations utilisant une ressource disjonctive ne peuvent être exécutées simultanément.

- Les *contraintes cumulatives* sont liées à l'utilisation simultanée d'une même ressource cumulative par plusieurs opérations. L'ensemble des opérations utilisant simultanément une ressource cumulative doit occuper une quantité de la ressource inférieure à sa capacité.

- Les *contraintes multi-ressources* imposent à une opération d'utiliser simultanément plusieurs ressources pour son exécution (par exemple une machine et un opérateur). Ces ressources peuvent être de type disjonctif ou cumulatif. Pour ces problèmes, on introduit la notation $res\lambda\sigma\rho$ où λ est le nombre maximum de ressources requises simultanément par toute opération, σ est une borne supérieure de la capacité des ressources et ρ est la quantité maximum requise par toute opération sur chaque ressource. Par exemple, le cas disjonctif multi-ressources est noté $res \circ 11$. Le cas cumulatif mono-ressource est noté $res1 \circ \circ$.

- On peut considérer enfin qu'un calendrier de fonctionnement est associé à chaque ressource. Un calendrier est défini par un ensemble de périodes. La quantité de ressource disponible est constante au cours d'une période mais peut varier d'une période à l'autre. Les problèmes où les ressources ont des calendriers seront notés Cal.

Les *contraintes externes* expriment des décisions du niveau supérieur au problème d'ordonnancement et sont aussi appelées *contraintes relatives*. Il s'agit essentiellement des dates de fin au plus tard d_i des ordres de fabrication.

Un ordonnancement respectant les contraintes internes est dit *réalisable*. Un ordonnancement respectant les contraintes internes et externes est dit *admissible*.

Aperçu des problèmes fréquemment rencontrés Dans la littérature, on trouve tout d'abord les problèmes d'ateliers classiques, composés uniquement de ressources disjonctives. On peut les classer en cinq catégories, à l'aide du champ α qui donne des renseignements sur la nature des gammes et l'environnement lié aux ressources.

- *machine unique* (noté $\alpha = 1$). Dans ce problème mono-opération, une seule ressource disjonctive est disponible pour exécuter toutes les opérations.

- *machines parallèles*: ce problème mono-opération consiste à ordonnancer un ensemble de n opérations sur m machines parallèles de telle sorte qu'une opération nécessite pour son exécution une et une seule machine à choisir parmi les m machines. Un double problème d'affectation des opérations aux machines et d'ordonnancement des opérations sur les machines est ainsi défini. On distingue :

  - le problème à machines identiques (noté Pm) pour lequel l'opération i a la même durée p_i sur toutes les machines,

  - le problème à machines uniformes (Qm) pour lequel la durée p_{ik} d'une opération i sur la ressource k est égale à p_i/v_k où v_k est la vitesse de la ressource k.

  - le problème à machines indépendantes (Rm) pour lequel la durée p_{ik} d'une opération i sur la machine k est une donnée du problème.

- Le problème d'*atelier à cheminement quelconque* (*open shop*, noté Om) consiste à ordonnancer un ensemble de n ordres de fabrication multi-opérations sur un ensemble de m ressources disjonctives. Les opérations de chaque ordre de fabrication doivent être exécutées sur des machines différentes mais dans un ordre quelconque.

- Le problème d'*atelier à cheminement unique* (*flow shop*, noté Fm) consiste à ordonnancer un ensemble de n ordres de fabrication multi-opérations sur un ensemble de m ressources disjonctives dans le cas où la gamme de chaque ordre de fabrication est linéaire et identique pour tous les ordres de fabrication. Pour ce problème, on considère souvent par extension du cas à deux machines que la séquence des opérations des différents ordres de fabrication doit être la même sur chaque machine (problème dit de permutation). Ce cas étant le plus fréquemment rencontré, le cas plus général est noté ici $non - prmu$.

- Le problème d'*atelier à cheminements multiples* (*job shop*, noté Jm) consiste à ordonnancer un ensemble de n ordres de fabrication multi-opérations sur un ensemble de m ressources disjonctives sachant que la gamme de chaque ordre de fabrication est linéaire et différente pour au moins un des ordres de fabrication. Une extension classique de ce problème est le cas où les ordres de fabrication peuvent revenir plusieurs fois sur la même machine (noté $recrc$ dans le champ β).

De nombreuses extensions des modèles de base ont été étudiées dans le but de les rapprocher de la réalité des ateliers. On parle alors de problèmes généralisés.

Une partie de ces extensions concerne l'introduction dans le problème d'atelier à cheminements multiples de contraintes multi-ressources, de contraintes cumulatives et de gammes non-linéaires (qu'on peut noter $Jm|res \circ \circ\circ, prec|\gamma$). On peut rapprocher ce problème du problème d'ordonnancement de projet à moyens limités noté $P\infty|res \circ\circ\circ, prec|\gamma$ [Lawler *et al*, 1989].

Une autre partie des extensions concerne l'introduction de la flexibilité de ressource dans les problèmes multi-opérations. Dans le problème d'atelier à cheminement unique flexible ou hybride, noté FFm, un ensemble de machines parallèles est disponible à chaque étage de l'atelier pour exécuter les opérations [Botta, 1996], [Vignier, 1997]. Une telle extension est envisagée pour le problème d'atelier à cheminements multiples (notée FJm) [Le Gall, 1989], [Brucker et Schlie, 1990]. Nous appelons *pool* un ensemble des ressources possibles pour exécuter une opération.

L'introduction de la flexibilité de ressource dans les problèmes multi-ressources cumulatifs peut être effectuée de deux manières différentes :

- utilisation des pools de ressources [Billaut, 1993], [Dauzère Péres et al, 1996] ($FJm|res \circ \circ\circ, prec|\lambda$) : plusieurs pools sont définis pour une opération et une ressource doit être sélectionnée dans chaque pool pour exécuter cette opération.

- utilisation des modes multiples. Dans le problème d'ordonnancement de projet à moyens limités et modes multiples [Kolish, 1995], (noté ici $P\infty|res \circ\circ\circ\circ, prec|\lambda$), la sélection d'un mode (quatrième champ de la notation res) pour chaque opération détermine l'ensemble des ressources et la quantité nécessaire à son exécution.

Les modèles avec pools et les modèles avec modes seront comparés dans le paragraphe II.2.1.a) pour justifier notre choix d'un modèle mixte.

Dans la figure I.2 les différents modèles évoqués sont rappelés. Un modèle est relié à un autre par un arc si le modèle destination est un cas particulier du modèle origine.

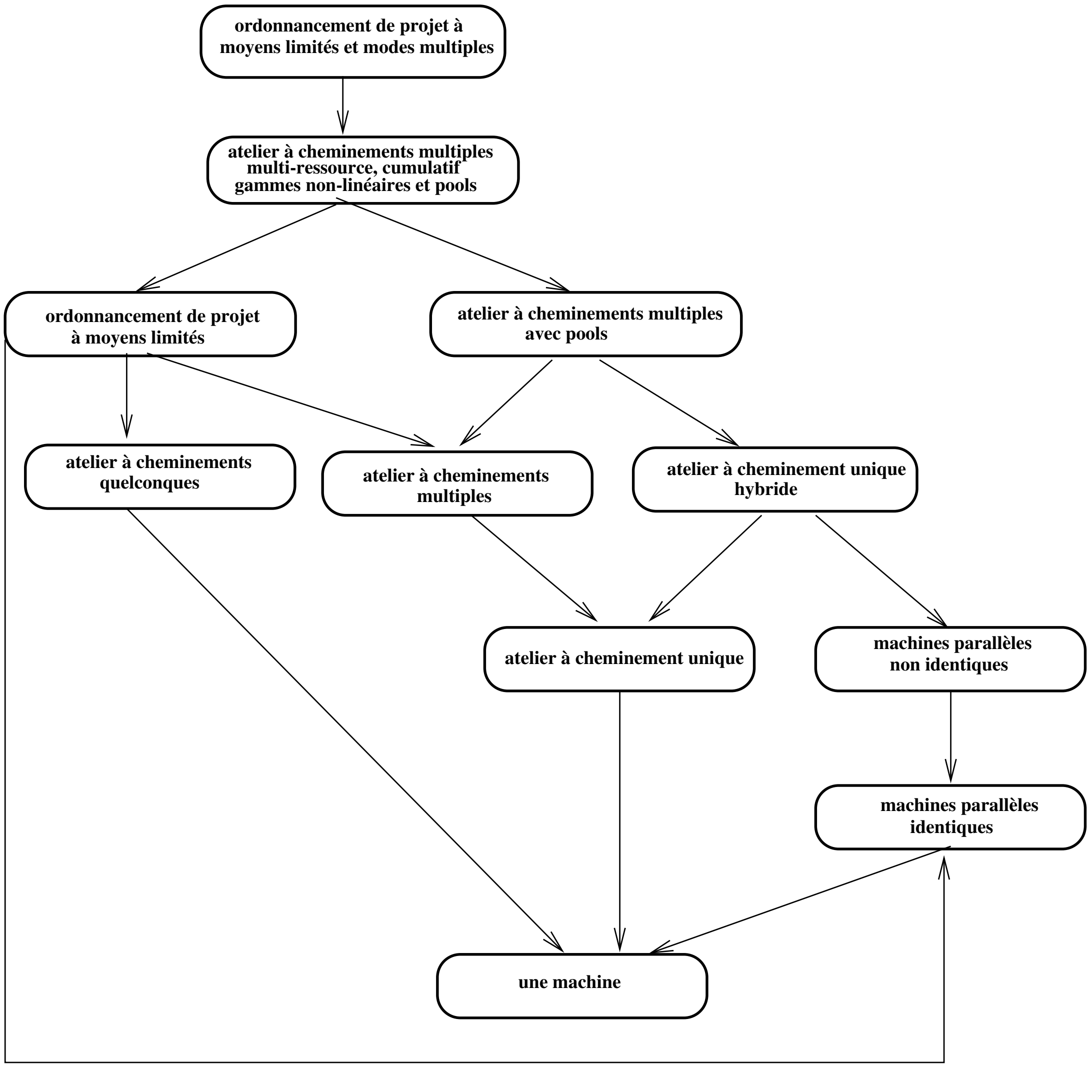


FIG. I.2 – *Hiérarchie des modèles*

D'autres extensions concernent la prise en compte de l'impossibilité d'attente entre opérations d'un même ordre de fabrication ($no-wait$). Combinées à la prise en compte de durées variables d'exécution et de ressources additionnelles de transport (robots), elles modélisent les problèmes d'ateliers de traitement de surface (*hoist scheduling problems*) dont la formulation la plus simple peut être notée $Fm|res1r1, r_j, a \leq p_j \leq b, no-wait|\lambda$ où r est le nombre maximum de robots utilisables simultanément pour le transport [Manier et Baptiste, 1994].

D'autres extensions concernent la préparation des ressources, énumérées dans le paragraphe II.1.

Objectifs L'objectif le plus souvent rencontré dans la littérature est l'obtention d'un ordonnancement optimisant un certain critère. Le critère est indiqué dans le champ γ de la notation adoptée. Le plus souvent, le critère est fonction d'une des données suivantes (ou de plusieurs) associées à chacun des n ordres de fabrication :

- la date de fin C_i,
- la durée de séjour $F_i = C_i - r_i$,
- le retard algébrique $L_i = C_i - d_i$,
- le retard vrai $T_i = \max(0, C_i - d_i)$,
- la pénalité unitaire de retard U_i égale à 1 si $T_i > 0$ et égale à 0 sinon,
- l'avance $E_i = \max(0, d_i - C_i)$.

Parmi les critères fréquemment rencontrés, on trouve :

- la durée totale $C_{max} = \max_i C_i$,
- le plus grand retard vrai $T_{max} = \max_i T_i$,
- la somme pondérée des retards vrais $\sum_i w_i T_i$,
- le nombre d'ordres de fabrication en retard $N_T = \sum_i U_i$
- le plus grand retard algébrique $L_{max} = \max_i L_i$,
- la durée moyenne de séjour $\overline{F} = \frac{\sum_i C_i}{n}$,
- la somme pondérée des retards et des avances $\sum_i \alpha_i E_i + \beta_i T_i$ etc...

On dit qu'un critère est régulier s'il évolue de façon non décroissante lorsque les dates de fin des ordres de fabrication augmentent. Les critères présentés ci-dessus sont tous réguliers à l'exception de la somme pondérée des retards et des avances.

Un certain nombre d'approches issues de [Erschler, 1976], dont celle présentée dans ce mémoire, transforment les critères en contraintes externes et se donnent pour objectif de caractériser les ordonnancements admissibles (ou tout au moins un sous-ensemble de ceux-ci).

I.1.2.b) Propriétés des ordonnancements

Pour un problème donné, il peut être intéressant de considérer certains sous-ensembles de l'ensemble des ordonnancements réalisables, possédant des propriétés communes. Certaines de ces propriétés donnent à ces sous-ensembles un caractère de dominance vis à vis de certains critères ou de l'admissibilité selon l'objectif considéré. On dit qu'un sous-ensemble est *dominant* vis à vis d'un critère donné s'il contient au moins un ordonnancement optimal. On dit qu'un sous-ensemble d'ordonnancements est *dominant* [Fontan, 1980] vis à vis de l'admissibilité si l'existence d'un ensemble d'ordonnancements admissibles implique l'appartenance d'au moins un de ces ordonnancements au sous-ensemble dominant.

On peut définir trois propriétés correspondant à trois sous-ensembles classiques des ordonnancements réalisables.

- Dans un ordonnancement *semi-actif* (ou calé au plus tôt), aucune opération ne peut être décalée localement vers la gauche dans le respect des contraintes de précédence.

- Dans un ordonnancement *actif*, aucune opération ne peut être démarrée plus tôt sans décaler vers la droite une autre opération. L'ensemble des ordonnancements actifs est un sous-ensemble de l'ensemble des ordonnancements semi-actifs.

- Dans un ordonnancement *sans délai*, aucune ressource n'est laissée inactive alors qu'elle pourrait commencer l'exécution d'une opération. L'ensemble des ordonnancements sans-délai est un sous-ensemble de l'ensemble des ordonnancements actifs.

Quel que soit le problème, les ordonnancements semi-actifs et actifs sont dominants si le critère est régulier. Ce n'est pas le cas pour les ordonnancements sans délai [Baker, 1974], [Sprecher *et al.*, 1995].

Lorsqu'on considère par exemple la recherche d'ordonnancements admissibles avec des dates de fin au plus tard des ordres de fabrication comme contraintes externes, les ordonnancements semi-actifs et actifs sont dominants vis-à-vis de l'admissibilité.

I.1.2.c) Les approches de résolution

On peut tout d'abord classer les approches par l'hypothèse qu'elles font des connaissances par le système des données du problème.

Les approches d'ordonnancement *prévisionnel* considèrent uniquement le problème d'ordonnancement *statique*, dont toutes les données sont supposées connues à l'avance.

Les approches d'ordonnancement *réactif* considèrent éventuellement un problème statique dans un premier temps puis proposent des méthodes pour faire face à des événements générant des perturbations, comme l'arrivée *dynamique* des ordres de fabrication, les pannes de ressource, l'absence de personnel,...

Ordonnancement prévisionnel En ordonnancement prévisionnel, les méthodes classiques ont pour but de générer un ordonnancement optimal, c'est-à-dire minimisant un des critères présentés (ou une combinaison de plusieurs critères). Les problèmes d'ordonnancement ainsi définis sont des problèmes d'optimisation combinatoire. La plupart d'entre eux appartient à la classe des problèmes *NP-difficiles* (voir [Lawler *et al*, 1989]) qui ne peuvent être résolus au moyen d'un algorithme polynomial. Aussi, à coté des algorithmes classiques de l'optimisation combinatoire ont été développées un grand nombre d'*heuristiques*, algorithmes approchés souvent polynomiaux, qui permettent d'obtenir un ordonnancement pour lequel la valeur du critère n'est *pas très* éloignée de la valeur optimale. Les méthodes exactes et les méthodes heuristiques classiques sont successivement présentées.

- Méthodes exactes.
  Elles ont pour but de rechercher un ordonnancement optimal, c'est-à-dire minimisant un des critères présentés (ou une combinaison de plusieurs critères).

  - La **programmation linéaire** : les problèmes d'ordonnancement peuvent pour la plupart être modélisés sous la forme d'un programme en variables mixtes (entières et 0-1) et résolus en théorie à l'aide de solveurs. En pratique, cette méthode, utilisée de façon isolée, permet de résoudre uniquement des problèmes de très petite taille (voir par exemple les tests effectués par [Portmann et Bolzoni, 1994] sur le problème à une machine).
  - La **programmation dynamique** : cette méthode d'énumération implicite est applicable uniquement si le problème est décomposable en phases, et si le critère possède certaines propriétés, vérifiées en particulier s'il a une forme additive i.e. s'il peut être défini comme une somme de fonctions de la date de fin de chaque opération. En règle générale, à partir d'un ordonnancement partiel, une formulation donne la valeur du critère comme une fonction récurrente. On trouve la valeur optimale en énumérant un ensemble d'ordonnancements partiels, qu'on complète à chaque étape par une opération (décision). Grâce à la formulation récurrente, on réduit l'énumération. Le résultat ne fournissant que la valeur optimale du critère, il est nécessaire d'effectuer un parcours arrière de l'ensemble des décisions effectuées qui ont mené à cette valeur pour obtenir l'ordonnancement optimal. (voir par exemple [Schrague et Baker, 1978]).
  - Les **procédures par séparation et évaluation** (PSE) explorent l'ensemble des ordonnancements en construisant une arborescence dont chaque nœud correspond à un sous-ensemble d'ordonnancements. Le principe de séparation sépare un nœud en plusieurs fils correspondant à des sous-ensembles disjoints du nœud père (par exemple, sélection d'une disjonction à arbitrer). Pour chaque noeud exploré on calcule une borne inférieure (évaluation par défaut) et une borne supérieure (évaluation par excès) du critère. Une feuille correspond à un unique ordonnancement. L'efficacité de ces méthodes dépend fortement de la qualité de la borne inférieure obtenue. Si à un nœud donné la borne inférieure est supérieure à la meilleure borne supérieure trouvée jusqu'ici on peut élaguer la partie de l'arbre issue de ce nœud. Parmi les procédures les plus efficaces développées, on peut citer [Carlier, 1982], [Carlier et Pinson, 1989], [Demeulemeester et Herroelen, 1992] respectivement pour le problème à une machine, le problème d'atelier à cheminements multiples et le problème d'ordonnancement de projets à moyens limités.

- Méthodes heuristiques

  - Les **méthodes de construction par règles de priorité** sont basées sur des algorithmes de liste construisant progressivement un ordonnancement en affectant les opérations aux ressources et en résolvant les conflits d'utilisation des ressources par des règles de priorité. Les ordonnancements générés sont en général soit actifs, soit sans-délai. Ces méthodes sont très rapides et très utilisées en pratique. L'écart par rapport à la valeur optimale du critère peut être élevé. Cette approche est utilisée dans cette étude (voir paragraphe V.2).

  - Les **méthodes de voisinage** sont basées sur la génération itérative d'un ensemble d'ordonnancements voisins à partir d'un ordonnancement initial, dans le but d'améliorer le critère. On peut citer les *méthodes de descente* pour lesquelles un ordonnancement voisin doit obligatoirement améliorer le critère. Elles permettent d'obtenir un optimum local. D'autres méthodes autorisent la dégradation provisoire du critère en vue d'obtenir un optimum global. C'est le cas du *recuit simulé* [Van Laarhoven *et al.*, 1992] qui choisit un ordonnancement voisin au hasard dans l'espace défini par le voisinage. L'algorithme accepte de dégrader le critère avec une probabilité qui dépend de l'ampleur de la dégradation et d'autres paramètres, qu'on fait décroître au fur et à mesure de l'avancement de la procédure. C'est aussi le cas de la *méthode tabou* [Hertz et Widmer, 1995] qui explore l'ensemble des ordonnancements définis par le voisinage et qui choisit le meilleur (celui qui maximise la diminution du critère ou qui minimise sa dégradation). N'étant pas basée sur le hasard, cette méthode risque de revenir à un ordonnancement déjà visité (phénomène de cycle). Aussi, une liste tabou des ordonnancements voisins (ou de leur caractéristique forte) est maintenue. La qualité des résultats obtenus par ces méthodes, qui sont très utilisées, dépend fortement de la définition de la structure de voisinage et du réglage des divers paramètres qui doit être spécifique au problème traité. Des méthodes de voisinage de type tabou sont proposées dans cette étude (paragraphes IV.1, IV.2 et V.3). Les *algorithmes génétiques* (voir [Portmann, 1996]) sont basés sur une notion étendue de voisinage. Plusieurs solutions sont maintenues simultanément et évoluent par des techniques de recombinaison et de mutation, notions inspirées par la biologie. Une des difficultés rencontrées est le codage des solutions.

  - Les **méthodes de décomposition** [Portmann, 1988] visent à construire à partir du problème d'ordonnancement une suite de sous-problèmes plus faciles à résoudre. On distingue les méthodes de décomposition spatiale pour lesquelles les machines sont regroupées en ilôts de fabrication des méthodes de décomposition temporelle pour lesquelles on regroupe les produits en ordonnant les sous-problèmes le long de l'axe des temps. Pour ce dernier cas, on peut consulter [Levy, 1996].

  - Les **méthodes liées à l'intelligence artificielle** sont basées sur la représentation des connaissances au moyen d'un ensemble de règles de production. Un moteur d'inférence contenant la stratégie de contrôle (comparable à un algorithme de résolution) est nécessaire (voir [Lecomte, 93] pour un état de l'art de ce domaine).

En ordonnancement prévisionnel, une deuxième approche vise non pas à minimiser un critère mais à caractériser les ordonnancements admissibles dans le but d'offrir au(x) décideur(s) des degrés de liberté nécessaires pour assurer la robustesse des solutions proposées vis-à-vis de contraintes non modélisées et des perturbations. Ces approches sont basées sur l'*Analyse sous contraintes* dont les principes sont définis dans [Erschler, 1976]. C'est l'approche retenue ici qui est décrite plus précisément dans le paragraphe I.2.

Ordonnancement réactif Un système d'ordonnancement réactif est un système qui inclut une méthode pour réagir en temps réel face aux aléas. De la même façon qu'on distingue les contraintes internes ou externes, on peut distinguer les aléas internes survenant à l'intérieur de l'atelier (pannes, absences) des aléas externes provenant de son environnement (retard d'approvisionnement, arrivée imprévue d'un ordre de fabrication) (voir par exemple [Lamothe, 1996]).

On peut distinguer deux approches d'ordonnancement réactif :

- les approches purement réactives considèrent le cas où aucun ordre de fabrication n'est connu à l'avance. Les décisions d'affectation et de séquencement sont prises dès qu'un nouvel ordre de fabrication survient (instant de prise de décision). Dans ce cas, la mise en œuvre peut être effectuée localement à chaque ressource par l'utilisation de règles de priorité, sans remettre en cause les décisions précédentes, ou bien par une procédure d'optimisation globale, intégrant progressivement les ordres de fabrication, relancée à chaque instant de prise de décision. Les performances sont alors limitées à cause de l'impossibilité d'anticipation et ce choix délibéré n'est pas toujours justifié.

- les approches prédictives/réactives proposent une intégration de l'ordonnancement prévisionnel et de l'ordonnancement réactif. Le système SONIA basé sur l'intelligence artificielle et intégrant la notion de contraintes absolues et relatives [Collinot *et al.*, 1988] est un exemple de ce type d'approche. La méthode ORABAID (Ordonnancement d'Atelier Basé sur une AIde à la Décision) [Thomas, 1980] dans laquelle s'inscrit notre approche, issue de la problématique de l'analyse sous contraintes, est décrite dans le paragraphe I.2.

I.2 Approche retenue pour l'ordonnancement : la méthode ORABAID

La méthode ORABAID se compose d'une partie prévisionnelle (modules d'ordonnancement statique SINIT et ANALYSE) et d'une partie réactive (module d'ordonnancement en temps réel ARBITRE).

I.2.1 Partie prévisionnelle : modules SINIT et ANALYSE

L'idée de base de l'approche est de proposer au niveau prévisionnel non pas un ordonnancement mais un ensemble d'ordonnancements admissibles.

I.2.1.a) Approche par caractérisation d'un ensemble d'ordonnancements admissibles

La caractérisation de l'ensemble des ordonnancements admissibles, bien que possible en théorie [Erschler, 1976], pose un problème de complexité. En effet, considérons un problème d'atelier à cheminements multiples. On considère les dates de fin au plus tard des ordres de fabrication comme des contraintes externes à respecter. L'obtention d'un unique ordonnancement admissible dans ce contexte revient à résoudre le problème $Jm|\tilde{d}_i|-$, qui est NP-difficile. En pratique, on peut considérer deux cas. Si le problème est très contraint ($\tilde{d}_i$ *suffisamment petits*), le problème est réellement difficile. Le problème de l'obtention de l'ensemble des ordonnancements admissibles est au moins aussi complexe que celui de l'obtention d'un ordonnancement. Si le problème est peu contraint (d_i *suffisamment grands*), il est en réalité assez facile d'obtenir un ordonnancement réalisable admissible mais l'ensemble de tous les ordonnancements admissibles peut être très grand. Dans ce cas l'application des règles d'analyse sous-contraintes peut être lourde et sans grand résultat.

C'est pourquoi cette approche s'est scindée en deux voies, dans le but de traiter ces deux cas :

- la première voie traite les problèmes relativement contraints en cherchant à déterminer certaines caractéristiques des solutions admissibles sans aller en général jusqu'à la résolution complète du problème. On parle de recherche de conditions nécessaires d'admissibilité puisque l'ensemble des ordonnancements ainsi caractérisé peut contenir des ordonnancements non admissibles. Des règles d'analyse sous contraintes sont utilisées pour enrichir les connaissances sur le problème initial, et réduire l'ensemble des valeurs possibles des variables de décision sans écarter une solution admissible. Les mécanismes de déduction par propagation de contraintes sont décrits dans [Esquirol et Lopez, 1997]. Ces mécanismes peuvent être intégrés dans une procédure d'optimisation par séparation et évaluation pour réduire l'espace des solutions explorées [Erschler *et al.*, 1982], [Caseau et Laburthe, 1995], [Baptiste et Le Pape, 1997], ou dans une procédure heuristique en alternative aux règles de priorité [Ulusoy et Ozdamar, 1994], [Levy, 1996]. Les déductions peuvent être utilisées directement comme support d'une aide à la décision pour l'ordonnancement [Haudot, 1996], ou d'une aide à la négociation pour la gestion de production distribuée [Camalot et Esquirol, 1997]. L'appréhension des systèmes de production comme un réseau de centres de décision conduit à expliciter l'autonomie d'un centre par l'analyse sous contraintes et permet de repérer les régulations qui s'opèrent au cours du travail. Ces travaux, menés conjointement avec des sociologues, ont donné naissance à une problématique pluridisciplinaire, validée par des tests sur site industriel (voir par exemple [De Terssac *et al*, 1996]).

- la seconde voie traite les problèmes non fortement contraints en cherchant à générer un sous-ensemble des ordonnancements admissibles. On parle de recherche de conditions suffisantes d'admissibilité puisque l'ensemble des ordonnancements caractérisé est inclus dans l'ensemble des ordonnancements admissibles. Le problème est la représentation du sous-ensemble d'ordonnancements qui doit être à la fois compréhensible par l'utilisateur, interprétable en termes temporels (pour obtenir les dates de début des opérations), facilement modifiable en temps réel pour coller à l'état de l'atelier lors de l'exploitation réactive.

La méthode ORABAID [Demmou, 1977], [Thomas, 1980], [Le Gall, 1989], [Billaut, 1993] utilise une représentation d'un sous-ensemble d'ordonnancements par une séquence de groupes d'opérations permutables. La sélection d'une séquence d'opérations au sein de chaque groupe correspond à un ordonnancement unique. Si chaque ordonnancement possible est admissible, on dit que la séquence de groupes est admissible. L'utilisation d'un graphe potentiels-tâches étendu associé à la séquence de groupes permet de vérifier l'admissibilité ou de juger de l'écart de cette séquence par rapport à l'admissibilité sans énumérer l'ensemble des ordonnancements possibles, par calcul du plus long chemin, ou chemin critique. La simplicité de cette caractérisation est ainsi son principal avantage. Parmi les approches directement issues de ces principes on peut citer [Pellet, 85] et [Cho, 1989].

Une autre approche vise à caractériser des ensembles plus variés d'ordonnancement réalisables en utilisant les PQR-arbres à la place des graphes potentiels-tâches. Les PQR-arbres permettent d'exprimer des possibilités plus riches que la simple permutation d'opérations au sein d'un groupe [Zouhri, 1993]. Le problème pour ce type de méthodes est la difficulté de traiter pour l'instant des problèmes multi-opérations de taille réaliste.

I.2.1.b) Organisation de l'ordonnancement prévisionnel

Une heuristique est proposée pour la construction d'une séquence de groupe admissible. Elle est basée sur deux modules SINIT et ANALYSE. Le module SINIT cherche à générer une séquence de groupes admissible en utilisant un algorithme de construction progressive. Basé sur une règle de priorité, cet algorithme cherche à effectuer un compromis entre la constitution de groupes, i.e. l'obtention d'un nombre important de solutions et la recherche de l'admissibilité, i.e. la tenue des délais. Si le problème est relativement contraint, l'heuristique ne parvient pas à obtenir une séquence admissible et seule une séquence réalisable est générée. La recherche de l'admissibilité constitue alors une deuxième étape et est effectuée par le module ANALYSE. Cette recherche est basée sur l'amélioration itérative de la séquence issue de SINIT par l'analyse du chemin critique du graphe potentiels-tâches représentant la séquence. Deux principes sont appliqués de façon itérative :

- l'exclusion des ordonnancements non admissibles par *scission* d'un groupe,
- la modification de l'affectation de certaines opérations par *transfert* d'une opération.

Ceci revient à utiliser une méthode de voisinage pour se rapprocher de l'admissibilité. Tout au long de l'analyse, la longueur du chemin critique représente le plus grand retard algébrique des ordres de fabrication et constitue ainsi un indicateur de l'écart par rapport à l'admissibilité. A la fin de l'analyse, une phase de regroupements cherche à augmenter le nombre d'opérations de chaque groupe sans remettre en cause l'admissibilité.

On remarque dans la figure I.3 que l'application successive des modules SINIT et ANALYSE constitue un processus interactif. Le décideur peut intervenir au cours de la génération de la séquence initiale pour décider de l'ordre relatif des opérations correspondant à des ordres de fabrication en retard. A l'issue de la génération et en cas de non-admissibilité, le décideur peut sélectionner les ordres de fabrication pour lesquels la tenue des délais est impérative afin d'orienter l'étape d'analyse. Enfin, c'est à lui d'effectuer le compromis entre la durée de l'étape d'analyse et la qualité (l'admissibilité) de l'ensemble d'ordonnancements obtenu. A l'issue de ce processus qui est un exemple de boucle interactive "analyse/ construction/ évaluation" (voir figure I.5), la séquence de groupes est utilisée pour une exploitation en vue de la conduite en temps réel de l'atelier, si les délais proposés sont satisfaisants. Sinon, le plan de production est remis en cause.

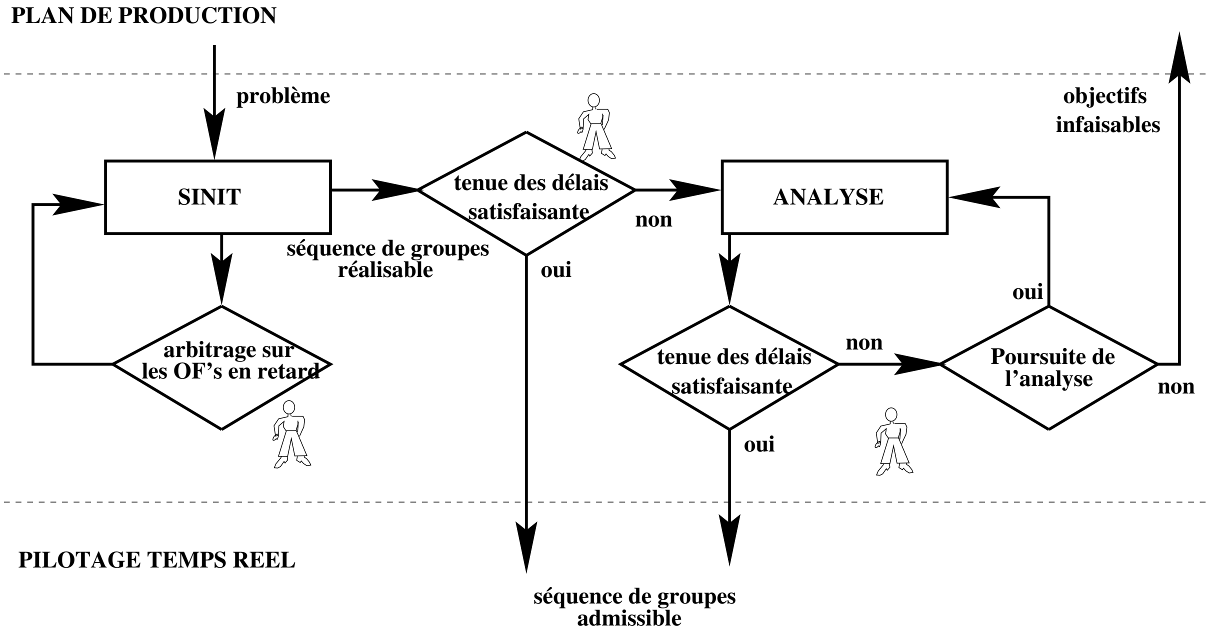


FIG. I.3 – *Ordonnancement prévisionnel*

I.2.2 Partie réactive : le module ARBITRE

Sur la base d'un système de suivi de l'atelier, le module ARBITRE est chargé de l'exploitation de la séquence de groupes admissible issue de l'ordonnancement prévisionnel pour la conduite en temps réel. Il s'agit d'un système interactif d'aide à la décision pour la conduite dont la description fonctionnelle est représentée dans la figure I.4.

Il s'agit d'assurer une aide pour tout événement appelant une décision en maintenant une adéquation entre la séquence de groupes utilisée et l'état réel de l'atelier. Pour cela, il est nécessaire de collecter les informations significatives des changements d'état dans l'atelier. Ces informations sont de deux types :

– les *décisions* d'utilisation des ressources par les opérations et d'engagement de ces opérations par le (ou les) décideur(s).

- les *événements* qui sont de deux types :
  - les *événements attendus* correspondent à une évolution prévisible de la séquence de groupes.
  - les *événements inattendus* correspondent à des aléas. On distingue les aléas internes et les aléas externes déjà définis.

Le décideur peut de plus effectuer des *requêtes* auprès du système pour obtenir des informations significatives ou simuler les décisions qu'il envisage.

En réponse à un événement, ARBITRE est amené à déterminer un ensemble de décisions possibles et/ou souhaitables ce qui constitue un ensemble de *propositions*. En réponse à une décision ou une requête, ARBITRE détermine l'impact sur la tenue des délais, ce qui constitue un ensemble d'*informations* pertinentes. Ces propositions et ces informations sont des éléments destinés à aider l'utilisateur dans l'élaboration de sa décision.

La procédure de génération d'une nouvelle séquence de groupes n'est ainsi réutilisée que quand le système ne parvient plus à absorber les perturbations et qu'une dérive trop importante de la tenue des délais est observée.

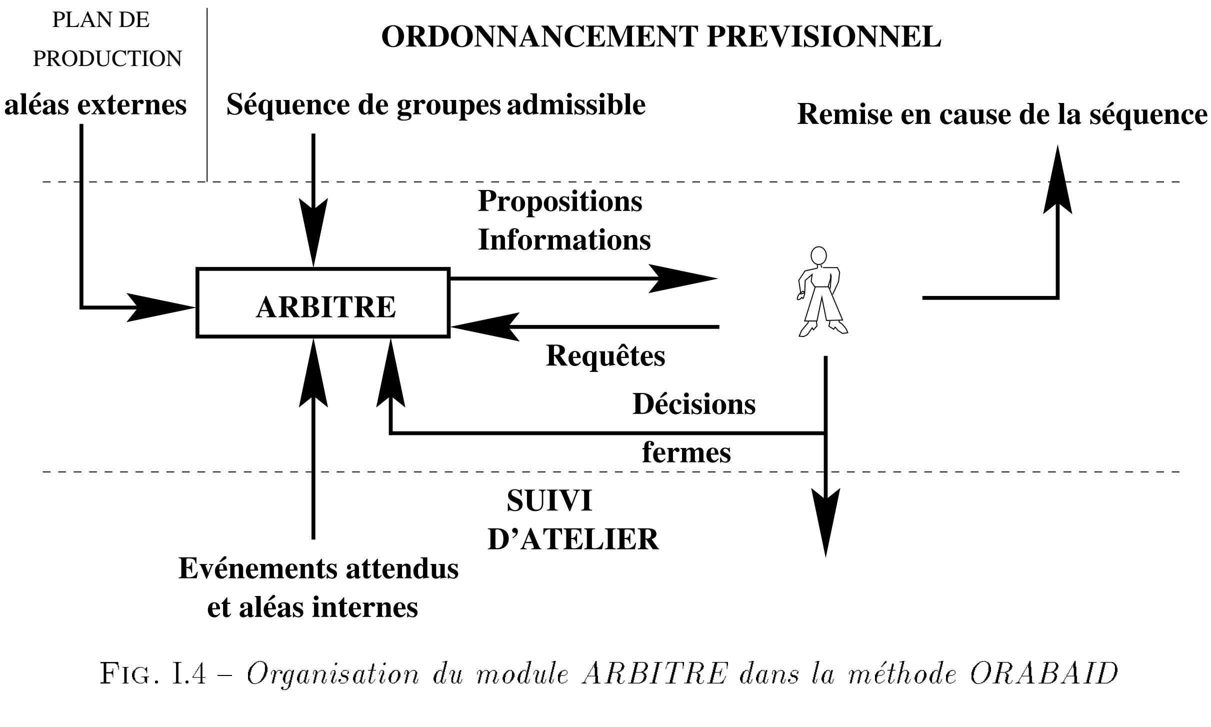


FIG. I.4 – *Organisation du module ARBITRE dans la méthode ORABAID*

I.2.3 Un système d'ordonnancement de la production intelligent

[Blazewicz *et al.*, 1996] définissent les caractéristiques d'un "système d'ordonnancement de la production intelligent" en soulignant qu'une approche pratique de l'ordonnancement doit tenir compte du caractère non prévisible et dynamique de l'atelier réel. Ces caractéristiques sont données dans le schéma I.5. La partie prévisionnelle (haut de la figure) analyse le problème à résoudre et construit un ordonnancement qui est évalué. Si l'ordonnancement proposé par le système ne convient pas au décideur, une nouvelle phase d'analyse est nécessaire. Ce processus interactif se poursuit itérativement jusqu'à ce qu'un ordonnancement réalisable acceptable soit trouvé. L'ordonnancement proposé se traduit en une stratégie pour le contrôle en temps réel de l'atelier. On conserve la même stratégie tant que l'état de l'atelier colle au problème initial. En cas de perturbation, le système propose une décision adaptée pour modifier la stratégie. En cas de perturbation trop importante, un nouvel ordonnancement doit être généré. On retrouve bien les deux parties : l'ordonnancement prévisionnel (*off-line*) interactif et l'ordonnancement en temps réel (*on-line*) caractéristiques de notre méthode.

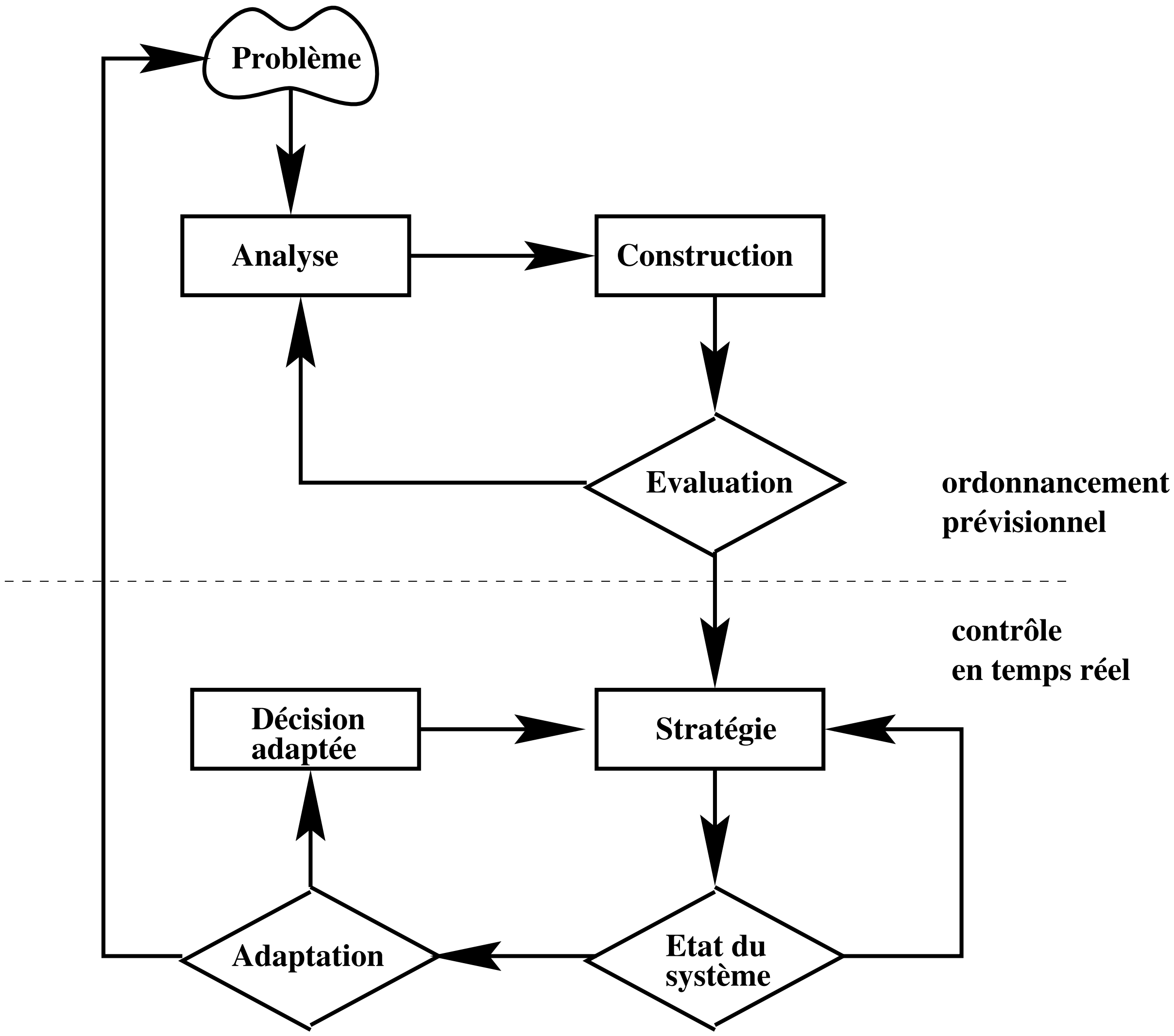


FIG. I.5 – *Décomposition chronologique et fonctionnelle d'un système d'ordonnancement de production intelligent d'après [Blazewicz et al., 1996]*

I.3 Evolution de l'approche, contexte de l'étude

Le but de ce paragraphe est de décrire brièvement les travaux précédents liés à la méthode ORABAID afin de situer les besoins qui sont à l'origine de cette étude.

Face à la diversité des problèmes d'ordonnancement, aux exigences des utilisateurs et à la concurrence des autres logiciels d'ordonnancement, de nouvelles caractéristiques des ateliers ont été intégrées progressivement et des améliorations successives de l'approche de résolution ont été apportées :

- [Demmou, 1977] définit le concept de groupes d'opérations permutables dans le cas d'un atelier à cheminements multiples avec dates de début au plus tôt et dates de fin au plus tard des ordres de fabrication. Une procédure de génération d'une séquence de groupes admissible est définie avec les principes de scission et de regroupement pour l'amélioration de cette séquence.

- [Thomas, 1980] définit les bases de l'exploitation en temps réel (module ARBITRE) pour le même problème d'ordonnancement et définit notamment la notion de marge libre séquentielle (voir chapitre 5) qui permet de choisir la meilleure opération à exécuter en temps réel au sein d'un groupe et de mesurer l'impact des aléas. La première version du logiciel ORABAID est développée.

- [Le Gall, 1989] intègre plusieurs caractéristiques issues de la réalité des ateliers, en relation avec le développement engagé depuis 1985 du progiciel associé ORDO. Il s'agit de la notion de gammes non-linéaires par la constitution de réseaux d'ordres de fabrication, des temps de transports, des calendriers des ressources, de la prise en compte des pools de ressources disjonctives et de la prise en compte des ressources cumulatives. Une procédure de transfert par changements d'affectation est définie pour l'amélioration de la séquence initiale. Les principes de l'aide à la décision sont adaptés à ces nouvelles contraintes et une procédure d'insertion en temps réel d'un nouvel ordre de fabrication dans une séquence de groupes est définie.

- [Billaut, 1993] intègre les contraintes multi-ressources ainsi que le chevauchement entre opérations d'un même ordre de fabrication et introduit la recherche interactive d'une séquence de groupes. La prise en compte des gammes non-linéaires et des calendriers est améliorée. La procédure de génération de la séquence est améliorée par l'intermédiaire d'une étude portant sur les règles de priorité utilisées. Pour l'exploitation de la séquence en temps réel, une formalisation des états de l'atelier sous la forme de réseaux de Petri est effectuée pour représenter l'ensemble des événements et des états significatifs pour la prise de décision. Enfin, une extension de la méthode de génération d'une séquence de groupes initiale aux temps de préparation est envisagée dans un contexte disjonctif mono-ressource.

L'étude présentée dans ce mémoire porte sur la prise en compte des temps de préparation dépendant de la séquence dans un contexte cumulatif, multi-ressources, aussi bien pour la génération de la séquence de groupes que pour son exploitation en vue de la conduite en temps réel. Elle a également conduit à proposer l'amélioration des procédures existantes suivantes :

- la procédure d'amélioration de la séquence (ANALYSE) qui est basée sur un algorithme exponentiel dans le contexte cumulatif multi-ressoures.

- la procédure d'insertion en temps réel d'un ordre de fabrication qui s'interdit jusqu'à présent d'augmenter le retard des ordres de fabrication déjà présents. Cette caractéristique provoque bien souvent une forte pénalisation de l'ordre de fabrication inséré, ce qui peut être en contradiction avec son urgence. De plus la procédure d'insertion, comme la procédure d'amélioration, devient inutilisable pour des problèmes multi-ressources cumulatifs de taille réaliste.

Chapitre II

Approche pour la prise en compte des temps de préparation

Résumé

Un état de l'art présente la prise en compte de temps de préparation dans les problèmes d'ordonnancement à travers différents modèles et différentes approches de résolution rencontrés dans la littérature. Le modèle retenu dans cette étude est présenté et justifié. Le concept d'opérations de préparation est défini. La notion de groupes d'opérations permutables est étendue à ce contexte et la caractérisation d'une séquence de groupes au moyen d'un graphe potentiels-tâches est présentée.

II.1 Les temps de préparation dépendant de la séquence dans les problèmes d'ordonnancement

Ce paragraphe présente un état de l'art ayant pour thème la préparation des ressources dans les différents problèmes d'ordonnancement. Selon la difficulté générée par l'introduction de la préparation, les méthodes classiques présentées dans le chapitre précédent sont adaptées ou de nouvelles méthodes spécifiques sont proposées. Cette difficulté dépend essentiellement :

- de la finesse du modèle retenu pour la préparation,
- de l'objectif à atteindre (critère à optimiser ou contraintes à satisfaire),
- du type d'atelier considéré.

Les deux premiers aspects sont décrits dans le paragraphe II.1.1. Les paragraphes suivants décrivent les ateliers dans un ordre de complexité croissante : problème à une machine dans le paragraphe II.1.2, problème de machines parallèles dans le paragraphe II.1.3, problème d'atelier à cheminement unique dans le paragraphe II.1.4, problème d'atelier à cheminements multiples dans le paragraphe II.1.5 et un ensemble de problèmes généralisés dans le paragraphe II.1.6.

II.1.1 Modèles pour la préparation des ressources

Ce paragraphe est un résumé des différents modèles et critères rencontrés dans les principales publications concernées. La préparation des ressources peut être modélisée par deux composantes bien distinctes :

- les temps de préparation (*setup ou changeover time*).
- les coûts de préparation (*setup ou changeover cost*).

II.1.1.a) Temps de préparation

On peut trouver dans [Crouhy, 1983] la description détaillée des temps liés à l'exécution d'une opération sur une ressource disjonctive. Parmi ces temps, le temps de préparation est un temps immobilisant la ressource préalablement ou suite à l'exécution d'une opération.

On distingue :

- Les temps de montage indépendants de la séquence, notés S_{nsd} (*non sequence dependent Setup times*). Le temps de montage qui immobilise la ressource avant une opération i et qui ne dépend que de i, est noté S_i.
- Les temps de démontage indépendants de la séquence, notés R_{nsd} (*non sequence dependent Removal times*). Le temps de démontage qui immobilise la ressource après une opération i et qui ne dépend que de i, est noté R_i.
- Les temps de préparation dépendant de la séquence, notés S_{sd} (*sequence dependent Setup times*). Le temps de préparation est nécessaire entre deux opérations i et j et dépend à la fois de l'opération précédente i et de l'opération suivante j. Il est noté S_{ij}. Pour un problème donné, l'ensemble $\{S_{ij}\}$ définit une matrice des temps de préparation.

Dans les problèmes comportant plus d'une ressource, les temps définis précédemment peuvent aussi dépendre de la ressource préparée et sont notés respectivement S_{ik} (temps de montage associé à l'opération i sur la ressource k), R_{ik} (temps de démontage associé à l'opération i sur la ressource k), et S_{ijk} (temps de préparation de la ressource k entre l'opération i et l'opération j). Dans ce dernier cas, on définit ainsi une matrice des temps de préparation par ressource.

Certaines opérations sont parfois regroupées par caractéristiques communes de préparation. Le temps de préparation par lots ou par familles (*batch* ou *family setup time*) est un cas fréquemment rencontré. Chaque opération est supposée appartenir à une famille (ou un lot). Aucun temps de préparation n'est nécessaire entre deux opérations de la même famille. Un temps de préparation est nécessaire entre deux opérations de familles différentes. Ce temps est soit dépendant uniquement de la famille de l'opération qui le suit (S_{nsd},*batch*), soit dépendant uniquement de la famille de l'opération qui le précède (R_{nsd},*batch*), soit dépendant de la séquence des familles (S_{sd},*batch*).

Une extension du modèle précédent introduit la notion de temps de préparation majeurs et mineurs (*Major* et *minor setup times*). Le temps de préparation majeur est le temps de préparation nécessaire entre deux opérations de familles différentes alors qu'un temps de préparation mineur est nécessaire entre deux opérations de même famille. Ce type de problème sera noté *maj, min*.

Dans les problèmes multi-opérations ou dans les problèmes mono-opération comportant des dates de début au plus tôt, il est nécessaire d'indiquer si la préparation peut être ou non anticipée i.e. commencée avant l'arrivée de l'opération (arrivée des pièces sur la machine, par exemple). Si la préparation peut être anticipée, on parle alors de temps de préparation *séparé* du temps d'exécution. Dans les modèles multi-opérations, la préparation d'une opération donnée peut dans ce cas commencer avant la fin de l'opération précédente dans la gamme. Dans le cas contraire, le temps de préparation est inclus dans la durée de l'opération (temps de réglage des pièces sur la machine par exemple). S'il ne dépend pas de la séquence et s'il n'y a pas de famille, il est inutile de faire apparaître un tel temps dans le modèle. Sinon c'est la durée de l'opération elle-même qui dépend de la séquence et on a le cas d'un temps de préparation immobilisant à la fois la machine et l'opération. Ce dernier cas étant plus rarement rencontré il sera explicitement indiqué et noté S_{inclus}.

La figure II.1 résume ces différents modèles et donne pour chacun d'eux des exemples de représentation des temps de préparation dans des diagrammes de Gantt.

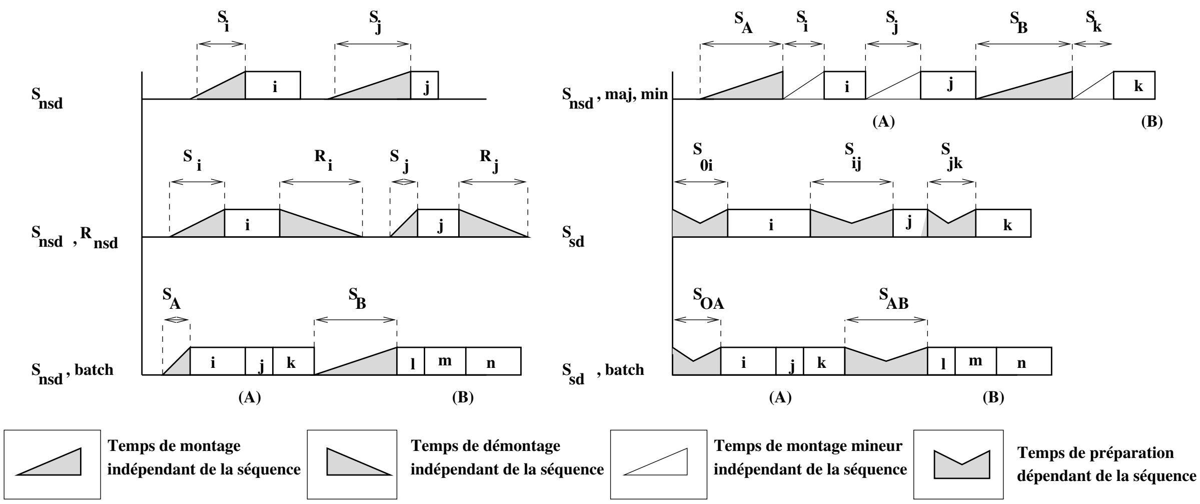


FIG. II.1 – *Différents modèles de temps de préparation*

Le critère à minimiser lié aux temps de préparation le plus fréquemment utilisé est le temps total de préparation, noté $\sum S_{ij}$ ou $\sum S_i$, selon le cas.

Ce critère peut être combiné avec d'autres critères classiques sous la forme d'une somme pondérée. La préparation peut aussi être absente du critère, l'objectif étant la minimisation d'un critère classique (C_{max}, L_{max}, ...) avec prise en compte des contraintes de préparation.

II.1.1.b) Coûts de préparation

Le coût de préparation, noté C, est un coût associé à la préparation d'une ressource préalablement ou suite à l'exécution d'une opération, mais qui ne se traduit pas par un temps d'occupation de la ressource. Il peut être ou non utilisé conjointement au temps de préparation dont il peut aussi dépendre. On distingue les coûts de préparation dépendant de la séquence (C_{sd}, C_{ij}) des coûts de préparation indépendants de la séquence (C_{nsd}, C_i). On peut remarquer que le cas C_{nsd} n'a de sens que dans des problèmes d'ordonnancement par familles (lots). Si les coûts de préparation sont pris en compte, ils sont obligatoirement inclus dans le critère à minimiser. On considère souvent le coût total de préparation noté TSC (*Total Setup Cost*) égal à $\sum C_{ij}$ ou $\sum C_i$ selon le cas. Dans le cas de coûts unitaires, la minimisation du coût total de préparation permet par exemple de minimiser le nombre de préparations effectuées. On trouve aussi des critères combinant les coûts de préparation avec les critères classiques. Ce genre de critères est noté TC (*Total Cost*).

II.1.2 Problème d'ordonnancement à une machine

II.1.2.a) Problème de minimisation de la durée maximale

Le problème de minimisation de la durée maximale (ou *makespan*) dans le cas où les opérations n'ont pas de contraintes temporelles et les temps de préparation sont dépendants de la séquence est noté $1|S_{sd}|C_{max}$. Dans le problème sans temps de préparation ($1||C_{max}$), ou avec des temps de préparation indépendants de la séquence ($1|S_{nsd}|C_{max}$), toute séquence est optimale. Dans le problème avec temps de préparation indépendants de la séquence par familles $1|S_{nsd}, batch|C_{max}$, toute séquence dans laquelle les opérations de la même famille sont regroupées est optimale. Par contre, le problème $1|S_{sd}|C_{max}$ est $NP-$difficile car il est équivalent au problème du voyageur de commerce (PVC) (voir [Conway *et al.*, 1967], [Baker, 1974]). Dans sa formulation classique, le PVC est symétrique. Or, dans les problèmes de préparation, ce n'est parfois pas le cas ($S_{ij} \neq S_{ji}$). Dans le cas classique d'enchaînement de couleurs donné dans [Conway *et al.*, 1967], le passage d'une couleur claire vers une couleur foncée est plus rapide que l'inverse. Le problème est alors équivalent au problème du voyageur de commerce asymétrique (PVCA). Toute méthode traitant ce problème est applicable au problème considéré ici (voir [Lawler *et al.*, 1985] pour un état de l'art de ces méthodes).

Parmi les méthodes optimales utilisées, on trouve des procédures par séparation et évaluation (PSE) dont la performance dépend des bornes inférieures utilisées. Une borne inférieure du PVC asymétrique peut être obtenue par la résolution du problème d'affectation associé [Baker, 1974].

Le PVC peut être aussi résolu par la programmation dynamique, le temps total de préparation ayant une forme additive [Baker, 1974]. Dans [Aguilera, 1993] il est rapporté que ce type de méthodes n'est capable de résoudre que des problèmes d'une dizaine de tâches.

Parmi les heuristiques de listes utilisant des règles de priorité, on peut citer l'heuristique très connue "Next best", encore appelée "Plus proche ville" de [Gavett, 1965] qui consiste à choisir toujours l'opération qui minimise le temps de préparation.

Une heuristique originale décrite dans [White et Wilson, 1977] est basée sur une équation de prédiction des temps de préparation. Une description fine des diverses opérations de préparation pouvant intervenir au cours d'une activité de préparation est effectuée. Chaque activité de préparation est alors spécifiée comme une combinaison de certaines de ces opérations.

Trois méthodes de voisinage (un recuit simulé, un algorithme de seuil, une méthode de descente) sont comparées dans [Hansmann et Höck, 1995] pour un problème à 120 opérations issu d'une industrie de fabrication de cigarettes. Il est alors mis en évidence la supériorité des méthodes acceptant localement des dégradations du critère (recuit simulé et seuil) sur la méthode de descente qui n'accepte que des mouvements améliorant le critère.

II.1.2.b) Problème de minimisation des coûts de préparation avec contraintes temporelles

Lorsque le changement d'opération sur la ressource se traduit par un coût C_{ij} et non par un temps, le problème de la minimisation du coût total de préparation est équivalent au problème de la minimisation du temps total de préparation. Cette équivalence n'est plus valide lorsque le coût de préparation est une composante non unique du coût à minimiser. Les autres composantes sont par exemple des coûts de stockage des opérations terminées ou des pénalités de retard. Les méthodes proposées sont pour la plupart des méthodes optimales (voir [Aguilera, 1993] pour un état de l'art de ces méthodes). [Driscoll et Emmons, 1977] proposent un algorithme de programmation dynamique pour résoudre optimalement le problème de la minimisation du coût total de préparation lorsque les dates de fin au plus tard sont prises comme des contraintes (*deadline* $\tilde{d}_i$). Ce problème est équivalent au PVC avec dates d'achèvement. Dans [Barnes et Vanston, 1981], une procédure mixte de PSEP et de programmation dynamique est proposée pour minimiser un coût total dépendant à la fois du coût de préparation et du retard des tâches.

II.1.2.c) Prise en compte conjointe des temps de préparation et des contraintes temporelles

Les méthodes prenant en compte conjointement des contraintes temporelles et les temps de préparation sont plus rares. Elles sont étudiées pour la plupart dans le cadre de temps de préparation par lot. Dans [Bruno et Downey, 1978], le problème de minimisation du temps total de préparation avec des dates de fin au plus tard imposées ($1|S_{sd}, \tilde{d}_i|\sum s_{ij}$) est étudié. Il est prouvé que le problème de la détermination d'une solution admissible est $NP-$ difficile, même lorsque les temps de préparation sont indépendants de la séquence. Sans les temps de préparation et sans dates de début au plus tôt, le problème de minimisation du retard maximum ($1||L_{max}$) peut être résolu de façon optimale par la règle de priorité EDD (Earliest Due-date) [Jackson, 55] qui consiste à ordonnancer les opérations dans l'ordre croissant de leur date de fin au plus tard. [Monma et Potts, 1989] introduisent les temps de préparation dépendant de la séquence par lot dans ce problème et proposent un algorithme de programmation dynamique polynomial en fonction du nombre d'opérations mais exponentiel en fonction du nombre de lots pour le résoudre optimalement. Les auteurs supposent que l'inégalité triangulaire est vérifiée sur les temps de préparation, ce qui s'exprime par $S_{ij} \leq S_{ik} + S_{kj}$, $\forall i, j, l$. Des propriétés intéressantes découlent de cette supposition raisonnable. [Rubin et Ragatz, 1995] utilisent un algorithme génétique pour la minimisation du retard vrai total ($1|S_{sd}|\Sigma T_i$). Des tests sur des problèmes générés aléatoirement montrent que la procédure obtient de bons résultats en comparaison avec une procédure par séparation et évaluation, surtout lorsque les dates de livraison sont ne sont pas serrées. [Chen, 1997] propose une procédure d'optimisation par programmation dynamique pour la minimisation de la somme pondérée des retards et des avances.

Le problème de minimisation du retard maximum sans temps de préparation et avec dates de début au plus tôt ($1|r_i|L_{max}$) est $NP-$difficile mais peut être résolu optimalement par l'algorithme très efficace de Carlier [Carlier, 1982]. [Ovacik et Uzsoy, 1994a] prennent en compte des temps de préparation dépendant de la séquence dans ce problème ($1|r_i, S_{sd}|L_{max}$) et proposent une procédure d'horizon glissant en considérant une arrivée dynamique des opérations. L'heuristique proposée résout optimalement un problème local en ne considérant connu à l'avance qu'un sous-ensemble des opérations, déterminé par l'horizon glissant de l'ordonnancement. Cette heuristique permet ainsi d'effectuer un compromis entre la performance et le temps de traitement. [Schutten, 1996] propose une PSEP pour le problème avec temps de préparation par familles et indépendant de la séquence ($1|r_i, S_{nsd}|, batch|L_{max}$). Le calcul d'une borne inférieure est basé sur la modélisation du temps de préparation comme une opération ayant une date de début au plus tôt, une date de fin au plus tard et des contraintes de précédence particulières. Une méthode pour déterminer des opérations de préparation obligatoires est présentée. Cette considération est l'idée forte de cette méthode qui résout de façon optimale des problèmes comportant jusqu'à 40 tâches.

II.1.2.d) Problème de détermination de la taille des lots et d'ordonnancement

Le problème d'ordonnancement avec temps de préparations dépendant de la séquence par familles est équivalent au problème de détermination de la taille des lots (*lot-sizing*) avec temps de préparation dépendant de la séquence (voir par exemple [Schutten, 1996], [Jordan et Drexl, 1994]). Dans le problème $DLSP$ (*Discrete Lot Sizing and Scheduling Problem*), décrit dans [Salomon *et al.*, 94], le temps est discrétisé en périodes. Dans une période donnée la machine est soit en train d'exécuter un lot, soit en train d'être préparée entre deux lots. Le nombre de périodes requises pour passer d'un lot i à un lot j dépend des deux lots. Un algorithme basé sur le PVC avec fenêtres temporelles (PVCFT) est défini pour minimiser le temps total de préparation et les coûts de stockage.

II.1.2.e) Résumé des travaux sur le problème à une machine

Les principaux résultats prenant en compte les temps de préparation dans le problème à une machine sont présentés dans la table II.1, ainsi que quelques travaux sur le problème du $DLSP$. H signifie que la méthode proposée est une heuristique. O signifie que la méthode proposée est optimale. "Prog. dyn." signifie "programmation dynamique". Sinon "dyn." signifie "arrivée dynamique des opérations" par opposition à l'hypothèse que toutes les opérations sont connues à l'avance notée "stat.".

Référence	Problème		Méthode employée
[Gavett, 1965]	$1\|S_{sd}\|C_{max}$	H	heur. de liste (Next best)
[Conway *et al.*, 1967]		O	PSE, prog. dynamique (PVC)
		H	heur. de liste (plus proche ville)
[Presby et Wolfson, 1967]		O	construction progressive
[Lockett et Muhlemann, 1972]		O	PSE (chang. d'outil)
		H	PVC
[Baker, 1974]		O	PSE, prog. dyn. (PVC)
[White et Wilson, 1977]		H	heur. de liste équation prédictive
[Farn et Muhlemann, 1979]		H	heur de liste (stat. et dyn.)
[Hansmann et Höck, 1995]		H	recuit simulé, alg. de seuil (PVC)
[Burstall, 1966]	$1\|C_{sd}\|TC$	H	décomposition
[Buzacott et Dutta, 1971]		O	prog. dyn.
[Taha, 1971]		O	prog. non lin. entier, 0-1
[Sielken, 1976]		O	prog. lin. entier, 0-1
[Barnes et Vanston, 1981]		O	PSEP+prog. dyn.
[Glassey, 1968]	$1\|C_{sd}, C_{ij} = 1, \tilde{d}_i\|\Sigma C_{ij}$	O	prog. dyn.
[Mitsumori, 1972]	$1\|C_{sd}, \tilde{d}_i\|\Sigma C_{ij}$	H	prog. dyn.
[Driscoll et Emmons, 1977]		O	prog. dyn.
[Picart et Queyranne, 1977]	$1\|C_{sd}\|\Sigma w_i T_i$	O	PSEP (PVCDT)
[Weeda, 1978]	$1\|S_{sd}, C_{sd}\|TC$	O	prog. dyn.
[Bruno et Downey, 1978]	$1\|S_{sd}, \tilde{d}_i\|\Sigma s_{ij}$	-	étude de complexité
[Coffman *et al.*, 1989]	$1\|S_{nsd}, prec, 2batch\|\Sigma C_i$	O	prog. dyn. (2 familles)
[Mason et Anderson, 1991]	$1\|S_{nsd}, batch\|\Sigma w_i C_i$	O	PSE
[Ham *et al.*, 1985]		H	techn. groupe
[Sahney, 1972]	$1\|S_{sd}, 2batch\|\Sigma C_i$	H	(2 familles)
[Gupta, 1984]		O	
[Potts, 1991]		O	
[Gupta, 1988]	$1\|S_{sd}, batch\|\Sigma C_i$	H	
[Ahn et Hyun, 1990]		H	
[Psaraftis, 1980]	$1\|S_{sd}, p_i = p_f, w_i = W_f, batch\|\Sigma w_i C_i$	O	prog. dyn.
[Dobson *et al.*, 1987]		O	
[Monma et Potts, 1989]	$1\|S_{sd}, batch\|L_{max}$	O	prog. dyn.
	$1\|S_{sd}, batch\|\Sigma w_i C_i$	O	prog. dyn.
	$1\|S_{sd}, batch\|\Sigma U_i$	O	prog. dyn.
[Ghosh, 1994]	$1\|S_{sd}, batch\|\Sigma w_i C_i$	-	Etude de complexité
[Rubin et Ragatz, 1995]	$1\|S_{sd}\|\Sigma Ti$	H	alg. génétique
[Chen, 1997]	$1\|S_{sd}, batch\|\Sigma(\alpha_i E_i + \beta_i T_i)$	O	prog. dyn.
[Santos et França, 1997]	$1\|S_{sd}\|\Sigma(\alpha_i E_i + \beta_i T_i + \gamma_{ij} S_{ij})$	H	proc. tabou
[Ovacik et Uzsoy, 1994a]	$1\|r_i, S_{sd}\|L_{max}$	H	(dyn.) proc. d'horizon glissant
[Schutten *et al.*, 1996]	$1\|r_i, S_{nsd}, batch\|L_{max}$	O	PSEP (tâches de préparation)
[Fleischmann, 1990]	DLSP C_{nsd}	O	PSEP/prog. dyn.
[Magnanti et Vachani, 1990]		O	méthode polyhédrale
[Cattrysse *et al.*, 1993]	DLSP s_{nsd}, C_{nsd}	H	génération de colonnes
[Jordan et Drexl, 1994]		O	prog. dyn.
[Fleishmann, 1994]	DLSP C_{sd}	H	heur. basée sur PVCFT
[Salomon *et al.*, 94]	DLSP S_{sd}, C_{sd}	O	prog. dyn. (PVCFT)

TAB. II.1 – *Temps ou coûts de préparation dans le problème à une machine*

II.1.3 Problème d'ordonnancement à machines parallèles

II.1.3.a) Problème de minimisation de la somme des temps ou des coûts de préparation sans contraintes temporelles

Le problème de la minimisation de la somme des temps ou des coûts de préparation (respectivement $Pm|S_{sd}|\sum S_{ij}$ ou $Pm|C_{sd}|\sum C_{ij}$) est équivalent au problème des tournées de véhicules (*Vehicle Routing Problem*), avec de légères modifications. Les procédures dédiées à ce problème (voir [Desrochers *et al.*, 1990]) peuvent être appliquées au problème $Pm|S_{sd}|\sum S_{ij}$. [Deane et White, 1975] proposent une PSEP dans le cas noté *wlr* (*workload restriction*) où la charge de chaque machine est contrainte. [Parker *et al.*, 1977] établissent l'équivalence avec le problème du véhicule. Ils utilisent une heuristique dédiée à la résolution de ce problème. Plus récemment, [Aguilera *et al.*,1996] proposent une heuristique en deux phases pour minimiser la somme des coûts de changement dans un problème industriel avec des machines uniformes ($Qm|C_{sd}|\sum C_{ij}$). La première phase affecte les tâches aux machines, la seconde phase optimise l'ordonnancement des machines. Toutefois, contrairement au problème à une machine, la minimisation des temps de préparation n'est pas équivalente à la minimisation de la durée totale de l'ordonnancement.

II.1.3.b) Problème de minimisation des coûts de préparation avec contraintes temporelles

On trouve quelques méthodes prenant en compte les coûts de préparation et des contraintes temporelles: [Geoffrion et Graves, 1976] proposent une procédure d'optimisation prenant en compte des pénalités de retard, une fenêtre temporelle étant associée à chaque opération. [Hu *et al.*, 1987] considèrent le problème de la minimisation de la somme des coûts de préparation sous contraintes de dates d'achèvement et donnent un algorithme optimal. [Aguilera, 1993] propose une procédure de réaffectation pour tenter de respecter les fenêtres temporelles associées aux opérations. Ces méthodes dissocient totalement les problèmes temporels et les problèmes de préparation puisque le temps de préparation est supposé nul. Elles sont bien appropriées à des problèmes où les temps de préparation n'ont pas d'influence sur la durée totale de l'ordonnancement ni sur le respect des contraintes temporelles.

II.1.3.c) Problème de minimisation de la durée totale

Le problème de minimisation de la durée totale sans temps de préparation noté $Pm||C_{max}$ est $NP-$difficile (voir par exemple [Blazewicz *et al.*, 1996]). La plupart des méthodes considérant ce problème avec temps de préparation dépendant de la séquence ($Pm|S_{sd}|C_{max}$) sont des heuristiques.

[Ovacik et Uzsoy, 1993] montrent que dans ce cas, les ordonnancements sans-délai ne sont plus dominants, ce qui pénalise fortement les heuristiques de liste qui génèrent de tels ordonnancements. Dans [Schutten, 1996], un algorithme de liste alternatif générant des ordonnancements dominants est proposé.

Une des premières heuristiques utilisées pour résoudre ce problème est l'heuristique du plus proche voisin dans [Frederickson *et al*, 1978], suivant le même principe que l'heuristique de la plus proche ville pour le problème à une machine.

[Guinet, 1993] montre que le problème $Pm|S_{sd}|C_{max}$ est encore équivalent à un problème de véhicules avec une fonction coût particulière et propose une heuristique en deux phases basée sur ce problème.

[França *et al.*, 1995] proposent une méthode efficace de recherche locale de type tabou, inspirée par le PVC. La génération d'une solution voisine est effectuée par une procédure qui vise à réinsérer une opération sur une machine en réoptimisant la préparation. Des problèmes à 50 opérations sont résolus presque optimalement avec toutefois un temps de calcul pouvant aller jusqu'à 9000 secondes sur SUN-SPARC2.

Le problème à machines non identiques est traité par [Guinet, 1991] à partir de problèmes issus de l'industrie textile. Le critère considéré est la somme des durées $\sum C_i$. L'heuristique proposée est encore inspirée par le problème des tournées de véhicules.

La minimisation de la durée totale est aussi étudiée pour le problème des temps de préparation mineurs et majeurs par familles, indépendant de la séquence. Des heuristiques "deux-phases" d'affectation et d'ordonnancement [Tang, 1990], [So, 1990] et une procédure optimale [Bitran et Gilbert, 1990] sont proposées.

Pour le problème de temps de préparation par lots avec préemption autorisée ($Pm|prmnt, S_{sd}, batch|C_{max}$), [Monma et Potts, 1993] proposent une heuristique de liste améliorée plus tard par [Chen, 1993].

II.1.3.d) Prise en compte conjointe des temps de préparation et des contraintes temporelles

Les problèmes à machines parallèles dans lesquels les temps de préparation dépendent de la séquence et les opérations sont soumises à des contraintes temporelles sont très peu étudiés. [Monma et Potts, 1989] montrent que le problème de minimisation du plus grand retard, dans le cas préemptif avec temps de préparation par lots indépendants de la séquence est $NP-$difficile, même pour le problème à deux machines identiques. [Echalier, 1991] utilise le recuit simulé pour minimiser la somme des retards dans un problème à machines indépendantes et temps de préparation dépendant de la séquence. Peu d'approches sont proposées à notre connaissance pour le problème de minimisation du plus grand retard avec dates de début au plus tôt des opérations. Dans [Kedad. 1997], un ensemble de méthodes stochastiques (méthodes de voisinage et algorithmes génétiques) et une heuristique en deux phases sont testés sur le problème à machines indépendantes. [Ovacik et Uzsoy, 1995] utilisent une procédure d'horizon glissant pour le problème général $Pm|r_i, S_{sd}|L_{max}$ en environnement dynamique. Des résultats performants sont obtenus en comparaison avec des règles de priorité qui, toutefois, ne prennent pas en compte explicitement les temps de préparation. Enfin, [Schutten, 1996] propose une PSEP pour le problème avec temps de préparation indépendants de la séquence par familles et dates de début au plus tôt ($Pm|r_i, S_{snd}, batch|L_{max}$). Il utilise deux bornes inférieures du problème sans temps de préparation et propose des règles de dominance. Les problèmes résolus comportent jusqu'à 25 opérations et 3 machines.

II.1.3.e) Résumé des travaux sur le problème à machines parallèles

Les principales publications dans ce domaine sont résumées dans la table II.2. Certaines références sont issues de l'état de l'art effectué dans [Botta, 1996].

Référence	Problème		Méthode employée
[Kirby et Scobey, 1970]	$Pm\|C_{sd}\|TC$	H	prog. lin. (machines couplées)
[Arthanari et Ramamurthy, 1971]		O	PSEP
[Sumichrast et Baker, 1987]		H	prog. lin.
[Bernardo et Lin, 1994]		H	proc. interactive
[Prabhakar, 1974]	$Pm\|C_{sd}, \mathrm{prmnt}\|TC$	O	prog. lin. prmt non autorisée sur la même machine
[Geoffrion et Graves, 1976]	$Pm\|r_i, C_{sd}, prmnt\|TC$	0	alg. aff. quad/prog. lin.
[Marsh et Montgomery, 1973]	$Pm\|S_{sd}\|\Sigma S_{ij}$	O	PSEP
[Deane et White, 1975]	$Pm\|C_{sd}, wlr\|\Sigma C_{ij}$	O	PSEP
[Parker *et al.*, 1977]	$Pm\|C_{sd}, wlr\|TSC$	H	prob. de véhicules
[Hu *et al.*, 1987]	$Pm\|C_{sd}, \tilde{d}_i\|\Sigma C_{ij}$	O	deux phases
[Frendewey et Sumichrast, 1988]		O	prog. lin.
[Aguilera *et al.*,1996]	$Qm\|C_{sd}\|\Sigma C_{ij}$	H	deux phases
[Frederickson *et al*, 1978]	$Pm\|S_{sd}\|C_{max}$	H	plus proche voisin (PVC)
[Guinet, 1993]		H	méthode Hongroise(PVC)
[França *et al.*, 1995]		H	méthode tabou (PVC)
		O	recherche dichotomique (prob. de véhicules)
[Guinet, 1991]	$Qm\|S_{sd}\|\Sigma C_i$	H	prog. linéaire (PVC)
[Randhawa et Smith, 1995]	$Qm\|S_{sd}\|\bar{F}$	H	heur de liste
[Elmaghraby et Guinet, 1992]	$Rm\|S_{sd}\|C_{max}$	H	(PVC)
[Dietrich, 1989]	$Rm\|S_{sd}\|\alpha C_{max} + \beta F$	H	deux phases
[Flipo, 1997]	$Rm\|S_{sd}\|\alpha C_{max} + \beta TC$	H	deux phases (prob. véhicules)
[Lee et Kim, 1993]	$Pm\|S_{sd}\|\Sigma T_i$	H	réseau de neurones
[Armentano et Mazzini, 1997]		H	alg. génétique
[Echalier, 1991]	$Rm\|S_{sd}\|\Sigma T_i$	H	recuit simulé
[Bitran et Gilbert, 1990]	$Pm\|C_{sd}, \mathrm{maj}, min\|\Sigma C_{ijk}$	O	PSEP
[Tang et Wittrock, 1985]	$Pm\|S_{nsd}, \mathrm{maj}, min\|C_{max}$	H	2 phases (prob. du sac à dos)
[Wittrock, 1990]			
[Tang, 1990]			
[So, 1990]	$Pm\|S_{nsd}, \mathrm{maj}, min, wlr\|\Sigma w_i C_i$	H	prog. dyn.
[Rajgopal et Bidanda, 1991]	$Pm\|S_{nsd}, \mathrm{maj}, min S_i = S, s_i = s\|C_{max},$	H	proc. de descente
[Dobson *et al.*, 1989]	$Pm\|S_{nsd}, 1\mathrm{batch}, p_i = p_f, w_i = w_f\|\Sigma w_i C_i$	H	
[Monma et Potts, 1993]	$Pm\|\mathrm{prmnt}, S_{nsd}, \mathrm{batch}\|C_{max}$	H	heur. de liste
[Chen, 1993]		H	heur. de liste
[Cheng et chen, 1994]	$Pm\|S_{nsd}, batch, m = 2\|C_{max}$	-	étude de complexité
[Monma et Potts, 1989]	$Pm\|S_{sd}, \mathrm{batch}\|\Sigma w_i C_i$	-	étude de complexité
[Psaraftis, 1980]	$Pm\|S_{sd}, \mathrm{batch}, m = 2, p_i = p_f, w_i = w_f\|\Sigma w_i C_i$	O	
[Ghosh, 1994]	$Pm\|S_{sd}, \mathrm{batch}, p_i = p_f, w_i = w_f\|\Sigma w_i C_i$	O	prog. dyn.
[Ovacik et Uzsoy, 1993]	$Pm\|S_{sd}, S_{ij} \leq p_j\|C_{max}$	H	heur. liste(Anal. du pire cas)
[Kedad *et al.*, 1995]	$Rm\|r_i, S_{sd}\|TC$	H	2 phases (prob véhicules)
[Kedad. 1997]		H	rec. sim., tabou, alg. gen., descente
[Ovacik et Uzsoy, 1995]	$Pm\|r_i, S_{sd}\|L_{max}$	H	(dyn) proc. d'horizon glissant
[Schutten, 1996]	$Pm\|r_i, S_{nsd}, batch\|L_{max}$	O	PSEP

TAB. II.2 – *Temps ou coûts de préparation dans le problème à machines parallèles*

II.1.4 Problème d'ordonnancement d'atelier à cheminement unique

II.1.4.a) Minimisation de la durée totale avec temps de montage et démontage indépendant de la séquence

La prise en compte du temps de préparation sous la forme du temps de montage et de démontage indépendant de la séquence dans le problème d'atelier à cheminement unique a donné lieu à de nombreux travaux. La différence avec le modèle classique est que le temps de préparation est séparé du temps d'exécution et peut ainsi être anticipé. La préparation ne peut intervenir dans le critère à optimiser car on ne peut pas agir sur les temps de préparation. Par contre, elle modifie les contraintes du problème pour la minimisation de la durée totale. Dans ce contexte ($Fm|S_{nsd}, R_{nsd}|C_{max}$), [Sule, 1982] propose un algorithme de liste optimal pour le problème à deux machines. [Szwarc, 1983] et [Proust *et al.*, 1991] proposent des heuristiques inspirées de l'algorithme de Johnson [Johnson, 1954]. Un nombre assez important de méthodes sont proposées pour intégrer d'autres contraintes. Par exemple, des contraintes de précédence généralisées entre travaux ($j-prec$) et des décalages temporels a_i entre opérations successives du même travail (*time-lags*) sont pris en compte dans [Botta et Guinet, 1996]. Ces décalages temporels (positifs ou négatifs) permettent de modéliser aussi bien les temps de transport que le chevauchement lorsqu'un sous-lot peut être transféré à l'étage suivant avant la fin du lot complet. [Cetinkaya, 1994] considère en plus que la taille des sous-lots à transferer (*transfer batches*) peut être déterminée par la procédure d'ordonnancement ($F2|S_{nsd}, R_{nsd}, t.b.|C_{max}$) et propose ainsi une méthode de partage de lot (*lot streaming*). On peut aussi considérer que les zones de stockage intermédiaire des pièces ont des capacités limitées (problèmes notés *block*). Dans [Levner *et al.*, 1995], une méthode basée sur le PVC est proposée pour traiter le cas où un robot unique doit en plus effectuer les transports d'un étage à l'autre.

Les temps de montage et démontage indépendants de la séquence sont utilisés dans le cadre de la technologie de groupe. Lorsque des temps de préparation par familles sont définis, cette méthode regroupe tous les travaux de la même famille et le problème est ainsi décomposé en deux : ordonnancer dans un premier temps les travaux de la même famille regroupés au sein d'un groupe dans le cadre d'un problème d'atelier à cheminement unique sans préparation $Fm||C_{max}$ et dans un deuxième temps, ordonnancer les groupes avec des temps de montage et de démontage ([Baker, 1990]).

II.1.4.b) Temps de préparation dépendant de la séquence

[Gupta et Darrow, 1986] montrent que les ordonnancements de permutation ne sont pas dominants pour la minimisation de la durée totale, en présence de temps de préparation dépendant de la séquence. Néanmoins, pour des raisons de complexité du problème considéré ou de réalité pratique de l'atelier, la plupart des approches rencontrées considèrent uniquement les ordonnancements de permutation.

[Gupta *et al.*, 1995] présentent une PSE pour ce problème. Des heuristiques de liste sont développées dans [Szwarc et Gupta, 1987] (basées sur l'algorithme de Johnson), et dans [Gupta, 1986a] (basées sur le problème du voyageur de commerce). Un modèle linéaire en variables mixtes est proposé dans [Srikar et Ghosh, 1986], amélioré dans [Stafford et Tseng, 1990] et une résolution optimale par la programmation linéaire en est déduite.

Une méthode de recherche locale basée sur la ré-insertion d'un ordre de fabrication est développée dans [Rìos-Mercado et Bard, 1996]. Des résultats sont présentés sur des problèmes comportant jusqu'à 100 travaux et 6 machines avec des temps de calculs maximum de 440 secondes sur SUN-SPARC10.

A notre connaissance, la seule méthode prenant en compte des contraintes temporelles pour les travaux est présentée dans [Parthasarathy et Rajendran, 1997] sous la forme d'un recuit simulé. L'objectif est la minimisation du retard moyen pondéré sans prise en compte de dates de début au plus tôt des travaux ($Fm|S_{sd}|\overline{w_iT_i}$). La génération d'une solution voisine se fait aussi par la ré-insertion d'un ordre de fabrication. Des résultats sur un problème réel de fabrique de forets à 95 ordres de fabrication par mois sont présentés avec un temps de calcul de 11100 secondes sur HP-9000.

II.1.4.c) Résumé des travaux dans le problème d'atelier à cheminement unique

Les principales publications liées aux temps de préparation sont présentées dans la table II.3. Certaines des références sont issues de l'état de l'art effectué dans [Botta, 1996].

Référence	Problème		Méthode employée
[Uskup et Smith, 1975]	$F2\|\tilde{d}_i, S_{sd}\|\sum S_{ij}$	O	PSEP
[Yoshida et Hitomi, 1979]	$F2\|S_{nsd}\|C_{max}$	O	alg. de liste
[Sule, 1982]	$F2\|S_{nsd}, R_{nsd}\|C_{max}$	O	alg. de liste
[Strusevich, 1990]	$F2\|S_{nsd}, R_{nsd}, \mathrm{n}onpmru\|C_{max}$	-	Etude de complexité
[Nabeshima et Maruyama, 1983]	$F2\|S_{nsd}, R_{nsd}, a_i\|C_{max}$	H	heur. basée sur Johnson
[Strusevich, 1990]	$F2\|S_{nsd}, R_{nsd}, \mathrm{no} - wait\|C_{max}$	O	algo. polynomial
[Baker, 1990]	$F2\|S_{nsd}, a_i\|C_{max}$	H	technologie de groupe
[Logendran et Sriskandarajah, 1993]	$F2\|S_{nsd}, \mathrm{bl}ock\|C_{max}$	H	2 phases, insertion de prépa.
[Levner *et al.*, 1995]	$F2\|S_{nsd}$,res111$, a_i, \mathrm{bl}ock\|C_{max}$	O	cas spécial du PVC
[Cetinkaya, 1994],	$F2\|S_{nsd}, R_{nsd}, split\|C_{max}$	O	alg. polynomial
[Sule et Huang, 1983]	$F3\|S_{nsd}, R_{nsd}\|C_{max}$	H	heur. de liste
[Szwarc, 1983]	$Fm\|S_{nsd}, R_{nsd}\|C_{max}$	H	heur. basée sur Johnson
[Proust *et al.*, 1988]			
[Proust *et al.*, 1991]			
[Han et Dejax, 1991]		H	mach. goulots
[Cao et Bedworth, 1992]	$Fm\|S_{nsd}, R_{nsd}, a_{i,v}\|C_{max}$	H	Johnson + descente
[Botta et Guinet, 1996]	$Fm\|j - prec, out - tree, S_{nsd}, R_{nsd}, a_{iv}\|C_{max}$	H	heur. basée sur Johnson
[Corwin et Esogbue, 1974]	$F2\|S_{sd}(sur\ 1\ seule\ mach)\|C_{max}$	O	prog. dyn.
[Gupta et Darrow, 1986]	$F2\|S_{sd}\|C_{max}$	H	étude de complexité
[Szwarc et Gupta, 1987]	$Fm\|S_{sd}\ additifs\|C_{max}$	H	heur. basée sur Johnson
[Gupta, 1975]	$Fm\|S_{sd}\|C_{max}$	O	énum.
[Srikar et Ghosh, 1986]		O	prog. lin.
[Stafford et Tseng, 1990]		O	prog. lin.
[Gupta, 1986a]		H	heur. basée sur PVC
[Das *et al.*, 1995]		H	heur. de liste
[Gupta *et al.*, 1995]		O	PSEP
[Rìos-Mercado et Bard, 1996]		H	GRASP
[Parthasarathy et Rajendran, 1997]	$Fm\|S_{sd}\|\overline{w_iT_i}$	H	recuit simulé

TAB. II.3 – *Temps de préparation dans le problèmes d'atelier à cheminement unique*

II.1.5 Problème d'ordonnancement d'atelier à cheminements multiples

Très peu de travaux ont été menés pour prendre en compte les temps de préparation dans le problème d'ordonnancement d'atelier à cheminements multiples (*job-shop*), très étudié par ailleurs. A notre connaissance, aucune méthode n'a été proposée pour minimiser la durée totale. La seule méthode optimale proposée dans [Gupta, 1982] vise à minimiser la somme des temps de préparation dans le cas où les produits peuvent revenir plusieurs fois sur la même machine (*recrc*) et où une matrice de préparation est définie pour chaque machine. Une PSE basée sur une borne inférieure décomposant le temps total de préparation en un temps cumulé et un temps restant estimé est définie. Aucun résultat expérimental n'est donné. Un certain nombre de règles de priorité sont proposées dans le cadre de l'arrivée dynamique des produits, avec dates de livraison. [Wilbrecht et Prescott, 1969] proposent une règle de priorité similaire à l'heuristique de la plus proche ville, intitulée SIMSET (*Similar Setup*). [Kim et Bobrowski, 1994] proposent une amélioration de cette heuristique. Les opérations sont classées en familles. L'heuristique de Kim intitulée JCR (*Job of Critical Ratio*) sélectionne systématiquement une opération de la même famille que la dernière opération ordonnancée. Dans le cas où le changement de famille est obligatoire, l'opération est sélectionnée par la règle du coefficient critique CR (*critical ratio*) connue pour son efficacité dans la tenue des délais. L'inconvénient de cette méthode est que la sélection systématique d'une opération de même famille, peut pénaliser la tenue des délais des autres produits. [Billaut, 1993] propose une règle de priorité plus fine dans le cadre d'un ordonnancement statique avec dates de début au plus tôt. Une marge est estimée pour chaque opération candidate. Une opération de même famille que la dernière opération ordonnancée n'est sélectionnée que si cette décision ne consomme pas toute la marge allouée aux autres opérations. Si des opérations plus urgentes doivent être ordonnancées, la sélection se fait par une règle proche de CR. Dans le chapitre 5 (paragraphe V.2.3), nous montrons par un contre-exemple qu'il n'existe pas forcément d'heuristique de liste classique générant une solution optimale pour la minimisation de la durée totale dans le problème d'atelier à cheminements multiples avec des temps de préparation dépendant de la séquence. La seule alternative aux heuristiques de liste est proposée dans [Ovacik et Uzsoy, 1994b] dans un environnement dynamique par une procédure d'horizon glissant.

Toutefois, certains travaux rapportés dans le paragraphe suivant sont des généralisations du problème d'atelier à cheminements multiples et peuvent être appliqués à ce problème. Les méthodes proposées sont présentées dans la table II.4.

Référence	Problème		Méthode employée
[Wilbrecht et Prescott, 1969]	$Jm\|r_i, S_{sd}\|TC$	H	(dyn) heur. de liste (SIMSET)
[Hershauer, 1970]		H	(dyn) heur. de liste
[Flynn, 1987]		H	(dyn) heur. de liste (RL)
[Kim et Bobrowski, 1994]		H	(dyn) heur. de liste (JCR)
[Kim et Bobrowski, 1995]		H	(dyn) évaluation du lancement
[Gupta, 1982]	$Jm\|S_{sd}, \text{recrc}\|\sum S_{ij}$	O	PSEP
[Billaut, 1993]	$Jm\|r_i, S_{sd}, recrc, d_i\|\text{admiss.}$	H	heur de liste (méthode ORABAID)
[Ovacik et Uzsoy, 1994b]	$Jm\|r_i, S_{sd}\|L_{max}$	H	(dyn) proc. d'horizon glissant

TAB. II.4 – *Temps de préparation dans les problèmes d'atelier à cheminements multiples*

II.1.6 Problèmes généralisés

Nous avons regroupé dans ce paragraphe et dans la table II.5, des travaux généralisant les problèmes d'atelier à cheminement unique et à cheminements multiples et le problème d'ordonnancement de projet à moyens limités.

II.1.6.a) Préparation liée à des ressources multiples

[Bovet et Petrini, 1980] traitent un cas où des opérations indépendantes nécessitent simultanément plusieurs ressources (des bandes magnétiques), chacune d'elles devant être préparée (montée sur des disques disponibles en nombre limité) avant d'être utilisée par l'opération. L'opération doit ainsi attendre que toutes ses ressources soient préparées avant d'être exécutée. On a ainsi un problème de préparation multi-ressources.

[Assad *et all*, 1995] considèrent un cas industriel de production de matelas. Ce problème est décrit comme un problème à machines non identiques avec la prise en compte des opérateurs de la facon suivante : la préparation et l'exécution sont réalisées à la fois par une machine et un opérateur (le même pour la préparation et l'exécution). De plus, les opérateurs sont spécialisés : chaque opérateur ne peut utiliser que certaines machines. L'objectif est la minimisation d'un coût intégrant les heures supplémentaires des opérateurs, l'utilisation des machines et le retard des ordres de fabrication. Le problème est résolu par une heuristique d'insertion des ordres de fabrication.

Divers modèles intégrant la gestion des outils ont été proposés dans le problème à une machine [Daoud et Purcheck, 1981], [Tang et Denardo, 1988], [Crama *et al.*, 1994]. [Hertz et Widmer, 1995] traite le cas d'un atelier à cheminements multiples où les machines possèdent des magasins d'outils à capacité limitée et où les opérations à ordonnancer sur la machine utilisent un certain nombre d'outils. Un changement d'outil est nécessaire uniquement lorsque un des outils requis par une opération n'est pas dans le magasin. Un des problèmes principaux associés est le choix judicieux des outils à mettre dans le magasin. Une méthode tabou est proposée pour minimiser un coût qui dépend à la fois du nombre de changements d'outil, de la durée totale, du nombre de produits en retard et d'un dépassement d'une période de production allouée. Deux méthodes tabou différentes sont comparées sur des problèmes comportant jusqu'à 10 produits et 10 machines

II.1.6.b) Problème d'atelier à cheminement unique flexible

Dans ce contexte, quelques méthodes sont proposées pour la minimisation de la durée totale en présence de temps de préparation. [Aghezzaf *et al.*, 1995] proposent une méthode de programmation linéaire pour résoudre ce problème avec temps de préparation dépendant de la séquence en le décomposant en plusieurs sous problèmes à machines parallèles. [Botta, 1996] propose cinq heuristiques pour minimiser la durée totale et le retard maximum pour un problème issu de l'industrie textile comportant des contraintes de précédence généralisées entre travaux, des décalages temporels, des calendriers et des temps de montage et démontage indépendants de la séquence. Le problème d'affectation et le problème d'ordonnancement sont résolus en séquence pour se ramener successivement à un problème de machines parallèles et à un problème d'atelier à cheminement unique non flexible.

II.1.6.c) Problème d'atelier à cheminements multiples généralisé

Pour résoudre le problème d'atelier à cheminements multiples classique, une méthode très efficace est la procédure de machines goulots (*Shifting Bottleneck*) (SB) de [Adams *et al.*, 1988] dont l'idée de base est de résoudre une succession de problèmes à une machine en identifiant à chaque étape la machine goulot. Certaines approches visent à étendre cette méthode à des cas pratiques comportant notamment des temps de préparation. [Ivens et Lambrecht, 1994] y incluent les temps de préparation indépendants de la séquence. [Schutten, 1996] y inclut les temps de préparation dépendant de la séquence. Dans les deux cas, le problème est représenté par un graphe disjonctif inspiré de [Roy et Sussman, 1964] et l'ordonnancement est représenté par un graphe conjonctif obtenu en orientant tous les arcs disjonctifs. Un arc "machine" entre deux opérations successives (i,j) et (k,l) sur une machine nécessitant d'être préparée est valué par le temps de préparation $s_{ij,kl}$. [Schutten, 1996] présente les résultats de cette procédure sur un problème d'atelier de montage où des contraintes de précédence conjonctives existent entre les produits, assemblés sur une machine. Chaque produit appartient à une certaine famille, et un temps de préparation dépendant de la séquence est nécessaire entre deux produits de familles différentes. La procédure SB est testée pour la minimisation du plus grand retard, du retard moyen et du nombre de produits en retard. Des résultats très supérieurs à un ensemble d'heuristiques de liste sont obtenus en ordonnancement statique sur des problèmes comportant 500 produits pour un temps de calcul de 21 secondes et en ordonnancement dynamique sur des problèmes comportant jusqu'à 2000 produits pour un temps de calcul de 160 secondes (sur HP 9000). Ces problèmes, issus d'un cas réel, comportaient seulement 4 machines, ce qui explique les bons résultats de la procédure SB bien adaptée à des problèmes "horizontaux".

II.1.6.d) Problème d'ordonnancement de projet à moyens limités

A notre connaissance seules deux méthodes sont proposées pour prendre en compte les temps de préparation dans le problème très général d'ordonnancement de projet à moyens limités (*Resource constrained project scheduling problem*), noté RCPSP. Dans les deux cas, l'objectif est la minimisation de la durée totale du projet. [Kaplan, 1991] considère un problème préemptif ou un temps de préparation est nécessaire lors de la reprise de l'exécution d'une opération. Une heuristique de programmation dynamique est proposée, basée sur la modélisation du problème comme un RCPSP à temps unitaire. [Kolish, 1995] propose une modélisation en variables mixtes du problème, sous la forme d'un RCPSP à modes multiples. Chaque opération possède deux modes, un mode correspondant à la durée de l'activité sans préparation, un autre mode correspondant à la durée de l'activité augmentée d'un temps de préparation. Chaque opération appartient à une famille et le temps de préparation est nécessaire lorque deux activités de familles différentes se succèdent sur une ressource particulière. Ce temps dépend uniquement de la famille de l'opération suivante, et est inclus dans la durée de l'opération. Toutes les ressources de l'opération subissent ainsi la préparation même si une activité ne peut avoir qu'une seule ressource "concernée" par la préparation disponible en une seule unité (disjonctive). En utilisant la programmation linéaire, des problèmes à seulement 10 activités et 2 ressources sont résolus de façon optimale. Une heuristique de liste est proposée et testée par des lancements multiples aléatoires comparativement à la procédure optimale. La solution optimale est atteinte dans tous les cas au bout de 30 lancements aléatoires. Toutefois, les problèmes générés semblent d'une taille trop petite pour conclure sur la performance de l'heuristique. Pour un lancement unique, une déviation moyenne de $4,3\%$ est obtenue par rapport à la solution optimale.

Référence	Problème		Méthode employée
[Bovet et Petrini, 1980]	Multi-res.	H	simulation
[Assad *et all*, 1995]	$Rm\|res111, s_{sd}\|TC$	H	insertion
[Daoud et Purcheck, 1981]	$1\|chgtd'outils\|TC$	H	plus proche voisin
[Bard, 1988]	$1\|chgtd'outils\|nbdechang.$	H	relax. lagrangienne
[Tang et Denardo, 1988]		H	"keep Tool Needed Soonest"
[Crama *et al.*, 1994]		H	alg. basé sur le PVC
[Widmer, 91]	$Jm\|chgtd'outils\|TC$	H	rech. locale (tabou)
[Hertz et Widmer, 1995]		H	rech. locale (tabou)
[Gupta et Tunc, 1994]	$FF2\|m_1 = 1, m_2 \geq 1, S_{nsd}, R_{nsd}\|C_{max}$	H	2 phases (Sule + règle LBM)
[Sherali *et al.*, 1990]	$FF2\|r_i, d_i, S_{sd}, DT, wlr\|TC$	H	2 phases
[Proust et Grunenberg, 1995]	$FF2\|S_{sd}, S_{nsd}, maj/min, a_i, DT, split\|TC$	H	SIAD
[Li, 1996]	$FF2\|S_{nsd}, maj/min, split, a\|C_{max}$	H	réduc. temps de réglage
[Egbelu, 1991]	$FFs\|S_{nsd}, a_{i,v}\|C_{max}$		
[Aghezzaf *et al.*, 1995]	$FFs\|S_{sd}\|C_{max}$	H	prog. lin.
[Elmaghraby et Karnoub, 1995]			
[Botta, 1996]	$FFs\|j - prec, out - tree, a_{i,v},$ $H_{i,v}, S_{nsd}, R_{nsd}\|C_{max}, L_{max}$	H	2 phases (5 heur.)
[Ivens et Lambrecht, 1994]	job shop généralisé, S_{nsd}	H	mach. goulôt
[Schutten, 1996]	job shop généralisé, S_{sd}	H	mach. goulôt
[Brucker et Thiele]	Job Shop généralisé, S_{sd}	O	PSEP
[Kaplan, 1991]	PSPST pmnt,S_{nsd}	H	prog. dyn.
[Kolish, 1995]	PSPST S_{nsd}, s_{inclus}, batch	O	prog. lin.
		H	alg. de liste

TAB. II.5 – *Temps de préparation dans certains problèmes généralisés*

II.1.7 Temps de préparation dans les méthodes d'aide à la décision pour l'ordonnancement

Dans les systèmes d'aide à la décision basés sur l'intelligence artificielle, des systèmes à base de règle comportant des critères concernant la préparation sont développés. C'est le cas de [Bensana, 1987] pour le système OPAL dans le cadre de l'analyse sous contraintes et plus tard [Lecomte, 93] dans une approche d'ordonnancement par règles de priorité. Dans un cadre de satisfaction et de propagation de contraintes, [Lepape, 88] prend en compte les contraintes de temps de préparation dépendant de la séquence et les changements d'outils dans le système d'ordonnancement SOJA.

Pour les méthodes d'aide à la décision basées sur la caractérisation de familles de solutions, dans lesquelles s'incrit notre approche, [Baptiste, 1985] propose une méthode pour construire des séquences de groupes d'opérations minimisant les temps de préparation dépendant de la séquence (temps de réglage) en utilisant les arbres PQ, représentant les permutations d'un ensemble donné possédant des propriétés particulières. Les degrés de liberté résultant permettent d'optimiser éventuellement d'autres critères et surtout d'absorber les aléas en temps réel. L'atelier considéré est à cheminement unique. [Zouhri, 1993] utilise une extension des arbres PQ permettant de respecter les contraintes de précédence : les arbres PQR. Cette extension est appliquée au même problème.

[Cho, 1989] génère une séquence de groupes d'opérations permutables pré-admissible en regroupant les opérations de même famille de préparation, puis recherche l'admissibilité par des scissions, dans le contexte d'un atelier à cheminements multiples. Dans le même contexte [Billaut, 1993], propose une heuristique de génération d'une séquence de groupes d'opérations permutables en effectuant un compromis entre la minimisation du temps de préparation et le respect des délais des ordres de fabrication.

II.1.8 Conclusion

Les temps de préparation restent relativement peu abordés dans les problèmes d'ordonnancement, en particulier pour les problèmes d'atelier multi-opérations en présence de contraintes temporelles. Néanmoins, cet état de l'art permet de dégager certaines propriétés sur le problème lui-même et sur les méthodes de résolutions adaptées.

- L'introduction des temps de préparation change profondément le problème, ce qui est démontré par l'augmentation de la complexité et la perte de certaines propriétés de dominance.
- L'objectif de minimisation des temps de préparation peut être antagoniste avec la minimisation des critères classiques.
- Les similarités avec les problèmes de voyageur de commerce et de tournées de véhicules sont fortes, mais il existe des caractéristiques spécifiques à la préparation (notion de famille, temps de montage et de démontage,...).
- Des hypothèses réalistes sur les temps de préparation permettent de simplifier le problème et de dégager des relations de dominance (symétrie, inégalité triangulaire).

En ce qui concerne les méthodes de résolution on peut tirer les conclusions suivantes :

- Les algorithmes du PVC et du problème de véhicules sont utilisés intensivement même dans les cas où des adaptations importantes sont nécessaires pour se ramener à un des deux problèmes. On peut noter toutefois l'émergence de méthodes utilisant explicitement les propriétés de la préparation.
- Pour résoudre les éventuels problèmes d'affectation, les algorithmes en deux phases sont plus utilisés que les approches de résolution intégrées.
- La vision de la préparation comme une opération particulière peut faciliter la résolution.
- Les méthodes de voisinage sont de plus en plus utilisées et obtiennent de façon générale de bons résultats même si les procédures par machine goulôt semblent plus efficaces pour les problèmes proches du problème à une machine.

II.2 Modèle retenu pour la prise en compte de la préparation

Le pararaphe II.2.1 effectue la synthèse des différents modèles d'atelier et de préparation rencontrés dans la littérature pour justifier la proposition d'un nouveau modèle. Ce modèle général est présenté dans le paragraphe II.2.2.

II.2.1 Positionnement par rapport aux modèles existants

Dans le paragraphe II.2.1.a), les différents modèles d'atelier répondant à nos attentes sont comparés. Le paragraphe II.2.1.b), montre la complémentarité des différents modèles de préparation. Dans les deux cas, la nécessité d'un modèle intégré est mise en évidence.

II.2.1.a) Modèle d'atelier

On considère dans le cadre de cette étude un atelier à cheminements multiples flexible généralisé. Une opération a besoin pour son exécution de plusieurs ressources simultanément. Cette caractéristique définit un problème multi-ressources lié à l'opération. De plus, plusieurs ensembles de ressources différents sont possibles pour exécuter l'opération. La durée de l'opération peut ou non varier en fonction de l'ensemble de ressources choisi pour son exécution. Cette caractéristique définit un problème d'affectation associé à l'opération.

On trouve dans la littérature trois classes de modèles exprimant la flexibilité en multi-ressources, résumés dans la figure II.2.

- Dans le *modèle d'atelier avec pools* utilisé dans [Billaut, 1993] et [Dauzère Péres et al, 1996] (voir paragraphe I.1.2.a), chaque opération nécessite un ensemble de ressources simultanément. Chaque ressource est sélectionnée dans un ensemble pré-défini appelé pool. Des différences existent au sein de ces modèles. Dans [Billaut, 1993], les pools requis par l'opération sont nécessairement disjoints, c'est-à-dire qu'une ressource donnée ne peut se trouver que dans un des pools de l'opération. Pour simplifier l'acquisition des données dans les problèmes réels, un des pools est défini comme guidant et la durée de l'opération ne dépend que de la ressource sélectionnée au sein de ce pool guidant. Dans [Dauzère Péres et al, 1996], les pools ne sont pas nécessairement disjoints mais la même ressource ne peut être sélectionnée qu'une fois. L'opération a une durée sur chaque ressource et la durée totale issue de l'affectation est la plus grande des durées sur chaque ressource sélectionnée.

- Le *modèle multi-modes* est issu de l'ordonnancement de projet à moyens limités (dit RCPSP) [Kolish, 1995], mais il peut être utilisé en ordonnancement d'atelier [Daniels et Mazzola, 1993]. Dans ce modèle, les ressources peuvent être disponibles en plusieurs lignes de charge (ressources cumulatives). Une opération est associée à un ensemble de modes. Un mode est défini par une liste de couples (ressource requise, quantité nécessaire) et par une durée d'exécution. La sélection d'un mode résout ainsi complètement le problème d'affectation.

- Le *modèle du compromis temps/ressource* (voir par exemple [Demeulemeester *et al.*, 1996]) est un cas particulier du modèle précédant et considère qu'une opération utilise une seule ressource cumulative. La durée de l'opération est une fonction non croissante du nombre de lignes de charge requises.

On remarque, au travers de cette classification sommaire, que le modèle multi-modes est le plus général des trois. Par contre, le modèle des pools est plus proche de la réalité des ateliers car il permet de regrouper naturellement dans un pool les ressources à fonctionnalité équivalente et de les classer en fonction de leur performance. Il permet ainsi de caractériser l'ensemble des affectations sans énumérer l'ensemble des solutions possibles. Par exemple dans la figure II.2, le modèle multi-modes équivalent au modèle à trois pools représentés dans la colonne de gauche contiendrait $3 \times 3 \times 4 = 36$ modes.

[Billaut, 1993] propose une intégration du modèle de compromis temps/ressource dans le modèle avec pools : le pool guidant peut contenir des ressources cumulatives. La durée de l'opération dépend alors du nombre de lignes de charge requises sur la ressource sélectionnée. Par contre, ce modèle étendu ne permet pas d'exprimer la spécialisation de certaines ressources. Ce cas intervient par exemple lorsqu'un opérateur A est compétent pour utiliser les machines M_1 et M_2 tandis qu'un autre opérateur B ne peut utiliser que la machine M_1. Il ne permet pas non plus d'exprimer que le nombre de ressources simultanément requises peut être variable. Une opération peut par exemple être exécutée par une machine à commande manuelle ou à commande numérique. Dans le premier cas, la présence d'un opérateur est requise mais pas dans le second cas. Le modèle multi-modes permet d'exprimer ces caractéristiques. Pour corriger les défauts du modèle avec pools sans toutefois énumérer l'ensemble des affectations possibles, le modèle proposé dans le paragraphe II.2.2, introduisant la notion de *mode étendu*, est un compromis entre les trois approches.

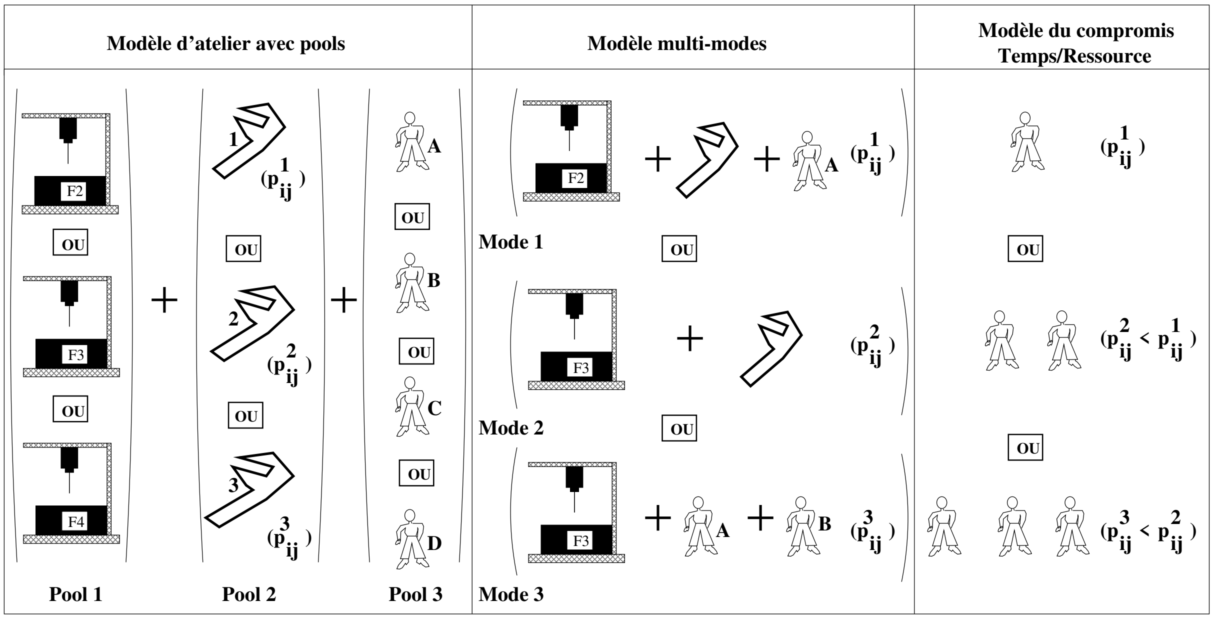


FIG. II.2 – *Trois modèles pour la flexibilité multi-ressources*

II.2.1.b) Modèle de préparation

La prise en compte explicite des temps de préparation a été retenue sous l'hypothèse qu'ils ont une influence importante dans les ateliers sur la durée totale de l'ordonnancement et surtout sur la tenue des délais des ordres de fabrication. Les coûts de préparation ne seront pas directement pris en compte dans le modèle retenu. Les différents modèles des temps de préparation décrits dans le paragraphe II.1 peuvent être intégrés. Dans un contexte multi-ressources, notre interprétation des temps de montage et démontage indépendants de la séquence est la suivante : le temps de montage est le temps nécessaire pour associer un sous-ensemble des ressources requises par une opération ; le temps de démontage est le temps nécessaire pour dissocier ce même sous-ensemble. Entre deux opérations successives utilisant un même sous-ensemble de ressources, aucun temps de montage ni de démontage n'est nécessaire. Par contre, un temps de préparation dépendant de la séquence lié à la succession de deux opérations possédant des caractéristiques différentes sur un sous-ensemble de ressources communes peut être en plus nécessaire.

La prise en compte conjointe de S_{nsd}, R_{nsd} et S_{sd} génère une activité complexe de préparation.

Enfin, le modèle proposé intègre la présence de ressources complémentaires spécifiques pour la préparation. A notre connaissance, cette caractéristique n'a pas été considérée jusqu'à présent dans les méthodes d'ordonnancement traitant les temps de préparation (voir le paragraphe II.1). La complexité de l'activité de préparation et la prise en compte de ces ressources nous amène à définir des *opérations de préparation*.

II.2.2 Description du modèle retenu

Le modèle est décrit par la définition des composantes de l'atelier et des contraintes qui les lient. Les ressources de l'atelier et la notion de pools sont définis en II.2.2.a). Les ordres de fabrication, les *opérations d'exécution* et les *modes étendus d'exécution* sont définis en II.2.2.b). Le paragraphe II.2.2.c) définit l'activité de préparation qui amène à considérer des *ressources principales*, des *ressources complémentaires*, des *opérations de préparation* et des *modes étendus de préparation*. Les contraintes de succession des opérations sur les ressources principales sont décrites en II.2.2.d). Les contraintes de succession des opérations sur les ressources complémentaires sont décrites en II.2.2.e). Un exemple illustratif est présenté en II.2.2.f).

II.2.2.a) Ressources

Nous considérons un atelier composé d'un ensemble $\mathcal{R}$ de M ressources $k = 1, ..., M$ de deux types :

- les ressources disjonctives qui ne peuvent exécuter qu'une opération à la fois,
- les ressources cumulatives qui peuvent exécuter un nombre limité d'opérations simultanément. Une ressource cumulative est composée d'un ensemble de lignes de charge totalement interchangeables telles que plusieurs lignes de charge peuvent être requises pour exécuter une même opération. On note E_k le nombre total de lignes de charge d'une ressource k. Remarquons que si k est disjonctive, alors $E_k = 1$.

Les ressources de l'atelier sont regroupées dans des ensembles de ressources appelés pools. Un pool de ressources contient des ressources semblables par leur fonctionnalité mais éventuellement différentes en terme de performance.

II.2.2.b) Ordres de fabrication

L'atelier doit réaliser un ensemble de $N, N \geq 1$ ordres de fabrication. Chaque ordre de fabrication i possède les caractéristiques suivantes:

- une date de début au plus tôt r_i,
- une date de livraison d_i,
- un ensemble de h_i, $h_i \geq 1$ opération(s) à réaliser selon une gamme g_i telle qu'une opération peut avoir plusieurs précédentes et plusieurs suivantes. On modélise les contraintes de précédence de g_i par un graphe potentiels-tâches $G_i = \{X_i, U_i\}$ tel que chaque sommet appartenant à X_i est une opération de i et chaque arc appartenant à U_i est une contrainte de précédence entre deux opérations de i.

Chaque opération (i, j) d'un ordre de fabrication i , appelée opération d'exécution, est caractérisée par :

- un ensemble de m_{ij} **modes étendus d'exécution**. Un ensemble $\mathcal{P}_{ij}^m$ de M_{ij}^m pools est donné pour chaque mode étendu $m = 1, ..., m_{ij}$. Une ressource doit être sélectionnée dans chaque pool $P_{ij}^{mr} \in \mathcal{P}_{ij}^m$, $r = 1, ..., M_{ij}^m$ pour l'exécution de l'opération. Ces pools ne sont pas nécessairement disjoints mais la même ressource ne peut être sélectionnée qu'une fois. Le nombre de lignes de charge e_{ij}^k que peut utiliser l'opération (i, j) sur une ressource k dans le mode étendu m, est compris entre une borne inférieure e_{ij-}^{mk} et une borne supérieure e_{ij+}^{mk}. Un problème d'affectation dans un contexte multi-ressources cumulatif est ainsi caractérisé. La résolution du problème d'affectation suppose à la fois le choix du mode étendu et la sélection d'une ressource dans chaque pool du mode étendu sélectionné. La solution de ce problème est représentée par une *occupation des ressources* $l_{ij} = (e_{ij}^1, ..., e_{ij}^M)$ donnant pour chaque ressource k de l'atelier le nombre de lignes de charge e_{ij}^k utilisées par (i, j). Cette occupation des ressources correspond à un mode d'exécution au sens classique [Kolish, 1995]. Les lignes de charge d'une ressource cumulative k étant identifiées et numérotées de 1 à E_k, on définit en outre une *occupation des lignes de charge* pour l'opération (i, j) sur la ressource k $l_{ij}^k = \{e_{ij}^{k1}, ..., e_{ij}^{kE_k}\}$. On a $e_{ij}^{kl} = 1$ si (i, j) est affectée à la ligne de charge l de la ressource k et $e_{ij}^{kl} = 0$ sinon. L'ensemble des ressources k de $\mathcal{R}$ pour lesquelles $e_{ij}^k > 0$ est noté $\mathcal{R}_{ij}$.
- une durée p_{ij} qui dépend d'une seule des ressources utilisées par l'opération, appelée ressource guidante. La ressource guidante est choisie au sein du pool guidant $Pg_{ij}^m \in \mathcal{P}_{ij}^m$ qui dépend du mode étendu m sélectionné. Les autres pools de l'opération sont appelés pools annexes. Les ressources contenues dans ces pools sont appelées ressources annexes. On a $p_{ij} = p_{ij}^{mge}$ où p_{ij}^{mge} est une fonction non croissante du nombre e de lignes de charge utilisées sur la ressource guidante g si (i, j) est exécutée dans le mode étendu m.
- une date de début t_{ij}. Cette date est soumise aux contraintes de gamme qu'on exprime par :

$$t_{ij} \geq t_{il} + p_{il} \qquad i = 1, ..., n; j = 1, ..., h_i + 1; \{(i, l)(i, j)\} \in U_i \tag{II.1}$$

$$t_{i1} \geq r_i \qquad i = 1, ..., n \tag{II.2}$$

L'ensemble des opérations précédentes de (i, j) dans la gamme de l'ordre de fabrication i est noté $prec^i_{ij}$. L'ensemble des suivantes est noté $succ^i_{ij}$.

L'ensemble des opérations d'exécution est appelé $\mathcal{O}$.

La figure II.3 donne un exemple de modes étendus pour une opération d'exécution. On suppose que cette opération (i, j) possède trois modes étendus d'exécution ($m_{ij} = 3$). Le premier mode étendu correspond à une exécution avec une machine à commande numérique et ne nécessite pas la présence d'opérateurs ($M^1_{ij} = 1$). Le pool (guidant) contient trois machines à commande numérique MN_1, MN_2 et MN_3 susceptibles d'exécuter l'opération mais avec des performances différentes. En effet, la durée de (i, j) sur la machine MN_1 est 5, sur MN_2, 6 et sur MN_3, 7. Les modes étendus 2 et 3 correspondent à l'exécution de (i, j) sur des machines à commande manuelle MM_1, MM_2 et MM_3 et nécessitent la présence d'opérateurs ($M^2_{ij} = 2$ et $M^3_{ij} = 2$). Le mode étendu 2 correspond à l'exécution de l'opération par l'équipe d'opérateurs 1 représentée par une ressource cumulative à deux lignes de charge $OPER1$. Ces deux opérateurs ont des compétences équivalentes et peuvent utiliser les trois machines. La durée d'exécution de (i, j) dépend du nombre d'opérateurs utilisé simultanément mais pas de la machine sélectionnée car c'est le pool contenant $OPER1$ qui est guidant. Le pool contenant les machines est un pool annexe. Le mode étendu 3 correspond à l'exécution de l'opération par l'équipe d'opérateurs 2 représentée par une ressource cumulative à deux lignes de charge $OPER2$. Ces deux opérateurs ont des compétences équivalentes mais ne peuvent utiliser que la machine $MM1$ et la machine $MM2$. La sélection du mode étendu 3 interdit l'utilisation de la machine $MM3$.

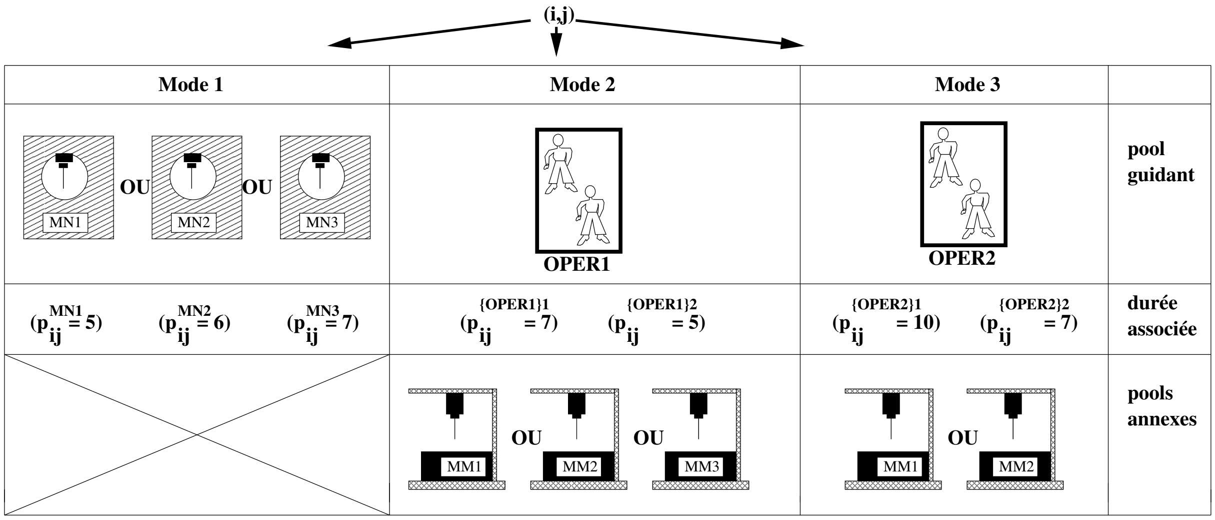


FIG. II.3 – *Un exemple de modes étendus associés à une opération d'exécution*

II.2.2.c) Activité de préparation

Ressources principales et complémentaires Dans les cas pratiques, une activité de préparation préalable à la réalisation d'une opération d'exécution peut être nécessaire sur un ensemble de ressources. Nous supposons que l'ensemble $\mathcal{R}$ des ressources de l'atelier est décomposable en deux sous-ensembles :

- l'ensemble $\mathcal{R}b$ des M_b ressources principales. Ces ressources peuvent nécessiter une préparation avant l'exécution d'une opération. On suppose que seules des ressources disjonctives peuvent être principales.
- l'ensemble $\mathcal{R}c$ des $M_c = M - M_b$ ressources complémentaires. Ces ressources ne nécessitent pas de préparation préalable à l'exécution d'une opération. Certaines de ces ressources, disjonctives ou cumulatives, peuvent représenter des opérateurs.

L'ensemble $\mathcal{P}_{ij}^m$ des pools définis pour chaque mode étendu m d'une opération d'exécution (i, j) peut être décomposé en deux sous-ensembles :

1. l'ensemble $\mathcal{P}b_{ij}^m$ des Mb_{ij}^m pools de ressources principales dans le mode m,
2. l'ensemble $\mathcal{P}c_{ij}^m$ des Mc_{ij}^m pools de ressources complémentaires dans le mode m.

L'occupation des ressources l_{ij} est décomposée en une occupation des ressources principales lb_{ij} et une occupation des ressources complémentaires lc_{ij}.

L'ensemble $\mathcal{R}_{ij}$ des ressources utilisées par (i, j) peut être décomposée en deux sous-ensembles

1. l'ensemble $\mathcal{R}b_{ij}$ des ressources principales $k \in \mathcal{R}b$ utilisées par (i, j) (telles que $e_{ij}^k = 1$).
2. l'ensemble $\mathcal{R}c_{ij}$ des ressources complémentaires $k' \in \mathcal{R}c$ utilisées par (i, j) (telles que $e_{ij}^{k'} > 0$).

Il est à noter que les notions de ressource principale et de ressource complémentaire caractérisent la possibilité d'une activité de préparation sur ces ressources. Ces notions ne sont aucunement liées aux notions de ressource guidante et de ressource annexe qui sont utilisées uniquement pour le calcul de la durée d'une opération (voir II.2.2.b), et plus loin II.2.2.c)).

Notion d'état d'une ressource principale La préparation peut être définie comme une activité nécessaire pour faire passer la ressource d'un certain état consécutif à l'exécution d'une opération d'exécution à un autre état permettant l'exécution d'une nouvelle opération d'exécution.

Chaque ressource principale peut être au cours du temps dans un des deux états suivants : l'état “associée” et l'état “isolée”. Par exemple, un outil est isolé s'il n'est monté sur aucune machine. A chaque instant, une ressource principale donnée est soit isolée, soit associée avec un seul ensemble de ressources principales. Un ensemble de ressources principales associées est appelé une *association*. L'ensemble $\mathcal{R}b_{ij}$ est l'association définie par l'occupation des ressources de (i, j)

Pour caractériser complètement l'état d'une ressource principale ou d'une association de ressources principales, on introduit un ensemble $\mathcal{T}$ de types d'opérations d'exécution. On considère que chaque opération d'exécution (i, j) appartient à un certain type RT_{ij}. Chaque type correspond à un état qu'une ressource principale doit atteindre pour exécuter l'opération. L'état d'une ressource correspond à chaque instant à un et un seul type d'opérations d'exécution. Les trois conditions suivantes sont caractéristiques de l'activité de préparation.

Condition 1 Une opération d'exécution ne peut être engagée sur un ensemble de ressources principales que si ces ressources principales sont associées.

Condition 2 Une opération d'exécution (i, j) ne peut être engagée sur un ensemble de ressources principales que si chacune de ces ressources est dans l'état correspondant au type RT_{ij}.

La condition suivante est considérée par la suite mais elle peut être relâchée selon la nature de l'activité de préparation dans l'atelier modélisé.

Condition 3 Seules des ressources principales dans l'état correspondant au même type peuvent être associées.

Opérations de préparation et modes étendus de préparation des ressources
On considère que l'activité de préparation d'un ensemble de ressources principales est décomposable en trois opérations élémentaires de préparation.

i) Une **opération de montage** (notée sm) qui fait passer chaque ressource principale de l'état "isolée" à l'état "associée". Cette opération est nécessaire pour satisfaire la condition 1. Une opération de montage a une date de début, une durée et nécessite éventuellement des ressources complémentaires spécifiques déterminées. Toute opération de montage est associée à un mode étendu de montage μ à sélectionner parmi μ_{sm} modes étendus possibles. Un mode étendu de montage μ est caractérisé par :

- un ensemble $\mathcal{P}b_{sm}^{\mu}$ de Mb_{sm}^{μ} pools de ressources principales. Le mode μ regroupe toutes les opérations de montage relatives à toutes les associations de ressources principales obtenues en sélectionnant une ressource principale dans chaque pool $Pb_{sm}^{\mu r} \in \mathcal{P}b_{sm}^{\mu}$, $r = 1, ..., Mb_{sm}^{\mu}$. L'ensemble des associations concernées par μ est noté $\mathcal{A}^{\mu}$.
- un ensemble $\mathcal{P}c_{sm}^{\mu}$ de Mc_{sm}^{μ} pools de ressources complémentaires de montage. Pour réaliser le montage d'une association $a \in \mathcal{A}^{\mu}$, une ressource doit être sélectionnée dans chaque pool $Pc_{sm}^{\mu r} \in \mathcal{P}c_{sm}^{\mu}$, $r = 1, ..., Mc_{sm}^{\mu}$. Le nombre de lignes de charge nécessaires au montage d'une association $a \in \mathcal{A}^{\mu}$ sur une ressource complémentaire cumulative k est compris entre $e_{sm-}^{\mu k}$ et $e_{sm+}^{\mu k}$.
- une durée de montage $p_{sm}^{\mu ge}$ dépendant d'une seule des ressources principales ou complémentaires appelée ressource guidante de montage g sélectionnée au sein du pool guidant $Pg_{sm}^{\mu} \in \mathcal{P}b_{sm}^{\mu} \cup \mathcal{P}c_{sm}^{\mu}$ et du nombre de lignes de charge e utilisées sur g.

L'ensemble des opérations de montage nécessaires est noté $\mathcal{S}m$.

ii) Une **opération de démontage** (notée sd) qui fait passer un ensemble de ressources principales de l'état "associée" à l'état "isolée". Une opération de démontage est nécessaire chaque fois que l'une des ressources concernées par une association à réaliser se trouve associée au sein d'un autre ensemble de ressources principales. Elle permet le respect de la condition 1.

Une opération de démontage a une date de début, une durée et nécessite éventuellement des ressources complémentaires spécifiques déterminées. Toute opération de démontage est associée à un mode étendu de démontage δ à sélectionner parmi δ_{sd} modes étendus possibles. Un mode étendu de démontage δ est caractérisé par :

- un ensemble $\mathcal{P}b_{sd}^{\delta}$ de Mb_{sd}^{δ} pools de ressources principales. Le mode δ regroupe toutes les opérations de démontage relatives à toutes les associations de ressources principales obtenues en sélectionnant une ressource principale dans chaque pool $Pb_{sd}^{\delta r} \in \mathcal{P}b_{sd}^{\delta}$, $r = 1, ..., Mb_{sd}^{\delta}$. L'ensemble des associations concernées par δ est noté $\mathcal{A}^{\delta}$.

- un ensemble $\mathcal{P}c_{sd}^{\delta}$ de Mc_{sd}^{δ} pools de ressources complémentaires de démontage. Pour réaliser le démontage d'une association $a \in \mathcal{A}^{\delta}$, une ressource doit être sélectionnée dans chaque pool $Pc_{sd}^{\delta r} \in \mathcal{P}c_{sd}^{\delta}$, $r = 1, ..., Mc_{sd}^{\delta}$. Le nombre de lignes de charge nécessaires au démontage d'une association $a \in \mathcal{A}^{\delta}$ sur une ressource complémentaire cumulative k est compris entre $e_{sd-}^{k\delta}$ et $e_{sd+}^{k\delta}$.

- une durée de démontage $p_{sd}^{\delta ge}$ dépendant d'une seule des ressources principales ou complémentaires appelée ressource guidante de démontage g sélectionnée au sein du pool guidant $Pg_{sd}^{\delta} \in \mathcal{P}b_{sd}^{\delta} \cup \mathcal{P}c_{sd}^{\delta}$ et du nombre de lignes de charge e utilisées sur g.

L'ensemble des opérations de démontage nécessaires est noté $\mathcal{S}d$.

iii) Une **opération de changement de type** (notée sct) qui fait passer soit une ressource isolée, soit un ensemble de ressources principales associées, d'un type $X \in \mathcal{T}$ à un autre type $Y \in \mathcal{T}$. Cette opération est nécessaire pour satisfaire les conditions 1 et 3.

Une opération de changement de type a une date de début, une durée et nécessite éventuellement des ressources complémentaires spécifiques déterminées. Toute opération de changement de type est associée à un mode étendu de changement de type τ à sélectionner parmi τ_{sct} modes étendus possibles. Un mode étendu de changement de type τ est caractérisé par :

- un ensemble $\mathcal{P}b_{sct}^{\tau}$ de Mb_{sct}^{τ} pools de ressources principales. Si un seul pool $Pb_{sct}^{\tau 1}$ est présent dans $\mathcal{P}b_{sct}^{\tau}$, le mode τ concerne les ressources isolées contenues dans le pool $Pb_{C}^{\tau,1}$. Sinon, le mode τ regroupe toutes les opérations de changement de type relatives à toutes les associations de ressources principales obtenues en sélectionnant une ressource principale dans chaque pool $Pb_{sct}^{\tau r} \in \mathcal{P}b_{sct}^{\tau}$, $r = 1, ..., Mb_{sct}^{\tau}$. L'ensemble des associations concernées par τ est noté $\mathcal{A}^{\tau}$.

- un ensemble $\mathcal{P}c^{\tau}_{sct}$ de Mc^{τ}_{sct} pools de ressources complémentaires de changement de type. Pour réaliser le changement de type d'une association $a \in \mathcal{A}^{\tau}$, une ressource doit être sélectionnée dans chaque pool $Pc^{\tau r}_{sct} \in \mathcal{P}c^{\tau}_{sct}$, $r = 1, ..., Mc^{\tau}_{sct}$. Le nombre de lignes de charge nécessaires au changement de type d'une association $a \in \mathcal{A}^{\tau}$ sur une ressource complémentaire cumulative k est compris entre $e^{k\tau}_{sct-}$ et $e^{k\tau}_{sct+}$.
- une durée de changement de type $p^{\tau XYge}_{sct}$ dépendant du type précédant X et du type suivant Y et d'une seule des ressources principales ou complémentaires appelée ressource guidante de changement de type g sélectionnée au sein du pool guidant $Pg^{\tau}_{sct} \in \mathcal{P}b^{\tau}_{sct} \cup \mathcal{P}c^{\tau}_{sct}$ et du nombre de lignes de charge e utilisées sur g.

L'ensemble des opérations de changement de type nécessaires est noté $\mathcal{S}ct$.

A une association a de cardinal $|a|$, $a = \{k_1, ..., k_{|a|}\}$, correspond au moins un mode étendu de montage $\mu(a)$, un mode étendu de démontage $\delta(a)$ et un mode étendu de changement de type $\tau(a)$. Par définition, une ressource principale isolée est une association de cardinal 1. A une ressource principale k isolée, correspond au moins un mode étendu de changement de type $\tau(k)$.

L'ensemble des opérations de préparation nécessaires est noté $\mathcal{S} = \mathcal{S}m \cup \mathcal{S}d \cup \mathcal{S}ct$.

La figure II.4 donne un exemple de modes étendus de préparation dont le but n'est pas d'être totalement réaliste mais d'exprimer un ensemble des possibilités offertes par le modèle. On donne un tableau par opération de préparation. On peut trouver un mode étendu par colonnes et un pool par ligne. Une ligne supplémentaires donne les durées de préparation en fonction de la ressource et du nombre de lignes de charge sélectionnés dans le pool guidant du mode étendu considéré. On distingue les pools principaux (P) des pools complémentaires (C). On considère un ensemble de trois presses $PR1$, $PR2$ et $PR3$ sur lesquelles peuvent être montés 3 moules $MO1$, $MO2$ et $MO3$. 3 types d'opérations d'exécution sont considérés $\mathcal{T} = \{ROUGE, VERT, BLEU\}$. Les modes étendus de montage sont au nombre de 2. Le mode étendu 1 concerne les six associations obtenues en combinant les ensembles $\{PR1, PR2, PR3\}$ et $\{MO1, MO2\}$. La ressource complémentaire de montage doit être sélectionnée au sein du pool $\{OA, OB\}$ contenant deux ressources cumulatives à deux lignes de charge (par exemple, 2 équipes A et B de monteurs composées chacune de 2 monteurs). Ce pool étant guidant, le temps de montage dépend de la ressource sélectionnée (équipe de monteurs A ou B) et du nombre de lignes de charge (1 ou 2 monteurs). Le mode étendu 2 concerne les 3 associations obtenues en montant le moule $MO3$ sur chacune des presses. Seule l'équipe de monteurs B peut effectuer ce montage dont la durée dépend du nombre de monteurs choisi. Un seul mode étendu de démontage est défini. Il concerne toutes les associations possibles presse/moule. Le pool guidant de démontage est le pool contenant les presses. Aussi, le temps de démontage dépend uniquement de la presse sélectionnée. Le pool de ressources complémentaires est le même que celui du mode étendu 1 de montage, mais il n'a pas d'influence sur la durée. Aussi, le nombre de lignes de charge requis est fixé à 1 pour la ressource OA, et à 2 pour la ressource OB. Deux modes étendus de changement de type sont définis. Le mode étendu 1 concerne toutes les associations presse/moule. Le pool guidant contient une ressource cumulative OC à 2 lignes de charge. Deux matrices de changement de type sont définies selon qu'on sélectionne un ou deux opérateur(s). Le mode étendu 2 concerne toutes les associations de cardinal 1, c'est-à-dire les ressources isolées. Le pool guidant est encore $\{OC\}$ mais seule une ligne de charge est requise. Aussi, la durée est-elle indépendante de la ressource choisie. Une seule matrice de changement de type est définie.

Modes étendus de montage			
$\mu = 1$	$\mu = 2$	P/C	
$Pc_{sm}^{11} = \{OA, OB\}$	$Pc_{sm}^{21} = \{OB\}$	C	Pool guidant Pg_{sm}^{μ}
$p_{sm}^{1\{OA\}1} = 4 \quad p_{sm}^{1\{OA\}2} = 2$ $p_{sm}^{1\{OB\}1} = 5 \quad p_{sm}^{1\{OB\}2} = 3$	$p_{sm}^{2\{OB\}1} = 5 \; p_{sm}^{2\{OB\}2} = 3$		Durée de montage
$Pb_{sm}^{11} = \{PR1, PR2, PR3\}$ $Pb_{sm}^{12} = \{MO1, MO2\}$	$Pb_{sm}^{21} = \{PR1, PR2, PR3\}$ $Pb_{sm}^{22} = \{MO3\}$	P P	Pools annexes

Mode étendu de démontage		
$\delta = 1$	P/C	
$Pb_{sd}^{11} = \{PR1, PR2, PR3\}$	P	Pool guidant Pg_{sd}^{δ}
$p_{sd}^{1\{PR1\}1} = 5 \; p_{sd}^{1\{PR2\}1} = 4 \; p_{sd}^{1\{PR3\}1} = 3$		Durée de démontage
$Pb_{sd}^{12} = \{MO1, MO2, MO3\}$ $Pc_{sd}^{11} = \{OA(1), OB(2)\}$	P C	Pools annexes

Modes étendus de changement de type				
$\tau = 1$	P/C	$\tau = 2$	P/C	
$Pc_{sct}^{11} = \{OC\}$	C	$Pc_{sct}^{21} = \{OC\}$	C	Pool guidant
$p_{sct}^{1XY\{OC\}1} = \begin{vmatrix} 0 & 4 & 6 \\ 8 & 0 & 8 \\ 10 & 12 & 0 \end{vmatrix}$ $p_{sct}^{1XY\{OC\}2} = \begin{vmatrix} 0 & 2 & 3 \\ 4 & 0 & 4 \\ 5 & 6 & 0 \end{vmatrix}$		$p_{sct}^{2XY\{OC\}1} = \begin{vmatrix} 0 & 3 & 4 \\ 5 & 0 & 5 \\ 6 & 7 & 0 \end{vmatrix}$		durée de CT
$Pb_{sct}^{11} = \{PR1, PR2, PR3\}$ $Pb_{sct}^{12} = \{MO1, MO2, MO3\}$	P P	$Pb_{sct}^{21} = \{PR1, PR2, PR3$ $, MO1, MO2, MO3\}$	P	Pools annexes

FIG. II.4 – *Exemple de modes étendus de préparation*

II.2.2.d) Contraintes de succession des opérations de préparation et d'exécution sur les ressources principales

Considérons qu'une opération d'exécution (i, j) est affectée à un ensemble de ressources représenté par une occupation des ressources l_{ij}. Cette occupation des ressources définit une unique association $\mathcal{R}b_{ij}$ de ressources principales requises par (i, j). Sur chaque ressource principale disjonctive $k \in \mathcal{R}b_{ij}$, (i, j) est précédée par une opération d'exécution $p(k)$.

L'activité de préparation nécessaire avant l'exécution de (i, j) est décomposée en *opé-*

rations de préparation conformément à une des deux configurations suivantes:

1. Si la même opération d'exécution p précède (i,j) sur chaque ressource principale $k \in \mathcal{R}b_{ij}$ et si $\mathcal{R}b_p = \mathcal{R}b_{ij}$, alors l'association $\mathcal{R}b_{ij}$ est déjà constituée. La condition 1 est vérifiée. Si p et (i,j) sont de types différents, une **opération de changement de type** sct_{ij} est nécessaire sur l'ensemble des ressources principales $\mathcal{R}b_{ij}$ entre p et (i,j) pour le respect de la condition 2. Soit $\tau(\mathcal{R}b_{ij})$ un mode de changement de type possible pour l'association $\mathcal{R}b_{ij}$. sct_{ij} est caractérisée par :

   – l'ensemble de ressources principales dont le type est changé ($\mathcal{R}b_{ij}$),

   – l'ensemble de pools de ressources complémentaires $\mathcal{P}c_{sct}^{\tau(\mathcal{R}b_{ij})}$ déterminé par $\tau(\mathcal{R}b_{ij})$. Une ressource et un nombre de lignes de charge doivent être sélectionnés dans chaque pool complémentaire.

   – une durée $p_{sct_{ij}} = p_{sct}^{\tau(\mathcal{R}b_{ij})RT_pRT_{ij}ge}$ déterminée par le mode $\tau(\mathcal{R}b_{ij})$, le type précédent RT_p, le type suivant RT_{ij}, la ressource guidante g de changement de type sélectionnée et le nombre de lignes de charge e. Si le pool guidant est un pool principal, la durée est déterminée par $\mathcal{R}b_{ij}$ car $g \in \mathcal{R}b_{ij}$ et $e = 1$. Sinon, la durée dépend de l'affectation sur les ressources complémentaires.

   – une date de début $t_{sct_{ij}}$ contrainte à la fois par la fin de l'opération p :

   $$t_{sct_{ij}} \geq t_p + p_p \tag{II.3}$$

   et par la disponibilité des ressources complémentaires sélectionnées.

   On a alors pour (i,j) la contrainte suivante:

   $$t_{ij} \geq t_{sct_{ij}} + p_{sct_{ij}} \tag{II.4}$$

   Si p et (i,j) sont de même type, aucune opération de préparation n'est nécessaire entre les deux opérations d'exécution et on a simplement :

   $$t_{ij} \geq t_p + p_p \tag{II.5}$$

2. S'il existe une opération d'exécution précédente $p(k)$ telle que $\mathcal{R}b_{p(k)} \neq \mathcal{R}b_{ij}$, la condition 1 n'est pas vérifiée. Une **opération de démontage** $sd_{p(k)}$ est nécessaire sur chaque ensemble de ressources principales associées $\mathcal{R}b_{p(k)}$ après $p(k)$ pour faire passer la ressource k à l'état "isolée". C'est la première étape pour vérifier la condition 1. Soit $\delta(\mathcal{R}b_{p(k)})$ un mode de démontage correspondant à l'association $\mathcal{R}b_{p(k)}$. Pour chaque ressource principale $k \in \mathcal{R}b_{ij}$, $sd_{p(k)}$ est caractérisée par :

   – l'ensemble de ressources principales à démonter ($\mathcal{R}b_{p(k)}$),

   – l'ensemble de pools de ressources complémentaires $\mathcal{P}c_{sd}^{\delta(\mathcal{R}b_{p(k)})}$ déterminé par $\delta(\mathcal{R}b_{p(k)})$. Une ressource et un nombre de lignes de charge doivent être sélectionnés dans chaque pool complémentaire.

- une durée $p_{sd_{p(k)}} = p_{sd}^{\delta(\mathcal{R}b_{p(k)})ge}$ déterminée par le mode $\delta(\mathcal{R}b_{p(k)})$, la ressource guidante g de démontage sélectionnée et le nombre de lignes de charge sélectionné e. Si le pool guidant est un pool principal, la durée est déterminée par $\mathcal{R}b_{p(k)}$ car $g \in \mathcal{R}b_{p(k)}$ et $e = 1$. Sinon, la durée dépend de l'affectation sur les ressources complémentaires.
- une date de début $t_{sd_{p(k)}}$ contrainte à la fois par la fin de l'opération $p(k)$:

$$t_{sd_{p(k)}} \geq t_{p(k)} + p_{p(k)} \tag{II.6}$$

et par la disponibilité des ressources complémentaires sélectionnées.

Puis, sur chaque ressource isolée $k \in \mathcal{R}b_{ij}$ si l'opération $p(k)$ n'est pas du même type que (i,j) ($RT_{p(k)} \neq RT_{ij}$) une **opération de changement de type** sct_{ij}^k est nécessaire. En effet, la condition 3 impose que des ressources principales de types différents ne peuvent être associées et la condition 2 impose de donner à ces ressources le type de (i,j). Soit $\tau(k)$ le mode de changement de type de la ressource isolée k. sct_{ij}^k est caractérisée par :

- la ressource principale k dont le type est à changer,
- un ensemble de pools de ressources complémentaires $\mathcal{P}c_C^{\tau_k}$ déterminé par τ_k. Une ressource et un nombre de lignes de charge doivent être sélectionnés dans chaque pool complémentaire.
- une durée $p_{sct_{ij}^k} = p_{sct}^{\tau(k)RT_{p(k)}RT_{ij}ge}$ déterminée par le mode $\tau(k)$, le type précédent $RT_{p(k)}$, le type suivant RT_{ij}, la ressource guidante g de changement de type sélectionnée et le nombre de lignes de charge choisi e. Si le pool guidant est un pool principal, la durée est déterminée par k car $g = k$ et $e = 1$. Sinon, la durée dépend de l'affectation sur les ressources complémentaires.
- une date de début $t_{sct_{ij}^k}$ contrainte par la fin de l'opération $sd_{p(k)}$:

$$t_{sct_{ij}^k} \geq t_{sd_{p(k)}} + p_{sd_{p(k)}} \tag{II.7}$$

et par la disponibilité des ressources complémentaires sélectionnées.

Si (i,j) et $p(k)$ sont du même type, l'opération sct_{ij}^k n'est pas nécessaire.

Enfin, une fois les changements de type éventuels effectués, une **opération de montage** sm_{ij} est réalisée sur l'ensemble de ressources principales $\mathcal{R}b_{ij}$. C'est la deuxième étape pour vérifier la condition 2 (faire passer chaque ressource principale de l'état "isolée" à l"etat associée"). Soit $\mu(\mathcal{R}b_{ij})$ le mode de préparation correspondant à l'association $\mathcal{R}b_{ij}$. sm_{ij} est caractérisée par :

- l'ensemble de ressources principales à monter ($\mathcal{R}b_{ij}$),
- un ensemble de pools de ressources complémentaires $\mathcal{P}c_{sm}^{\mu(\mathcal{R}b_{ij})}$ déterminé par $\mu(\mathcal{R}b_{ij})$. Une ressource et un nombre de lignes de charge doivent être sélectionnés dans chaque pool complémentaire.

- une durée $p_{sm_{ij}} = p_{sm}^{\mu(\mathcal{R}b_{ij})ge}$ déterminée par le mode $\mu(\mathcal{R}b_{ij})$, la ressource guidante g de changement de type sélectionnée et le nombre de lignes de charge choisi e. Si le pool guidant est un pool principal, la durée est déterminée par $\mathcal{R}b_{ij}$ car $g \in \mathcal{R}b_{ij}$ et $e = 1$. Sinon, la durée dépend de l'affectation sur les ressources complémentaires.
- une date de début $t_{sm_{ij}}$ contrainte sur chaque ressource $k \in \mathcal{R}b_{ij}$ par la fin de l'opération sct_{ij}^k si elle est nécessaire :

$$t_{sm_{ij}} \geq t_{sct_{ij}^k} + p_{sct_{ij}^k} \tag{II.8}$$

  où par la fin de $sd_{p(k)}$ si sct_{ij}^k n'est pas nécessaire :

$$t_{sm_{ij}} \geq t_{sd_{p(k)}} + p_{sd_{p(k)}} \tag{II.9}$$

  et par la disponibilité des ressources complémentaires sélectionnées.

Finalement, on a pour la date de début de (i, j) :

$$t_{ij} \geq t_{sm_{ij}} + p_{sm_{ij}} \tag{II.10}$$

Etats initiaux des ressources principales On suppose que chaque ressource principale k possède un état initial caractérisé par :

- un type initial $RT_k^{init} \in \mathcal{T}$,
- une association initiale $\mathcal{R}b_k^{init}$. Si $\mathcal{R}b_k^{init} = \emptyset$, alors la ressource est à l'état initial "isolée". Sinon, la ressource est à l'état initial "associée" et toute opération (i, j) affectée de telle sorte que $\mathcal{R}b_{ij} = \mathcal{R}b_k^{init}$ pourra utiliser cette association sans générer d'opération de démontage de $\mathcal{R}b_k^{init}$ ni de montage de $\mathcal{R}b_{ij}$ (configuration 1).

II.2.2.e) Contraintes de succession des opérations de préparation et d'exécution sur les ressources complémentaires

Chaque ligne de charge l d'une ressource complémentaire k est assimilée à une ressource disjonctive. Soit une opération o d'exécution ou de préparation affectée à $k \in \mathcal{R}c$. Soit une opération $p(k, l)$ d'exécution ou de préparation précédant o sur la ligne de charge l de la ressource k. La contrainte de succession entre o et $p(k, l)$ s'exprime par :

$$t_o \geq t_{p(k,l)} + p_{p(k,l)} \tag{II.11}$$

II.2.2.f) Exemple

Un exemple de la configuration 1 est donné figure II.5 en reprenant l'exemple de la figure II.4. L'opération $(2, 1)$ de durée 2 et l'opération $(1, 1)$ de durée 3 utilisent les mêmes ressources principales ($\mathcal{R}b_{21} = \mathcal{R}b_{11} = \{PR1, MO1\}$) mais sont d'un type différent ($RT_{21} = VERT$ et $RT_{11} = ROUGE$). On suppose ici que $RT_{PR1}^{init} = RT_{MO1}^{init} = ROUGE$ et que $\mathcal{R}b_{PR1}^{init} = \mathcal{R}b_{MO1}^{init} = \{PR1, MO1\}$, ce qui permet à l'opération $(1, 1)$ d'utiliser l'association $\{PR1, MO1\}$ sans générer d'opération de préparation. Une opération de changement de type sct_{21} est nécessaire sur l'association $\{PR1, MO1\}$ pour passer de l'opération $(1,1)$ à l'opération $(2,1)$. Cette association correspond au mode étendu de changement de type $\tau = 1$ de la figure II.4. Le changement de type peut être exécuté sur une ou deux lignes de charge de la ressource OC. Une opération $(3,1)$ de durée 3 est ordonnancée sur la ligne de charge 1 de OC. Une opération $(4,1)$ de durée 6 est ordonnancée sur la ligne de charge 1 de OC. Si on choisit $e^{OC}_{sct_{21}} = 2$, on obtient :

$$p_{sct_{21}} = p_{sct}^{1\{ROUGE\}\{VERT\}\{OC\}1} = 2$$

$(4,1)$ et $(3,1)$ précédent alors obligatoirement sct_{21} sur OC, on a d'après les contraintes II.11 : $t_{sct_{21}} \geq 3$ et $t_{sct_{21}} \geq 6$. D'après la contrainte II.3, on a : $t_{sct_{21}} \geq 2$. D'où $t_{sct_{21}} = 6$. D'après la contrainte, II.4, on a $t_{21} = 8$. Si on choisit $e^{OC}_{sct_{21}} = 1$, on a :

$$p_{sct_{21}} = p_{sct}^{1\{ROUGE\}\{VERT\}\{OC\}1} = 4$$

En affectant $(2,1)$ à la ligne de charge 1 de OC, seule $(3,1)$ précède sct_{21} sur OC et la contrainte II.11 donne uniquement $t_{sct_{21}} \geq 3$. D'où $t_{sct_{21}} = 3$. D'après II.4, on obtient : $t_{21} = 7$. Ici l'utilisation d'une seule ligne de charge au lieu de 2 sur OC augmente le temps de préparation mais permet de commencer $(2,1)$ plus tôt.

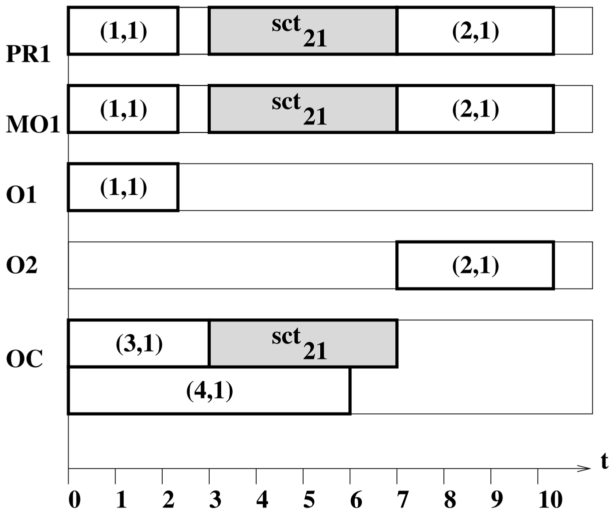


FIG. II.5 – *Exemple de changement de type sur ressources associées*

En conservant toujours l'exemple de la figure II.4, la figure II.6 montre un exemple de la configuration 2. Les états initiaux des ressources sont tels que $RT^{init}_{PR1} = RT^{init}_{MO1} = ROUGE$, $\mathcal{R}b_{PR1} = \mathcal{R}b_{MO1} = \{PR1, MO1\}$, $RT^{init}_{PR2} = RT^{init}_{MO2} = BLEU$ et $\mathcal{R}b_{PR2} = \mathcal{R}b_{MO2} = \{PR2, MO2\}$. Les associations $\{PR1, MO1\}$ et $\{PR2, MO2\}$ sont initialement constituées.

L'opération $(3,1)$ de type $VERT$ est affectée aux ressources principales $PR1$ et $MO2$ et à la ressource complémentaire $O3$. Sur $PR1$, $(3,1)$ a pour précédente $(1,1)$ de durée $p_{11} = 2$ et de type $ROUGE$ qui utilise l'association $\{PR1, MO1\}$ et la ressource complémentaire $O1$. On pose $t_{11} = 0$, ce que permet l'état initial des ressources $PR1$ et

$MO1$. Sur $MO2$, $(3,1)$ a pour précédente $(2,1)$ de durée $p_{21} = 3$de type $BLEU$ qui utilise l'association $\{PR2, MO2\}$ et la ressource complémentaire $O2$. On pose $t_{21} = 0$, ce que permet l'état initial des ressources $PR2$ et $MO2$. Les associations $\{PR1, MO1\}$ et $\{PR2, MO2\}$, toutes deux correspondant au mode 1 de démontage doivent d'abord être démontées. Une opération de démontage sd_{11} est ainsi ordonnancée sur les ressources principales $PR1$, $MO1$ après $(1,1)$ et sur une ligne de charge de la ressource complémentaire de démontage OA après une opération $(4,1)$ telle que $t_{41} = 0$ et $p_{41} = 3$. La contrainte II.6 impose que $t_{sd_{11}} \geq 2$. La contrainte II.11 impose que $t_{sd_{11}} \geq 3$. D'où $t_{sd_{11}} = 3$. Cette affectation correspond au mode étendu de démontage $\beta = 1$. La durée est fixée dans ce mode étendue par la presse démontée. On a ainsi

$$p_{sd_{11}} = p_{sd}^{1\{PR1\}1} = 5$$

. Une opération de démontage sd_{21} est ainsi ordonnancée sur les ressources $PR2$, $MO2$ après $(2,1)$ et sur une ligne de charge de la ressource complémentaire de démontage OA après $(4,1)$. On trouve par l'application des mêmes contraintes $t_{sd_{21}} = 3$ et

$$p_{sd_{21}} = p_{sd}^{1\{PR2\}1} = 4$$

On doit ensuite changer le type des ressources isolées $PR1$ et $MO2$. Les ressources isolées correspondent toutes les deux au mode étendu de changement de type $\tau = 2$. Une opération de changement de type sct_{31}^2 est ainsi ordonnancée sur $MO2$ après sd_{11} et une ligne de charge de la ressource complémentaire OC après une opération $(5,1)$ telle que $t_{51} = 0$ et $p_{51} = 7$. La contrainte II.7 impose que $t_{sct_{31}^2} \geq 7$. La contrainte II.11 impose que $t_{sct_{31}^2} \geq 7$. D'où $t_{sct_{31}^2} = 7$. Dans le mode étendu $\tau = 2$, aucune possibilité d'affectation n'est donnée et on a :

$$p_{sct_{31}^2} = p_{sct}^{2\{BLEU\}\{VERT\}\{OC\}1} = 7$$

Une opération de changement de type sct_{31}^1 est ordonnancée sur $PR1$ après sd_{11} et une ligne de charge de OC après une opération $(6,1)$ telle que $t_{61} = 0$ et $p_{61} = 9$. La contrainte II.7 impose que $t_{sct_{31}^1} \geq 8$. La contrainte II.11 impose que $t_{sct_{31}^2} \geq 9$. D'où $t_{sct_{31}^2} = 9$. Dans le mode étendu $\tau = 2$, on a :

$$p_{sct_{31}^2} = p_{sct}^{2\{ROUGE\}\{VERT\}\{OC\}1} = 3$$

Enfin l'association $\{PR1, MO2\}$ correspondant au mode 1, une opération de montage est ordonnancée sur $PR1$ après sct_{31}^1, sur $MO2$ après sct_{31}^2 et sur deux lignes de charge de la ressource complémentaire OA après sd_{11} et sd_{21}. La contrainte II.8 sur la ressource $PR1$ impose que $t_{sm_{31}} \geq 12$. La contrainte II.8 sur la ressource $MO2$ impose que $t_{sm_{31}} \geq 14$. La contrainte II.11 sur la ligne de charge 1 de OA impose que $t_{sm_{31}} \geq 8$. La contrainte II.11 sur la ligne de charge 1 de OA impose que $t_{sm_{31}} \geq 7$. D'où $t_{sm_{31}} = 14$. Dans le mode étendu de montage $\mu = 1$, on a

$$p_{sm_{31}} = p_{sm}^{1\{OA\}2} = 4$$

D'après la contrainte II.10, on a $t_{31} \geq 16$. D'après la contrainte II.11 sur la ressource $O3$, on a $t_{31} \geq 0$. D'où $t_{31} = 16$. Comme on peut le constater sur cet exemple deux difficultés sont liées à l'utilisation de ressources complémentaires de préparation :

- le décalage des dates de début de certaines opérations à cause de l'indisponibilité des ressources complémentaires (cas de l'opération sct^2_{31}). Un des rôles de la procédure d'ordonnancement est d'affecter judicieusement les opérations de préparation et d'exécution sur leurs ressources complémentaires.

- l'existence de conflits entre deux opérations de préparation nécessitant la même ressource complémentaire (ici sct^1_{31} et sct^2_{31} sur OC, ainsi que sd_{11} et sd_{21} sur OA). Un des rôles de la procédure d'ordonnancement est de résoudre ces conflits.

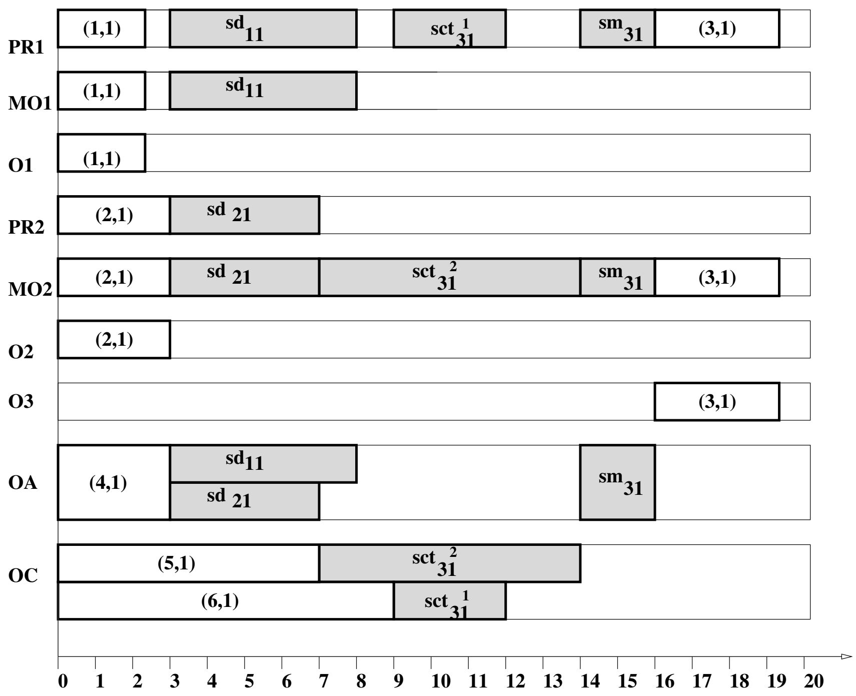


FIG. II.6 – *Exemple de montage, démontage et changement de type*

II.2.3 Expression des modèles classiques par le modèle proposé

II.2.3.a) Modèles de préparation

On montre dans ce paragraphe que le modèle proposé est très général car il permet d'exprimer la plupart des modèles de temps de préparation rencontrés dans la littérature et mentionnés en II.1.1.

Modèle du temps de montage et démontage indépendant de la séquence S_{nsd}, R_{nsd}
Dans ce modèle disjonctif mono-ressource, on considère qu'un temps de montage est nécessaire avant chaque opération, de même qu'un temps de démontage après chaque opération. Pour prendre en compte ces temps dans le modèle multi-ressources proposé, il suffit

d'ajouter pour chaque opération (i, j) une ressource principale fictive R_{ij} et de donner les temps de montage et de démontage de R_{ij} sur toute ressource (réelle) utilisable par (i, j).

Modèle du temps de préparation dépendant de la séquence s_{sd} On obtient ce modèle en associant un type différent à chaque opération, et en définissant pour chaque ressource une matrice de temps de changement de type.

Modèle du temps de préparation indépendant de la séquence par familles $s_{nsd}, batch$ On obtient ce modèle en définissant un type par famille, et en définissant les matrices de changement de type de telle sorte que les temps de changement ne dépendent que du type de l'opération d'exécution qui suit l'opération de changement de type.

Modèle des temps de préparations majeurs et mineurs s_{nsd}, maj, min En associant une ressource fictive supplémentaire à chaque opération et en définissant pour cette ressource un temps de démontage nul et un temps de montage non nul, on prend en compte le temps de préparation mineur. Le temps de préparation majeur est considéré de la même façon que pour le modèle $s_{nsd}, batch$.

Prise en compte des magasins d'outils Le modèle proposé peut par l'intermédiaire du montage et du démontage considérer les temps de changements d'outils entre deux opérations. La prise en compte de magasins d'outils pouvant contenir un ensemble d'outils utilisables par plusieurs opérations modifie légèrement la définition du montage et du démontage. En effet on peut définir le "remplissage" du magasin comme une opération de montage de tous les outils concernés sur la machine. Toute opération d'exécution nécessitant un ensemble d'outils tous présents dans le magasin (tous associés) peut être exécutée sur l'association, même si d'autres outils non nécessaires font partie de l'association. Lorsqu'une opération nécessite un outil non associé, une opération de démontage de l'ensemble des outils présents et une opération de montage d'un autre ensemble d'outils contenant le(s) outil(s) requis sont nécessaires. La seule différence est que le nombre de ressources associées pour chaque opération de démontage est fixé par la capacité du magasin et égale au nombre de ressources associables M^{μ}_{sm} et M^{δ}_{sd} des modes de montage et de démontage associés à chaque machine. On associe ainsi plus de ressources que n'en nécessite l'opération qui suit le montage et l'anticipation nécessaire pour définir les outils à mettre dans le magasin (l'ensemble des ressources à associer) introduit un problème de décision supplémentaire. Dans la suite du document, ce problème n'est pas explicitement géré.

II.2.3.b) Modèles d'atelier

Notre modèle est une extension du modèle d'atelier à cheminements multiples et peut être aisément réduit à un problème d'atelier à cheminement unique (classique ou hybride), à machines parallèles et à machine unique.

De plus, la gestion du multi-ressources permet de se ramener à un problème d'atelier à cheminement quelconque (*open shop*) en définissant pour chaque ordre de fabrication i une gamme sans contraintes de précédence ($U_i = \emptyset$) et en introduisant une ressource disjonctive fictive commune aux opérations du même ordre de fabrication (voir [Schutten, 1996],[Dauzère Péres et al, 1996]). La structure non-linéaire des gammes, la notion de modes étendus et la prise en compte des contraintes cumulatives permettent de formuler notre modèle comme un modèle d'ordonnancement multi-projets à moyens limités et modes multiples. Les ressources de transports entre opérations de la même gamme peuvent être gérées. Par contre, le modèle ne prend en compte ni les ressources non renouvelables, ni les zones de stockage à capacité limitée (*block*) ni l'impossibilité d'attente $no - wait$), ni la préemption ($pmtn$).

Dans un premier temps, le chevauchement, les temps de transport, et les calendriers de fonctionnement des ressources ne sont pas pris en compte, mais une extension à ces caractéristiques importantes pour les ateliers réels est évoquée dans le paragraphe V.5.

II.3 Caractérisation d'un ensemble d'ordonnancements avec temps de préparation

II.3.1 Extension du concept de groupe d'opérations permutables

Pour adapter le concept de groupes d'opérations permutables défini dans le paragraphe I.2.1.a) à la prise en compte des activités de préparation, la notion de *classe de préparation* est introduite.

Définition 1 *Deux opérations d'exécution sont de même classe de préparation si elles sont du même type et si elles sont affectées aux mêmes ressources principales.*

Par suite :

Définition 2 *Un groupe d'opérations d'exécution permutables est composé d'opérations d'exécution :*

- *de même classe de préparation,*
- *utilisant les mêmes ressources complémentaires et sur chaque ressource complémentaire cumulative, les mêmes lignes de charge,*
- *telles que la sélection de n'importe quelle séquence d'opérations d'exécution au sein du groupe respecte toutes les contraintes y compris les dates de fin au plus tard des ordres de fabrication*

Le groupe contenant l'opération d'exécution (i, j) est noté G_{ij}. Par suite, une activité de préparation ne peut exister qu'entre deux groupes d'opérations d'exécution. Les deux premières conditions de permutabilité peuvent sembler restrictives. De façon à proposer un nombre de solutions conséquent, il faut ainsi veiller :

- à éviter la modélisation de ressources complémentaires non critiques et de temps de préparation négligeables (qu'il convient d'inclure dans le temps d'exécution),

- à favoriser le regroupement d'opérations de même classe (voir V.3.3.a)) et utilisant les mêmes ressources complémentaires (voir V.3.3.b)), lors de la phase de génération de la séquence,
- à tester la permutabilité dans le cas plus général correspondant à la relaxation des deux premières conditions lors de la phase de pilotage en temps réel (voir V.4.5.c) ii2).

II.3.2 Impact des groupes sur les contraintes du problème

La notion de groupe d'opérations d'exécution modifie certaines des contraintes définies dans le paragraphe II.2.2.

II.3.2.a) Impact sur les opérations de préparation

On suppose, par souci d'homogénéité, que toute opération de préparation constitue un groupe à une seule opération. On note G_s le groupe d'une opération de préparation $s \in \mathcal{S}$. De plus l'activité de préparation ordonnancée sur une ressource avant un groupe d'opérations d'exécution donné G concerne toutes les opérations d'exécution constituant ce groupe. Toutefois, pour respecter la notation du paragraphe II.2.2, on peut continuer à noter les opérations de préparation (montage, démontage et changement de type) constituant cette activité s_{ij} où (i,j) est une opération quelconque du groupe.

II.3.2.b) Contraintes de succession sur les ressources

Lorsqu'une opération o (d'exécution ou de préparation) succède à une opération d'exécution (i,j) appartenant à un groupe G_p, o n'appartenant pas à G_p, la date de début de o est contrainte par l'ensemble des opérations du groupe. On considère pour l'expression de cette contrainte la permutation la plus défavorable, c'est-à-dire celle obtenue en positionnant en première position l'opération de G_p de plus grande date de début.

$$t_o \geq \max_{(k,l)\in G_p} t_{kl} + \sum_{(k,l)\in G_p} p_{kl} \tag{II.12}$$

De même, lorsqu'une opération o (d'exécution ou de préparation) précède une opération d'exécution (i,j) appartenant à un groupe G_s, o n'appartenant pas à G_s, la date de fin de o est contrainte par l'ensemble des opérations du groupe. On considère pour l'expression de cette contrainte la permutation la plus défavorable, c'est-à-dire celle obtenue en positionnant en dernière position l'opération de G_s de plus petite date de fin.

$$t_o + p_o \leq \min_{(k,l)\in G_s} (t_{kl} + p_{kl}) - \sum_{(k,l)\in G_s} p_{kl} \tag{II.13}$$

II.3.2.c) Contraintes de gamme

Une contrainte sur la date de début d'une opération d'exécution (i,j) est ajoutée. On considère que chaque opération précédente (i,h) dans la gamme de l'ordre de fabrication i est placée en dernière position de la permutation la plus défavorable de son groupe G_{ih}.

Il s'agit de la permutation telle que l'opération de plus grande date de début de G_{ih}, différente de (i,h), est placée en première position.

$$t_{ij} \geq \max_{(i,h) \in prec^i_{ij}} \left(\max_{(k,l) \in G_{ih}, (k,l) \neq (i,j)} t_{kl} + \sum_{(k,l) \in G_{ih}} p_{kl} \right) \qquad \text{(II.14)}$$

De même, une contrainte sur la date de fin d'une opération d'exécution (i,j) est ajoutée. On considère que chaque opération suivante (i,h) dans la gamme de l'ordre de fabrication i est placée en première position de la permutation la plus défavorable de son groupe G_{ih}. Il s'agit de la permutation telle que l'opération de plus petite date de début de G_{ih}, différente de (i,h), est placée en dernière position.

$$t_{ij} + p_{ij} \leq \min_{(i,h) \in succ^i_{ij}} \left\{ \min_{(k,l) \in G_{ih}, (k,l) \neq (i,j)} (t_{kl} + p_{kl}) - \sum_{(k,l) \in G_{ih}} p_{kl} \right\} \qquad \text{(II.15)}$$

II.3.3 Représentation d'une séquence de groupes par deux graphes potentiels-tâches

La définition de séquences de groupes d'opérations permutables offre des degrés de liberté de nature séquentielle. Des degrés de liberté de nature temporelle peuvent être mis en évidence en caractérisant la date de début d'une opération d'exécution $(i,j) \in \mathcal{O}$ (ou d'une opération de préparation $s \in \mathcal{S}$) au moyen d'une date de début au plus tôt r_{ij} (r_s) et d'une date de fin au plus tard d_{ij} (d_s). Une séquence de groupes, c'est-à-dire un ensemble de solutions au problème d'ordonnancement, peut être représentée au moyen de deux graphes potentiels-tâches. Ces graphes permettent en outre de calculer les dates de début au plus tôt et de fin au plus tard des opérations.

II.3.3.a) Graphe utilisé pour le calcul des dates de début au plus tôt

Un graphe $\mathcal{G} = (X, U)$ est défini pour le calcul des dates de début au plus tôt :

- L'ensemble des sommets $X = \{(i,j)/i = 1, ..., n; j = 1, ..., h_{i+1}\} \cup \mathcal{S} \cup \{O, H\}$ tel que : chaque opération d'exécution est un sommet (ensemble $\mathcal{O}$), chaque opération de préparation nécessaire est un sommet (ensemble $\mathcal{S}$). Pour chaque ordre de fabrication i, on ajoute une opération fictive (i, h_{i+1}) suivant la dernière opération de cet ordre de fabrication. On ajoute au graphe un sommet O origine des temps et un sommet H pour mesurer le plus grand retard.
- L'ensemble des arcs U représente l'ensemble des contraintes de potentiel. On distingue trois types d'arcs :
  - L'arc de type gamme correspond aux contraintes de succession des opérations d'exécution de la même gamme.

    Un arc de type gamme valué par $p_{i,m}$ est défini entre chaque couple de sommets (i,m) et (i,j) tel que $i = 1, ..., n$, $2 \leq j \leq h_{i+1}$ et $(i,m) \in prec^i_{ij}$. Un arc de type gamme valué par r_i est défini entre le sommet O et chaque sommet $(i,1)$ supposé unique. Un arc de type gamme valué par $-d_i$ est défini entre chaque sommet $(i, h_i + 1)$ et le sommet H. Le potentiel associé au sommet

(i, h_i+1) étant la date de fin au plus tôt de l'ordre de fabrication i, le potentiel associé au sommet H est le plus grand retard algébrique $L_{\max}$ de l'ensemble des ordres de fabrication.

- L'arc de type groupe correspond à la contrainte II.14 présentée dans le paragraphe II.3.2. Soit $(i, m) \in prec^i_{ij}$. Soit (r, s) une opération d'exécution appartenant au groupe G_{im} mais différente de (i, m). Il existe un arc de type groupe valué par $\sum_{(k,l)\in G_{im}} p_{kl}$ entre chaque opération (r, s) et (i, j).
- L'arc de type ressource entre deux opérations de préparation et/ou d'exécution peut être défini de la façon suivante :

  Définition 3 *il existe un et un seul arc de type ressource entre deux opérations d'exécution et/ou de préparation si et seulement si elles n'appartiennent pas au même groupe et si les groupes auxquels elles appartiennent sont consécutifs sur une ou plusieurs lignes de charge de l'atelier, l'opération O et l'opération H étant supposées affectées à toutes les lignes de charge de l'atelier.*

  L'arc de type ressource issu d'une opération o est valué par la durée du groupe G_o de o : $\sum_{o'\in G_o} p_{o'}$. Si une opération o est dans un groupe situé en première position sur une ligne de charge, un arc de type ressource valué par 0 est défini entre le sommet O et o Si une opération o' est dans un groupe $G_{o'}$ situé en dernière position sur une des ressources qu'elle utilise, un arc de type ressource est défini entre o' et le sommet H valué par $\sum_{o\in G_{o'}} p_o$. Cet arc ne s'impose pas pour le calcul des dates de début au plus tôt mais il sera utilisé pour la procédure d'insertion présentée dans la deuxième partie.

Les dates de début au plus tôt de toutes les opérations peuvent être obtenues par propagation de potentiel dans ce graphe à partir du sommet O auquel on associe un potentiel nul. Le potentiel obtenu sur le sommet H est le plus grand retard algébrique L_{max}.

II.3.3.b) Graphe utilisé pour le calcul des dates de fin au plus tard

Contrairement au PERT classique, on ne peut pas calculer les dates de fin au plus tard par propagation inverse dans le graphe $\mathcal{G}$ car les arcs de type groupe ne correspondent pas à l'expression de la contrainte II.15 sur les dates de fin présentée dans le paragraphe II.3.2. On est ainsi amené à utiliser un autre graphe contenant exactement les mêmes arcs de type gamme et ressource mais inversés et de poids opposé que les arcs correspondant du graphe $\mathcal{G}$. Les arcs de type groupe sont définis de la façon suivante :

Soit $(i, m) \in suiv^i_{ij}$. Soit (r, s) une opération d'exécution appartenant au groupe G_{im} mais différente de (i, m). Il existe un arc de type groupe valué par $-\sum_{(k,l)\in G_{im}} p_{kl}$ entre chaque opération (r, s) et (i, j).

Une autre différence est le poids des arcs de H vers $(i, h_i + 1)$. Un potentiel nul est associé à H. Pour tout ordre de fabrication qui n'est pas en retard, cet arc est valué par d_i. Pour tout ordre de fabrication i en retard, d_i est remplacé par r_{ih_i+1}, ce qui revient à caler cet ordre de fabrication sur sa date de fin au plus tôt. Les dates de fin au plus tard des opérations sont calculées par propagation dans le graphe à partir du sommet H. On calcule ainsi la date de fin au plus tard de chaque opération pour respecter les délais de chaque ordre de fabrication en avance et pour respecter la date de fin au plus tôt des ordres de fabrication en retard.

II.3.3.c) Exemple de graphe potentiels-tâches des opérations

Soit la séquence de groupes représentée dans la figure II.7. Les modes de préparation sont ceux de la figure II.4. On considère 5 ordres de fabrication de date de début au plus tôt 0 et à gamme linéaire. L'ordre de fabrication 1 a une date de livraison de 20 et deux opérations : $(1,1)$ de durée 2, de type $VERT$, affectée aux ressources $PR1(1), MO1(1), O1(1)$ et l'opération $(1,2)$ de type $ROUGE$, de durée 1 et affectée aux ressources $PR1(1), MO2(1), O3(1)$. L'ordre de fabrication 2 a aussi une date de livraison de 20 et deux opérations : $(2,1)$ de durée 1, de type $BLEU$, affectée aux ressources $PR2(1), MO2(1), O1(1)$ et l'opération $(2,2)$ de type $ROUGE$, de durée 2, affectée aux mêmes ressources que $(1,2)$. L'ordre de fabrication 3 a une date de livraison de 5 et une seule opération : $(3,1)$ de durée 2, de type $BLEU$, affectée aux mêmes ressources que $(2,1)$. L'ordre de fabrication 4 a une date de livraison de 10 et deux opérations : $(4,1)$ de durée 1, affectée à deux lignes de charge de la ressource OA, et l'opération $(4,2)$ de durée 4, affectée à une ligne de charge de la ressource OC. L'ordre de fabrication 5 a aussi une date de livraison de 10 et deux opérations : $(5,1)$ de durée 2, affectée aux mêmes lignes de charge que OA et $(5,2)$ de durée 6 affectée à une ligne de charge de OC. Les opérations des ordres de fabrication 4 et 5 n'ont pas de type. En effet, elles sont affectées uniquement a des ressources complémentaires. La séquence de groupes comporte 3 groupes de plus d'une opération : $\{(2,1)(3,1)\}$, $\{(1,2)(2,2)\}$ et $(4,1)(5,1)\}$. 8 ordonnancements sont ainsi représentés.

Les graphes potentiels-tâches associés à cette séquence sont représentés dans la figure II.8. Le potentiel associé à chaque sommet est entre crochets et correspond dans le graphe du haut aux dates de début au plus tôt des opérations et dans le graphe du bas aux dates de fin au plus tard des opérations. Les différences entre les deux graphes concernent les arcs de type groupe issus dans les deux cas d'opérations différentes appartenant au même groupe. Les opérations de préparation sont ordonnancées et affectées dans le respect des contraintes retenues. On observe que le potentiel du sommet H vaut -1, ce qui signifie qu'aucun des 5 ordres de fabrication n'est en retard dans aucun des 8 ordonnancements proposés.

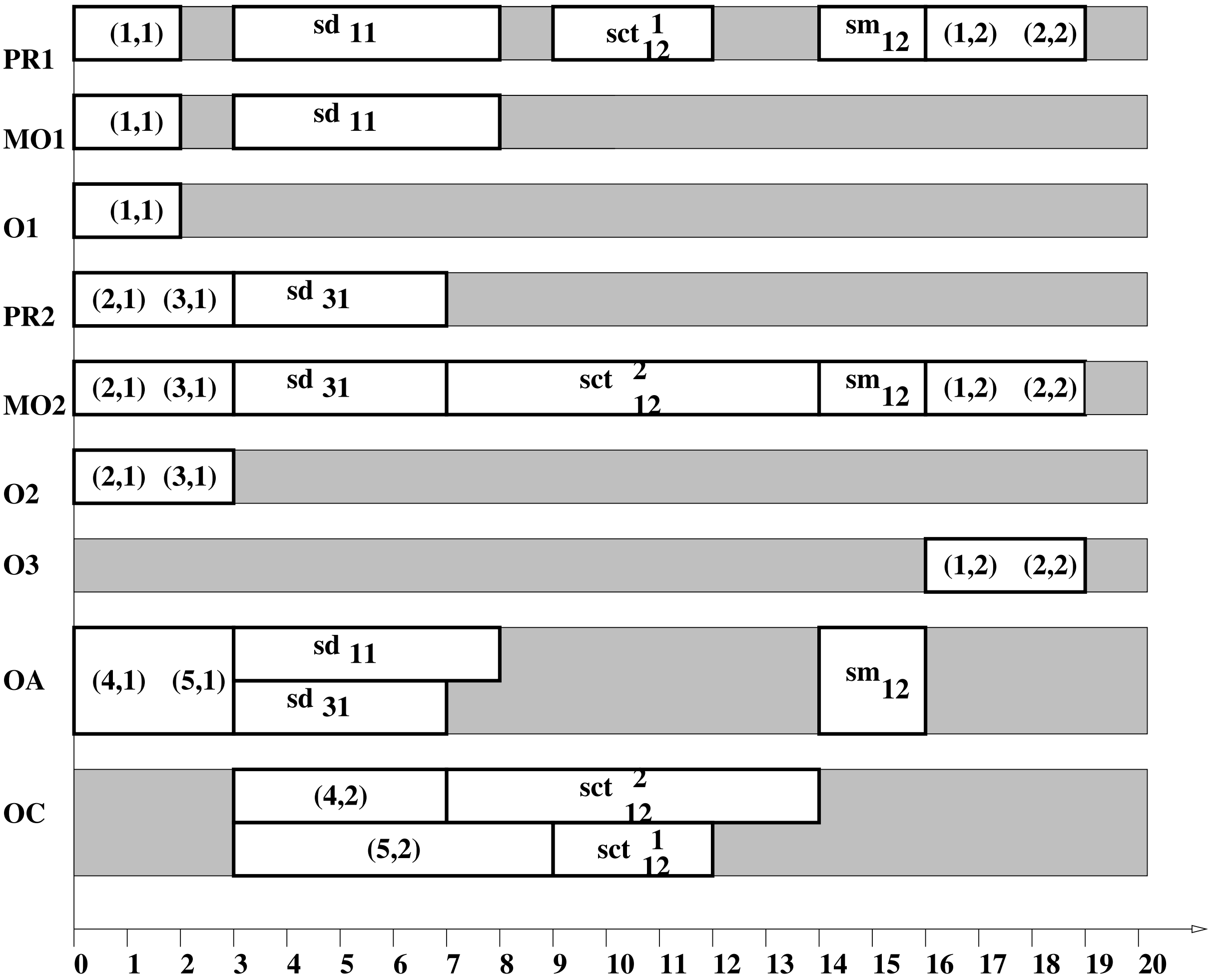


Fig. II.7 – *Exemple de séquence de groupes avec des opérations de préparation*

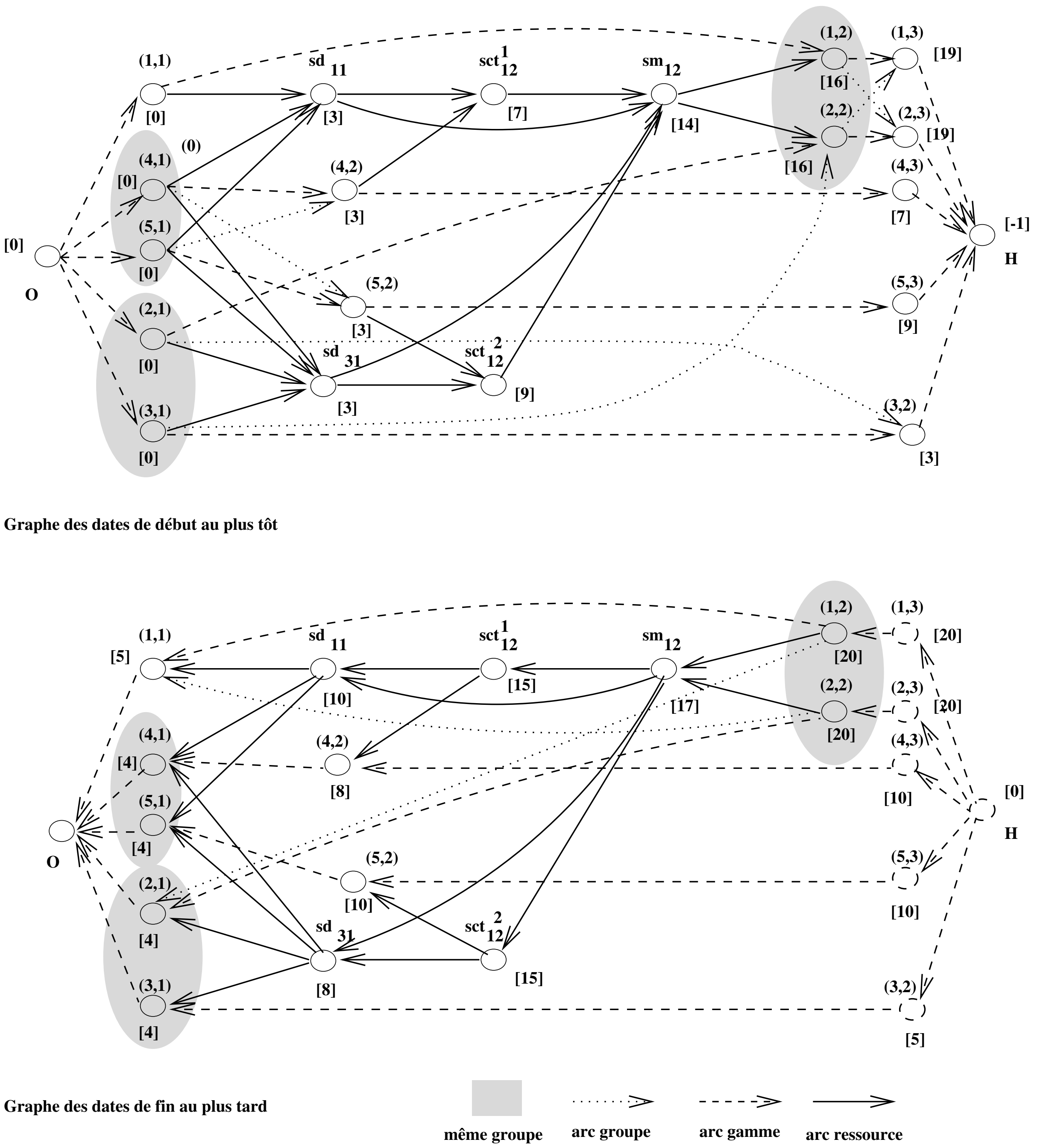


FIG. II.8 – *Graphes potentiels-tâches associés à l'exemple représenté dans la figure II.7*

Deuxième partie

Insertion d'une opération dans un ordonnancement et application à l'amélioration par méthode de voisinage

Chapitre III

Méthodes d'insertion d'une opération dans un ordonnancement

Résumé

Une approche pour la résolution du problème d'insertion d'une opération d'exécution dans un ordonnancement est proposée. L'objectif est la minimisation de l'impact de l'insertion sur l'admissibilité. Un algorithme polynomial et optimal basé sur des règles de dominance est présenté dans le cas cumulatif multi-ressources sans préparation. L'algorithme optimal est étendu successivement aux problèmes comportant des possibilités d'affectation, des opérations de préparation sans ressource complémentaire et enfin la combinaison de ces deux caractéristiques. Une heuristique pour l'insertion est présentée dans le cas où les opérations d'exécution et de préparation ont des ressources complémentaires.

III.1 Définition et intérêt du problème d'insertion

On peut définir le problème d'insertion d'une opération dans un ordonnancement représenté par une séquence d'opérations de la façon suivante. Etant donné :

- un problème d'ordonnancement A défini par un ensemble de ressources, d'ordres de fabrication et de contraintes,
- une séquence d'opérations S_A admissible pour le problème A,
- une opération (i, j) non incluse dans le problème A,
- un ensemble de contraintes d'utilisation des ressources de A par (i, j),

- un ensemble de contraintes de gamme et/ou temporelles associées à (i,j),

le problème d'insertion de (i,j) revient à proposer une nouvelle séquence d'opérations admissible ou *la plus proche possible* de l'admissibilité pour le problème B défini par le problème A auquel on a ajouté l'opération (i,j) et ses contraintes.

Pour traiter ce problème, une approche consiste à résoudre le nouveau problème d'ordonnancement B sans tenir compte de la séquence S_A. A l'opposé, la méthode proposée ici utilise cette séquence comme support pour prendre en compte l'opération (i,j). C'est pourquoi on se fixe comme objectif d'insérer (i,j) dans l'ordonnancement en respectant les conditions suivantes, en plus des contraintes décrites ci-dessus :

- pas de changement de la séquence ni de l'affectation des opérations d'exécution déjà ordonnancées,
- violation minimale des dates de fin au plus tard de (i,j) et des opérations déjà ordonnancées.

La première condition vise d'une part à ne pas provoquer des changements trop importants dans la séquence associée à l'ordonnancement, ce qui pourrait être mal accepté dans l'atelier. D'autre part elle rend le problème B plus simple à résoudre en réduisant l'ensemble des solutions possibles aux ordonnancements respectant la séquence S_A. Compte tenu de cette restriction, la deuxième condition assure que l'impact de l'insertion sur la tenue des délais est minimal.

Une séquence d'opérations S_A est un cas particulier de séquence de groupes dans laquelle chaque groupe ne contient qu'une opération. Elle peut être représentée par un graphe potentiels-tâches $\mathcal{G}$ pour le calcul des dates de début au plus tôt identique à celui présenté en II.3.3.a) mais ne contenant pas d'arc de type groupe. Aussi, $\mathcal{G}$ peut être utilisé pour le calcul des dates de fin au plus tard. A chaque opération $o \in \mathcal{O} \cup \mathcal{S}$ de $\mathcal{G}$ sont ainsi associées une date de début au plus tôt r_o et une date de fin au plus tard d_o.

Pour calculer les contraintes temporelles associées à (i,j), on considère que l'opération est ajoutée dans la gamme d'un ordre de fabrication i. Si i a d'autres opération que (i,j), elles sont présentes dans l'ordonnancement et constituent des sommets de $\mathcal{G}$. L'ensemble des opérations précédentes de (i,j) dans la gamme de i est noté $prec^i_{ij}$. L'ensemble des opérations suivantes de (i,j) dans la gamme de i est noté $succ^i_{ij}$. Pour assurer l'existence d'une solution au problème d'insertion, on suppose qu'il n'existe pas de chemin dans $\mathcal{G}$ entre toute opération de $succ^i_{ij}$ et toute opération de $prec^i_{ij}$. Cette hypothèse n'est pas restrictive car elle permet de traiter tous les cas possibles d'insertion présentés ci-après. Dans ces conditions, on affecte à (i,j) une date de début au plus tôt r_{ij} et une date de fin au plus tard d_{ij} définies par :

$$r_{ij} = r_i \qquad si\ prec^i_{ij} = \emptyset \tag{III.1}$$

$$r_{ij} = \max_{(i,h)\in prec^i_{ij}} r_{ih} + p_{ih} \qquad si\ prec^i_{ij} \neq \emptyset \tag{III.2}$$

$$d_{ij} = d_i \qquad si\ succ^i_{ij} = \emptyset \tag{III.3}$$

$$d_{ij} = \min_{(i,q)\in succ^i_{ij}} d_{iq} - p_{iq} \qquad si\ succ^i_{ij} \neq \emptyset \tag{III.4}$$

De plus, (i, j) possède toutes les caractéristiques d'une opération d'exécution du problème défini dans le paragraphe II.2, avec en particulier un ensemble de modes étendus.

Le problème d'insertion étendu à un ensemble d'ordonnancements caractérisés par une séquence de groupes d'opérations permutables est un des éléments centraux de notre étude. En effet, il apparaît dans chacune des phases de l'approche proposée. La procédure de génération d'une séquence initiale de groupe (module SINIT) peut consister à insérer une par une les opérations des ordres de fabrication dans une suite de problèmes A_k à partir d'un problème initial A_0 ne contenant aucune opération. On montre en V.2 l'intérêt de cette considération en présence de temps de préparation. Au cours de la procédure de recherche de l'admissibilité de la séquence (module ANALYSE), l'amélioration est effectuée par une succession de transferts d'opération, un transfert étant défini comme la suppression d'une opération, suivie de sa ré-insertion à une autre position ou sur d'autres ressources. Le chapitre IV est consacré à l'étude de procédures d'amélioration basées sur l'insertion qui sont utilisées dans le module ANALYSE présenté en V.3. Enfin, le problème d'insertion doit être fréquemment résolu en temps réel (module ARBITRE) lors d'une arrivée imprévue d'un ordre de fabrication, d'une opération imprévue à ajouter dans une gamme ou encore d'une requête de changement d'affectation (voir paragraphe V.4). Pour simplifier la présentation des méthodes d'insertion, on considère dans ce chapitre le cas d'un ordonnancement unique et l'extension aux séquences de groupes est présentée dans le chapitre V.

Le problème d'insertion peut être décomposé en deux sous-problèmes: le problème d'affectation de l'opération (sélection d'une occupation des ressources l_{ij} et d'une occupation des lignes de charge $l_{ij}^k, \forall k \in \mathcal{R}_{ij}$) et le problème d'ordonnancement de l'opération (sélection d'une position sur chacune des lignes de charge allouées). L'insertion de (i, j) sur ses ressources principales dans le contexte de la préparation peut entraîner des modifications importantes dans la séquence des opérations de préparation déjà présentes, introduisant ainsi des difficultés supplémentaires. Plutôt que de traiter directement le problème général, des hypothèses simplificatrices sont introduites de façon à proposer une méthode de résolution efficace pour chaque problème particulier. On peut ainsi ignorer la préparation des ressources (paragraphes III.2 et III.3), considérer comme partiellement résolu le problème d'affectation en fixant l'occupation des ressources de l'opération (i, j) (paragraphes III.2 et III.4) et enfin considérer que les opérations de préparation ne nécessitent pas de ressource complémentaire (paragraphe III.4 et III.5). Le cas général est traité en III.6. La figure III.1 résume les problèmes traités dans ce chapitre.

III.2 Algorithme polynomial d'insertion d'une opération dans un ordonnancement cumulatif multi-ressources

III.2.1 Définition du problème

Dans ce paragraphe, on considère que le problème de sélection du mode étendu m de l'opération (i, j), des ressources dans chaque pool du mode m et du nombre de lignes de charge sur chaque ressource sélectionnée est résolu. L'occupation des ressources $l_{ij} = \{e_{ij}^1, ..., e_{ij}^M\}$ et la durée p_{ij} sont donc fixées. De plus, on suppose que l'opération

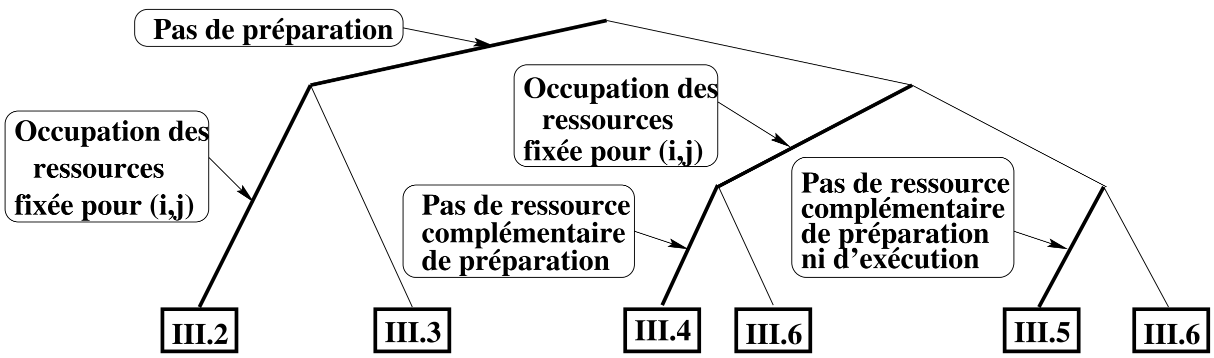


FIG. III.1 – Hiérarchie des problèmes traités dans le chapitre III

(i,j) utilise uniquement des ressources non concernées par la préparation (ressources complémentaires) ce qui se traduit par $\mathcal{P}b_{ij}^m = \emptyset$ et $\mathcal{P}c_{ij}^m \neq \emptyset$.

La préparation des ressources principales n'est alors pas remise en cause par l'insertion de (i,j). Bien que l'occupation des ressources l_{ij} soit fixée, le problème d'affectation n'est résolu que partiellement car l'occupation des lignes de charge l_{ij}^k sur chaque ressource cumulative k n'est fixée que par la contrainte $\sum_{l=1,\ldots,E_k} e_{ij}^{kl} = e_{ij}^k$. Il reste à choisir l'affectation de (i,j) sur e_{ij}^k lignes de charges à choisir parmi E_k sur chaque ressource k. On a un problème d'insertion d'une opération dans un ordonnancement à contraintes cumulatives et multi-ressources. On rappelle que $\mathcal{R}_{ij}$ est l'ensemble des ressources $k \in \mathcal{R}$ tel que $e_{ij}^k > 0$, c'est-à-dire l'ensemble des ressources (ici complémentaires) utilisées par (i,j).

En III.2.2, on définit formellement une position d'insertion et le problème de recherche d'une position d'insertion optimale. En III.2.3, un ensemble d'attributs est associé aux arcs de type ressource de $\mathcal{G}$ et un ensemble de propriétés de ces arcs sont définies. Ces propriétés permettent de transformer le problème de recherche d'une position d'insertion optimale en problème de recherche d'un ensemble d'arcs de type ressource de $\mathcal{G}$ (paragraphe III.2.4). Un algorithme polynomial de recherche d'un ensemble d'arcs de type ressource correspondant à une position d'insertion optimale est présenté dans les paragraphes III.2.5, III.2.6 et III.2.7. Un exemple illustratif du déroulement de l'algorithme est donné en III.2.8. Ce paragraphe est basé sur [Artigues et Roubellat, 1997a] et [Artigues et Roubellat, 1997b].

III.2.2 Problème de recherche d'une position d'insertion optimale

Pour insérer l'opération (i,j) dans l'ordonnancement, il est nécessaire de définir la position dans la séquence sur chaque ressource de $\mathcal{R}_{ij}$. L'insertion de (i,j) se traduira ensuite dans le graphe $\mathcal{G}$ représentant l'ordonnancement par la création d'un nouveau sommet et de nouveaux arcs de type ressource, de type gamme et éventuellement par la suppression d'arcs de type ressource présents dans $\mathcal{G}$ avant l'insertion. Soit $\mathcal{G}^*$ ce nouveau graphe. La position définie pour l'insertion de (i,j) détermine les opérations origine et destination des nouveaux arcs créés : elle doit être telle que l'insertion de (i,j) ne crée pas de cycle dans le graphe $\mathcal{G}^*$. Une telle position est appelée une position d'insertion ($\mathcal{PI}$) *valide*.

Nous étudions d'abord dans le paragraphe III.2.2.a) la caractérisation d'une insertion valide dans des cas particuliers d'occupation des ressources l_{ij} avant de donner dans le paragraphe III.2.2.b) la définition d'une position d'insertion dans le cas général et les conditions nécessaires et suffisantes de validité d'une position d'insertion. Les modifications du graphe $\mathcal{G}$ pour insérer l'opération (i,j) sont décrites en III.2.2.c) . Le calcul de l'impact de cette insertion sur l'admissibilité de l'ordonnancement est donné en III.2.2.d).

III.2.2.a) Etude introductive de cas particuliers d'occupation des ressources $\boldsymbol{l_{ij}}$

i) Position d'insertion valide sur une ressource disjonctive Soit une opération (i,j) et une ressource disjonctive k telles que $e_{ij}^k = 1$ et $e_{ij}^r = 0$, $\forall r \neq k$ ($\mathcal{R}_{ij} = \{k\}$). Sur une ressource disjonctive, les opérations séquencées sont totalement ordonnées. Une position dans la séquence est définie par exemple par une des opérations p affectées à k en supposant que (i,j) sera insérée dans la séquence immédiatement après p. Soit f l'opération immédiatement consécutive à p. (i,j) sera séquencée immédiatement avant f. Dans la figure III.2, la position d'insertion représentée est telle que $p = a$ et $f = b$.

Pour que cette insertion ne crée pas de cycle, il faut et il suffit qu'il n'existe pas de chemin dans le graphe $\mathcal{G}$ entre toute opération suivante de (i,j) dans l'ordre de fabrication i et p et qu'il n'existe pas non plus de chemin entre f et toute opération précédente de (i,j) dans l'ordre de fabrication i.

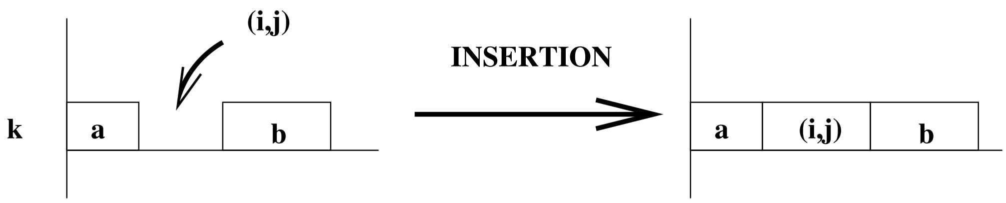


FIG. III.2 – Insertion sur une ressource disjonctive

ii) Position d'insertion valide sur un ensemble de ressources disjonctives Soit une opération (i,j) et un ensemble $\mathcal{R}_{ij}$ de M_{ij}^m ressources disjonctives tel que $\forall k \in \mathcal{R}_{ij}$, $e_{ij}^k = 1$ et $\forall q \in \mathcal{R} \setminus \mathcal{R}_{ij}$, $e_{ij}^q = 0$. Sur chaque ressource $k \in \mathcal{R}_{ij}$, on peut déterminer une position d'insertion de (i,j) c'est-à-dire une opération précédente $p(k)$ et une opération suivante $f(k)$ vérifiant les conditions de non création de cycle du cas mono-ressource. La position d'insertion est caractérisée par un ensemble de M_{ij}^m 3-uplets $(k, p(k), f(k))$. Dans la figure III.3, l'occupation des ressources de (i,j) est telle que $e_{ij}^{k_1} = e_{ij}^{k_2} = e_{ij}^{k_3} = 1$. La position d'insertion représentée est l'ensemble $\{(k_1,a,b),(k_2,a,c),(k_3,d,c)\}$ avec $p(k_1) = a$, $f(k_1) = b$, $p(k_2) = a$, $f(k_2) = c$, $p(k_3) = d$, $f(k_3) = c$.

Insérer une unique opération entre chaque couple d'opérations $\{p(k), f(k)\}$, $\forall k \in \mathcal{R}_{ij}$ crée un chemin entre chaque opération $p(k)$ et chaque opération $f(k')$, $\forall k, k' \in \mathcal{R}_{ij}$. Pour éviter la création d'un cycle, en plus des conditions établies pour le cas mono-ressource (i) , il faut et il suffit qu'il n'existe pas de chemin entre $f(k)$ et $p(k')$, ni entre $f(k')$ et $p(k)$, $\forall k, k' \in \mathcal{R}_{ij} \times \mathcal{R}_{ij}$.

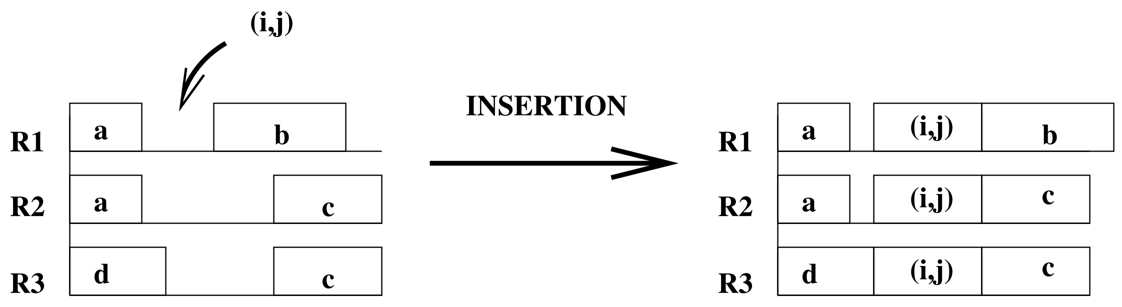


FIG. III.3 – Insertion sur plusieurs ressources disjonctive

iii) Position d'insertion valide sur une ressource cumulative Soit une opération (i, j) et une ressource cumulative k telles que $E_k \geq e_{ij}^k \geq 1$ et $e_{ij}^r = 0$, $\forall r \in \mathcal{R}$, $r \neq k$. L'ensemble $\mathcal{R}_{ij}$ est réduit à la ressource cumulative $\{k\}$. Les opérations séquencées sur k ne sont pas totalement ordonnées. Toutefois, une ressource cumulative est modélisée par un ensemble de lignes de charge interchangeables, chacune d'elles étant équivalente à une ressource disjonctive (voir paragraphe II.2). La définition d'une unique position d'insertion sur k peut ainsi se faire en deux étapes :

1. sélection d'un ensemble de e_{ij}^k lignes de charge parmi les E_k possibles, ce qui revient à déterminer l'occupation des lignes de charge l_{ij}^k et ainsi à résoudre totalement le problème d'affectation. Dans la figure III.4, une opération dont l'occupation des ressources Est telle que $e_{ij}^k = 3$, doit être insérée. L'occupation des lignes de charge sélectionnée sur la ressource k à 5 lignes de charge est $l_{ij}^k = \{1, 1, 1, 0, 0\}$.

2. sélection d'une position sur chaque ligne de charge l telle que $e_{ij}^{kl} = 1$, ce qui revient à un problème d'insertion sur e_{ij}^k ressources disjonctives. La position d'insertion est représentée par un ensemble de e_{ij}^k 3-uplets $(l, p(k, l), f(k, l))$. Dans l'exemple de la figure III.4, la position d'insertion est l'ensemble des 3-uplets $\{(1, a, b), (2, a, c), (3, d, c)\}$ avec $p(k, 1) = a$, $f(k, 1) = b$, $p(k, 2) = a$, $f(k, 2) = c$, $p(k, 3) = d$, $f(k, 3) = c$.

Les conditions de validité portant sur $p(k, l)$ et $f(k, l)$ sont les mêmes que les conditions de validité portant sur $p(k)$ et $f(k)$ dans le contexte multi-ressources disjonctif (ii).

III.2.2.b) Position d'insertion valide sur un ensemble quelconque de ressources

Soit une opération (i, j) et un ensemble $\mathcal{R}_{ij}$ de M_{ij}^m ressources cumulatives tel que $\forall k \in \mathcal{R}_{ij}$, $E_k \geq e_{ij}^k \geq 1$ et $\forall q \in \mathcal{R} \setminus \mathcal{R}_{ij}$, $e_{ij}^q = 0$.

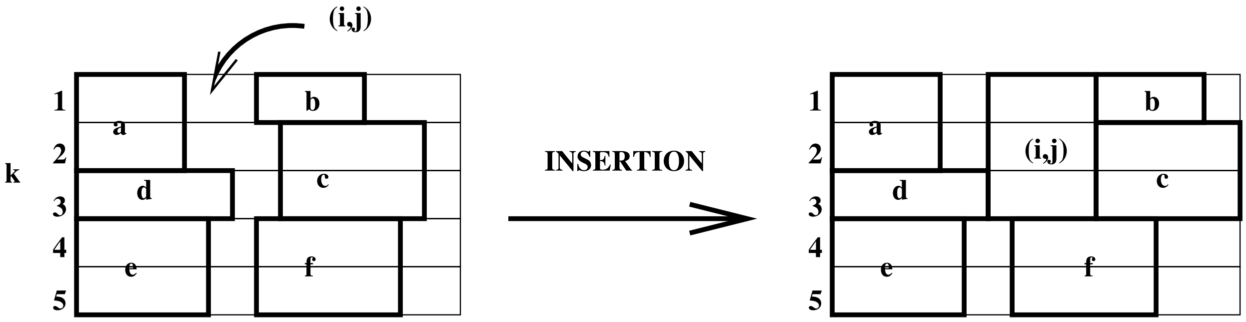


FIG. III.4 – Insertion sur une ressource cumulative

Ce cas correspond à une occupation des ressources l_{ij} quelconque et la définition d'une unique position d'insertion sur $\mathcal{R}_{ij}$ peut se faire en deux étapes :

- Sur chaque ressource $k \in \mathcal{R}_{ij}$, sélection d'un ensemble de e_{ij}^k lignes de charge parmi les E_k possibles, ce qui revient à déterminer l'occupation des lignes de charge l_{ij}^k. Dans la figure III.5, une opération (i,j), dont l'occupation des ressources l_{ij} est telle que $e_{ij}^{k_1} = 2$ et $e_{ij}^{k_2} = 1$, doit être insérée sur la ressource cumulative à 3 lignes de charge k_1 et sur la ressource disjonctive k_2. Sur k_1, on suppose que l'occupation des lignes de charge $l_{ij}^{k_1} = \{1,1,0\}$ a été sélectionnée. Sur k_2, l'occupation des lignes de charge $l_{ij}^{k_2} = \{1\}$ est obligatoirement sélectionnée.

- sélection d'une position sur chaque ligne de charge l de chaque ressource k telle que $e_{ij}^{kl} = 1$, ce qui revient à un problème d'insertion sur $\sum_{k \in \mathcal{R}_{ij}} e_{ij}^k$ ressources disjonctives. La position d'insertion est représentée par un ensemble de $\sum_{k \in \mathcal{R}_{ij}} e_{ij}^k$ 4-uplets $(k, l, p(k,l), f(k,l))$. Dans l'exemple de la figure III.5, la position d'insertion est l'ensemble $\{(k_1, 1, a, c), (k_1, 2, b, c), (k_2, 1, a, c)\}$ avec $p(k_1, 1) = a$, $f(k_1, 1) = c$, $p(k_1, 2) = b$, $f(k_1, 2) = c$, $p(k_2, 1) = a$, $f(k_2, 1) = c$.

On définit une position d'insertion dans un ordonnancement :

Définition 4 *Une position d'insertion dans un ordonnancement représenté par un graphe $\mathcal{G}$, est un ensemble $\mathcal{PI}$ de 4-uplets $(k, l, p(k,l), f(k,l))$ tels que :*

1. *$\forall (k, l, p(k,l), f(k,l)) \in \mathcal{PI}$, $p(k,l)$ et $f(k,l)$ soient consécutives sur la ligne de charge l de la ressource k.*

2. *$\forall \{(k, l, p(k,l), f(k,l)), (k', l', p(k', l'), f(k', l'))\} \in \mathcal{PI} \times \mathcal{PI}$, il n'existe pas de chemin dans $\mathcal{G}$ entre $f(k,l)$ et $p(k', l')$ ni entre $f(k', l')$ et $p(k,l)$ (relation de compatibilité entre 4-uplets)*

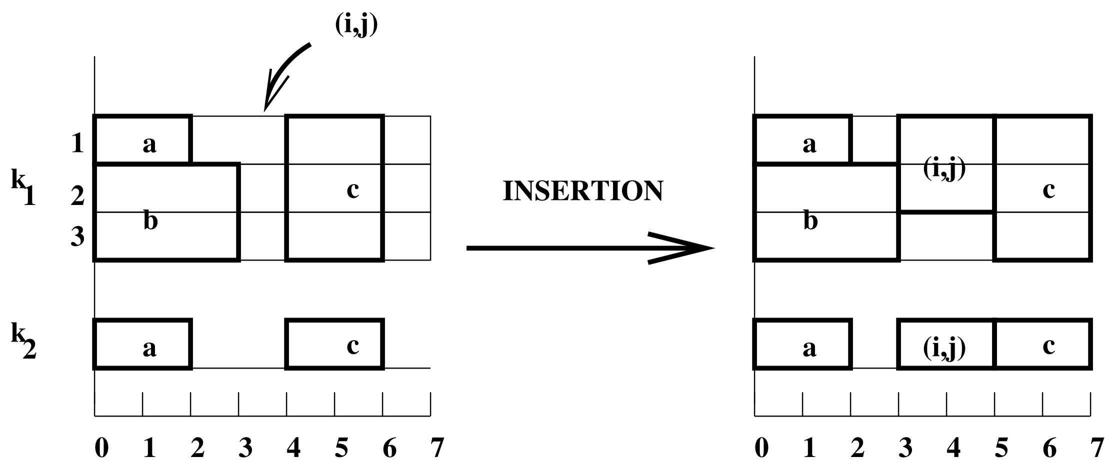


FIG. III.5 – Insertion sur une ressource cumulative et une ressource disjonctive

On définit une position d'insertion valide en donnant des conditions nécessaires et suffisantes de validité :

Définition 5 *Un ensemble* $\mathcal{PI} = \{(k, l, p(k, l), f(k, l))\}$ $k \in \mathcal{R}_{ij}, l \in 1, ..., E_k$ *tel que* $p(k, l)$ *et* $f(k, l)$ *sont consécutives sur la ligne de charge* l *de la ressource* k *est une position d'insertion valide de* (i, j) *si et seulement si*

1. $\mathcal{PI}$ *est une position d'insertion dans* $\mathcal{G}$
2. $\forall k \in \mathcal{R}_{ij}$, *il y a exactement* e_{ij}^k *lignes de charge différentes* l *telles que* $(k, l, p(k, l), f(k, l)) \in \mathcal{PI}$,
3. $\forall (k, l, p(k, l), f(k, l)) \in \mathcal{PI}$, $\forall (i, h) \in prec_{ij}^i, \forall (i, q) \in succ_{ij}^i$, *il n'existe pas de chemin dans* $\mathcal{G}$ *entre* $f(k, l)$ *et* (i, h) *ni entre* (i, q) *et* $p(k, l)$ *(relation de compatibilité entre le 4-uplet (k,l,p(k,l),f(k,l)) et l'opération* (i, j)*).*

III.2.2.c) Modification de $\mathcal{G}$ pour effectuer l'insertion

L'insertion de (i, j) dans une position d'insertion valide $\mathcal{PI}$ entraîne les modifications suivantes dans $\mathcal{G}$. Soit une opération p telle que $\exists k \in \mathcal{R}_{ij} / \exists l \in 1, ..., E_k$, $p(k, l) = p$. Un arc de type ressource $u_{p(i,j)}$ est généré entre chaque opération p et (i, j). Dans l'exemple de la figure III.5, un arc de type ressource $u_{a(i,j)}$ est créé entre a et (i, j). Soit une opération f telle que $\exists k \in \mathcal{R}_{ij} / \exists l \in 1, ..., E_k$, $f(k, l) = f$. Un arc de type ressource $u_{(i,j)f}$ est généré entre (i, j) et chaque opération f. Dans le graphe de la figure III.6 correspondant a l'exemple de la figure III.5, l'arc de type ressource $u_{(i,j)c}$ est créé entre (i, j) et c. Soit deux opérations p et f telles que $\exists k \in \mathcal{R}_{ij}$ et $\exists l \in 1, ..., E_k$ telles que $(k, l, p, f) \in \mathcal{PI}$. L'arc u_{pf} entre p et f est détruit (arc u_{ac} dans l'exemple de la figure III.6), à moins que p et f ne soient consécutives sur une ligne de charge l d'une ressource k telle que $e_{ij}^{kl} = 0$. C'est le cas de l'arc u_{bc} dans la figure III.6. Dans ce cas, l'arc entre p et f doit subsister dans $\mathcal{G}$ après l'insertion. Il est noté u_{pf}^*.

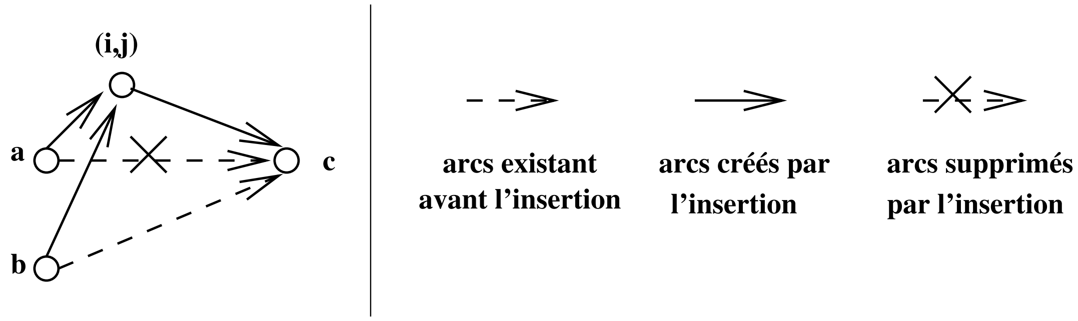


FIG. III.6 – Graphe correspondant à la figure III.5

III.2.2.d) Impact de l'insertion de l'opération dans une position d'insertion valide sur l'admissibilité

L'impact Δd de l'insertion de (i,j) sur l'admissibilité de l'ordonnancement représenté par $\mathcal{G}$ est calculé à l'aide des dates de fin au plus tard des opérations. En effet, ces dates ont été calculées de telle sorte que le décalage de la date fin au plus tôt d'une opération de Δd unités par rapport à sa date de fin au plus tard entraîne l'augmentation du retard vrai d'au moins un ordre de fabrication de Δd unités.

Soit $\mathcal{PI}$ une position d'insertion valide de (i,j). Etant donné qu'après l'insertion, (i,j) succède à toutes les opérations $p(k,l)$, la date de début au plus tôt de (i,j) s'exprime par :

$$r^*_{ij} = \max(r_{ij}, \max_{(k,l,p(k,l),f(k,l))\in\mathcal{PI}}(r_{p(k,l)} + p_{p(k,l)})) \tag{III.5}$$

Après l'insertion, (i,j) précède toutes les opérations $f(k,l)$. Chaque opération $f(k,l)$ et chaque opération $(i,q) \in succ^i_{ij}$ ne pourra pas commencer avant la date $r^*_{ij} + p_{ij}$. Si cette date est inférieure à la date de début au plus tard dans $\mathcal{G}$ de chaque opération $f(k,l)$, et inférieure à la date de fin au plus tard imposée par les contraintes de gamme de l'ordre de fabrication i, c'est-à-dire inférieure à la date de fin au plus tard donnée par :

$$d^*_{ij} = \min(d_{ij}, \min_{(k,l,p(k,l),f(k,l))\in\mathcal{PI}}(d_{f(k,l)} - p_{f(k,l)})) \tag{III.6}$$

alors l'insertion est admissible car elle respecte les dates de début au plus tard des opérations. Sinon l'insertion est dite non admissible et l'impact sur l'admissibilité vaut :

$$\Delta d_{max}(i,j,\mathcal{PI}) = \max(0, r^*_{ij} + p_{ij} - d^*_{ij}) \tag{III.7}$$

Dans l'exemple de la figure III.5, on suppose que l'opération (i,j) à insérer est telle que $p_{ij} = 2$, $r_{ij} = 0$ et $d_{ij} = 7$. De plus sur le diagramme de Gantt de gauche, les opérations a et b sont calées au plus tôt alors que l'opération c est calée au plus tard. On a ainsi $r^*_{ij} = \max(r_{ij}, 2, 3) = 3$ et $\Delta d_{max}(i,j,\mathcal{PI}) = \max(0, 3+2-7, 3+2-4) = 1$. L'insertion de (i,j) a un impact de 1 unité sur l'admissibilité.

III.2.2.e) Définition du problème de recherche d'une position d'insertion optimale

L'objectif de l'insertion revient ainsi à trouver un ensemble $\mathcal{PI}_{opt}$ de 4-uplets $\{(k, l, p(k,l), f(k,l))\}$, $k \in \mathcal{R}_{ij}$, $l \in 1, ..., E_k$ définissant une position d'insertion valide de (i,j) et d'impact minimal sur l'admissibilité parmi toutes les positions valides possibles.

$$\Delta d_{max}(i, j, \mathcal{PI}_{opt}) = \min_{\mathcal{PI} \in \mathcal{PI}(i,j,\mathcal{G})} \Delta d_{max}(i, j, \mathcal{PI}) \tag{III.8}$$

Une telle position d'insertion est appelée position d'insertion optimale.

L'énumération des positions d'insertions possibles pose un problème d'explosion combinatoire car elle suppose l'énumération des occupations des lignes de charge possibles de l'opération (i,j). Pour un nombre e_{ij}^k de lignes de charge requises par (i,j) sur une ressource k, il existe $C_{e_{ij}^k}^{E_k}$ occupations des lignes de charge possibles.

Dans le paragraphe suivant, nous montrons que le problème d'énumération des positions d'insertions valides est équivalent à un problème d'énumération d'ensembles d'arcs de type ressource du graphe $\mathcal{G}$ possédant certaines propriétés.

III.2.3 Définitions de base pour la caractérisation des positions d'insertion

Pour permettre à un arc de type ressource du graphe $\mathcal{G}$ de caractériser une position d'insertion, on lui ajoute des attributs supplémentaires dont une occupation des ressources et une occupation des lignes de charge. Ces attributs sont définis au paragraphe III.2.3.a). Des opérateurs, des relations et une condition de conservation des ressources sont définis pour les occupations dans le paragraphe III.2.3.b). On définit ensuite dans le paragraphe III.2.3.c) une relation de compatibilité entre les arcs de type ressource, le graphe de compatibilité représentant cette relation et des éléments remarquables de ce graphe appelés cliques.

III.2.3.a) Attributs associés aux arcs de type ressource de $\mathcal{G}$

Un arc de type ressource u_α du graphe $\mathcal{G}$ est caractérisé par :

- une opération origine $orig_\alpha$,
- une opération destination $dest_\alpha$,
- un poids p_o,
- une occupation des ressources $l_\alpha = \{e_\alpha^1, e_\alpha^2, ..., e_\alpha^M\}$ donnant pour chaque ressource de l'atelier le nombre de lignes de charge sur lesquelles les deux opérations sont consécutives.
- une occupation des lignes de charge sur chaque ressource k
  $l_\alpha^k = \{e_\alpha^{k1}, ..., e_\alpha^{kE_k}\}$ où $e_\alpha^{kl} = 1$ si $orig_\alpha$ et $dest_\alpha$ sont consécutives sur la ligne de charge l de la ressource k et $e_\alpha^{kl} = 0$ sinon. On a : $e_\alpha^k = \sum_{l=1,...,E_k} e_\alpha^{kl}$
- un intervalle I_α borné par
  - une date de début au plus tôt $r_\alpha = r_{orig_\alpha} + p_{orig_\alpha}$,

– une date de fin au plus tard $d_\alpha = d_{dest_\alpha} - p_{dest_\alpha}$,

$W_\alpha = d_\alpha - r_\alpha$ est la largeur de l'intervalle.

On dit que l'arc u_α "entre" dans une opération o si $o = dest_\alpha$. On note $\mathcal{U}_{entr}(o)$ l'ensemble des arcs entrant dans o. On dit que u_α "sort" de l'opération o si $o = orig_\alpha$. On note $\mathcal{U}_{sort}(o)$ l'ensemble des arcs sortant de o.

III.2.3.b) opérateurs, relations et propriétés des occupations des ressources

On définit les relations d'inclusion suivantes entre occupations des ressources.

- $l_1 \subset l_2$ ssi $\forall k \in [1, ..., M]$, $e_1^k < e_2^k$.
- $l_1 \supset l_2$ ssi $\forall k \in [1, ..., M]$, $e_1^k > e_2^k$
- $l_1 = l_2$ ssi $\forall k \in [1, ..., M]$, $e_1^k = e_2^k$.
- $l_1 \subseteq l_2$ ssi $\forall k \in [1, ..., M]$, $e_1^k \leq e_2^k$.
- $l_1 \supseteq l_2$ ssi $\forall k \in [1, ..., M]$, $e_1^k \geq e_2^k$.

On définit l'occupation des ressources vide $\emptyset_l$: $\forall k \in [1, .., M], e_\emptyset^k = 0$.
On définit les opérateurs suivants sur les occupations des ressources.

- l'union $l_1 \cup l_2$ de deux occupations l_1 et l_2 : $l_1 \cup l_2 = l_{\{u_1,u_2\}} = \{e_1^1 + e_2^1, ..., e_1^M + e_2^M\}$. L'union de deux occupations des ressources est une occupation des ressources.
- l'intersection $l_1 \cap l_2$ de deux occupations des ressources l_1 et l_2 : $l_1 \cap l_2 = \{\min(e_1^1, e_2^1), ..., \min(e_1^M, e_2^M)\}$. L'intersection de deux occupations des ressources est une occupation des ressources.
- Le complément $l_1 \setminus l_2$ de l'occupation des ressources l_2 dans l'occupation des ressources l_1 : $l_1 \setminus l_2 = \{e_1^1 - e_2^1, ..., e_1^M - e_2^M\}$ si $l_2 \subseteq l_1$ et $l_1 \setminus l_2 = l_1$ si $l_2 \not\subseteq l_1$

La relation suivante lie l'occupation des ressources d'une opération et les occupations des ressources des arcs de type ressource entrant et sortant de l'opération.

Remarque 1 *(Condition de conservation des ressources) L'occupation des ressources d'une opération sur l'ensemble des ressources est égale à l'union des occupations des ressources des arcs de type ressource entrant de l'opération ainsi qu'à l'union des occupations des ressources des arcs de type ressource sortant de l'opération.*

La condition de conservation des ressources s'exprime par :

$$l_o = (e_o^1, e_o^2, ..., e_o^M) = (\sum_{u_p \in \mathcal{U}_{entr}(o)} e_p^1, \sum_{u_p \in \mathcal{U}_{entr}(o)} e_p^2, ..., \sum_{u_p \in \mathcal{U}_{entr}(o)} e_p^M) \quad \text{(III.9)}$$

$$l_o = (\sum_{u_s \in \mathcal{U}_{sort}(o)} e_s^1, \sum_{u_s \in \mathcal{U}_{sort}(o)} e_s^2, ..., \sum_{u_s \in \mathcal{U}_{sort}(o)} e_s^M) \quad \text{(III.10)}$$

Ces relations, ces opérateurs et la condition de conservation des ressources peuvent être définis de la même manière pour les occupations des lignes de charge.

III.2.3.c) Compatibilité, graphe de compatibilité des arcs ressource, clique

i) relation de compatibilité On définit la relation suivante entre arcs de type ressource :

Définition 6 *Deux arcs u_α et u_β de type ressource de $\mathcal{G}$ sont* compatibles *s'il n'existe pas de chemin dans $\mathcal{G}$ entre $dest_\alpha$ et $orig_\beta$ ni entre $dest_\beta$ et $orig_\alpha$.*

ii) graphe de compatibilité des arcs de type ressource Pour un ensemble de ressources $\mathcal{R}s \subseteq \mathcal{R}$, on définit un graphe $\mathcal{G}u_{\mathcal{R}s}$ non orienté et non valué tel que les sommets de ce graphe correspondent aux arcs de type ressource u_α de $\mathcal{G}$ occupant une partie des lignes de charge d'au moins une ressource de $\mathcal{R}s$.

$$\forall u_\alpha \in \mathcal{G}u_{\mathcal{R}s}, \exists k \in \mathcal{R}s, e_\alpha^k > 0$$

Une arête relie deux sommets de $\mathcal{G}u_{\mathcal{R}s}$ si les deux arcs ressource associés sont compatibles. $\mathcal{G}u_{\mathcal{R}s}$ est appelé le graphe de compatibilité des arcs de type ressource de $\mathcal{R}s$.

iii) clique du graphe de compatibilité Une clique d'ordre q de $\mathcal{G}u_{\mathcal{R}s}$ représente un ensemble de q arcs de type ressource compatibles puisqu'ils sont reliés deux à deux par une arête dans le graphe $\mathcal{G}u_{\mathcal{R}s}$. On associe à une clique $\mathcal{C}$ les attributs suivants :

- un ensemble d'arcs de type ressource $\{u_1, u_2, ..., u_q\}$,
- une date de début au plus tôt $r_\mathcal{C} = \max\limits_{\alpha=1,\ldots,q} r_\alpha$,
- une date de fin au plus tard $d_\mathcal{C} = \min\limits_{\alpha=1,\ldots,q} d_\alpha$,
- une largeur $W_\mathcal{C} = d_\mathcal{C} - r_\mathcal{C}$,
- une occupation des ressources $l_\mathcal{C} = \bigcup\limits_{\alpha=1}^{q} l_\alpha$ et $e_\mathcal{C}^k = \sum\limits_{\alpha=1}^{q} e_\alpha^k$, $\forall k \in \mathcal{R}$,
- une occupation des lignes de charge sur chaque ressource $k \in \mathcal{R}$ $l_\mathcal{C}^k = \bigcup\limits_{\alpha=1}^{q} l_\alpha^k$ et $e_\mathcal{C}^{kl} = \sum\limits_{\alpha=1}^{q} e_\alpha^{kl}$, $\forall k \in \mathcal{R}$, $\forall l \in 1, ..., E_k$.

iv) clique maximale On appelle clique maximale de $\mathcal{G}u_{\mathcal{R}s}$, toute clique de $\mathcal{G}u_{\mathcal{R}s}$ non incluse dans une clique de $\mathcal{G}u_{\mathcal{R}s}$ d'ordre supérieur.

v) clique d'occupation des ressources maximale On appelle clique d'occupation des ressources maximale sur $\mathcal{R}$, toute clique de $\mathcal{G}_\mathcal{R}$ d'occupation $l_\mathcal{C}$ telle que $e_\mathcal{C}^k = E_k$, $\forall k \in \mathcal{R}$.

vi) exemple Dans la figure III.7, deux exemples de graphes $\mathcal{G}$ et $\mathcal{G}u_{\mathcal{R}s}$ sont représentés avec $\mathcal{R}s = \{k_1, k_2\}$. Dans l'exemple du haut de la figure, deux opérations $o1$ et $o2$ indépendantes sont ordonnancées respectivement sur deux ressources k_1 et k_2. 4 arcs de type ressource sont nécessaires pour représenter cet ordonnancement : u_1 entre l'opération fictive O et $o1$, u_2 entre l'opération fictive O et $o2$, u_3 entre $o1$ et l'opération fictive H et u_4 entre $o2$ et H. On peut représenter le graphe de compatibilité de ces arcs de type ressource en cherchant les chemins dans $\mathcal{G}$ entre les opérations origine et destination des arcs. Seules les paires d'arcs (u_1, u_3) et (u_2, u_4) ne sont pas compatibles. On remarque au passage qu'un tel graphe de compatibilité n'est pas un graphe d'intervalles au sens classique de la théorie des graphes puisque $\mathcal{G}u_{\mathcal{R}s}$ a un cycle d'ordre 4 sans corde [Berge, 1980]. Ajoutons une contrainte de précédence entre $o1$ et $o2$ matérialisée par un arc de type gamme u_5 dans le graphe du bas de la figure. Cet arc supplémentaire a pour effet de rendre incompatibles les arcs u_1 et u_4, ce qui se retrouve sur le graphe $\mathcal{G}u_{\mathcal{R}s}$ par l'absence de l'arête $u_1 u_4$.

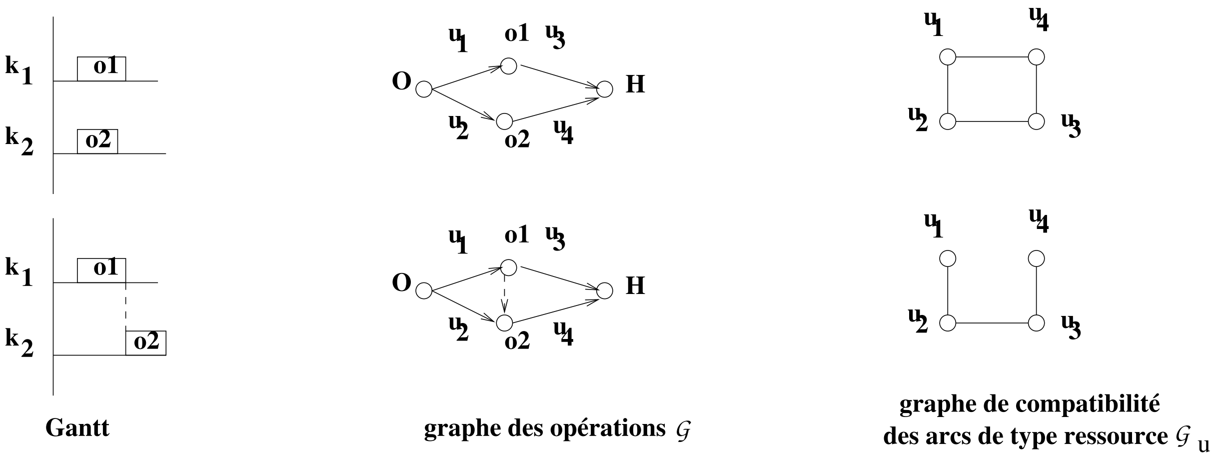


FIG. III.7 – Graphes de compatibilité des arcs de type ressource

III.2.4 Transformation du problème de recherche d'une position d'insertion optimale en problème de recherche d'une clique

On montre dans ce paragraphe que le problème de recherche d'une position d'insertion optimale est équivalent à un problème de recherche de clique. Pour cela, on donne des conditions suffisantes de caractérisation d'une position d'insertion par une clique (paragraphe III.2.4.a)) ainsi qu'une procédure permettant d'obtenir une position d'insertion à partir d'une clique suffisante (paragraphe III.2.4.b)). L'existence d'une clique suffisante qui caractérise toute position d'insertion est démontrée en III.2.4.c). Le paragraphe III.2.4.d) établit l'équivalence entre les deux problèmes. Une méthode générale de résolution du second problème est décrite en III.2.4.e).

III.2.4.a) Conditions suffisantes de caractérisation d'une position d'insertion par une clique

Dans ce paragraphe, on montre qu'une clique du graphe $\mathcal{G}u_{\mathcal{R}_{ij}}$ possédant certaines propriétés caractérise une position d'insertion incluant une position d'insertion valide pour l'opération (i,j).

clique compatible avec une opération d'exécution

Définition 7 *Une clique $\mathcal{C}$ et une opération (i,j) sont compatibles si et seulement si $\forall u_\alpha \in \mathcal{C}$, $\forall (i,h) \in prec^i_{ij}$, $\forall (i,q) \in succ^i_{ij}$, il n'existe pas de chemin entre $dest_\alpha$ et (i,h) ni entre (i,q) et $orig_\alpha$.*

Etant donné cette relation de compatibilité, qu'on peut rapprocher de la relation de compatibilité entre un 4-uplet et une opération décrite dans la définition 5, on donne une condition de caractérisation d'une unique position d'insertion par une clique.

clique valide

Théorème 1 *Toute clique $\mathcal{C}$ compatible avec une opération (i,j) et telle que $\forall k \in \mathcal{R}_{ij}$, $e^k_C = e^k_{ij}$ caractérise une unique position d'insertion valide $\mathcal{PI}(\mathcal{C})$ pour l'insertion de (i,j). Il s'agit de l'ensemble des 4-uplets $(k,l,p(k,l),f(k,l))$ tels que $\exists u_\alpha \in \mathcal{C}$, $k \in \mathcal{R}_{ij}$ et $e^{kl}_\alpha = 1$.*

Preuve Soit $\mathcal{C}$ une clique sur l'ensemble de ressources $\mathcal{R}_{ij}$. Pour chaque arc $u_\alpha \in \mathcal{C}$, on définit l'ensemble des 4-uplets $\{(k,l,orig_\alpha,dest_\alpha)\}$ tels que $k \in \mathcal{R}_{ij}$ et $e^{kl}_\alpha = 1$. k et l sont les ressources et lignes de charge sur lesquelles $orig_\alpha$ et $dest_\alpha$ sont consécutives. Soit $\mathcal{PI}(\mathcal{C})$ l'union des 4-uplets de tous les arcs de $\mathcal{C}$. Puisqu'une clique est un ensemble d'arcs compatibles, $\mathcal{PI}(\mathcal{C})$ vérifie par définition la condition 1 de la définition 5. De plus, la clique $\mathcal{C}$ et (i,j) étant compatibles, $\mathcal{PI}(\mathcal{C})$ vérifie la condition 3 de la définition 5 pour l 'insertion de (i,j). On démontre par l'absurde que la condition 2 de la définition 5 est vérifiée. Supposons (H_1) que $\forall k \in \mathcal{R}_{ij}$, $e^k_C = e^k_{ij}$. Supposons (H_2) qu'il existe une ressource $k \in \mathcal{R}_{ij}$ telle que le nombre de lignes de charge différentes l telles que $\{k,l,p(k,l),f(k,l)\} \in \mathcal{PI}(\mathcal{C})$ est différent de e^k_{ij} (négation de la condition 2). D'après l'hypothèse (H_1), $\forall k \in \mathcal{R}_{ij}$, $e^k_{\mathcal{C}} = \sum_{u_\alpha \in \mathcal{C}} e^k_\alpha = e^k_{ij}$. Or on sait que $\forall k \in \mathcal{R}_{ij}$, $e^k_\alpha = \sum_{l=1}^{E_k} e^{kl}_\alpha$ (voir II.3.3). Pour que (H_2) soit valide il faut et il suffit que $\exists k \in \mathcal{R}_{ij}$, $\exists l \in 1,...,E_k$, $\exists u_\alpha, u_\beta \in \mathcal{C} \times \mathcal{C}$, $e^{kl}_\alpha = e^{kl}_\beta = 1$, ce qui signifie que $orig_\alpha$ et $dest_\alpha$ d'une part et $orig_\beta$ et $dest_\beta$ d'autre part sont consécutives sur la même ligne de charge l. Il existe ainsi un chemin dans $\mathcal{G}$ entre $dest_\alpha$ et $orig_\beta$ ou entre $dest_\beta$ et $orig_\alpha$, ce qui est en contradiction avec la compatibilité de u_α et u_β. Les hypothèses (H_1) et (H_2) ne sont pas compatibles. $\mathcal{PI}(\mathcal{C})$ vérifie la condition 1 de la définition 5. En conséquence $\mathcal{C}$ définit une unique position d'insertion valide pour l'opération (i,j). □

On dit alors que $\mathcal{C}$ est une **clique valide** pour l'insertion de (i,j). On remarque que l'impact sur l'admissibilité de l'insertion de (i,j) à la position définie par $\mathcal{PI}(\mathcal{C})$ peut s'exprimer en fonction de $\mathcal{C}$ par :

$$\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C})) = \max\{0, \max(r_{ij}, r_{\mathcal{C}}) + p_{ij} - \min(d_{ij}, d_{\mathcal{C}})\} \qquad \text{(III.11)}$$

Exemple de clique valide Dans l'exemple de la figure III.5, la clique $\mathcal{C}_0$ $\{u_{ac}u_{bc}\}$ a une occupation des ressources $l_{\mathcal{C}_0} = \{3, 1\}$. Elle définit la position d'insertion $\{(k_1, 1, a, c), (k_1, 2, b, c), (k_1, 3, b, c), (k_2, 1, a, c)\}$. $\mathcal{C}_0$ est une clique valide pour une opération (r, s) d'occupation des ressources $l_{rs} = \{3, 1\}$. Supposons que dans le diagramme de gantt de la figure III.5, a et b sont calées au plus tôt alors que c est calée au plus tard. Supposons que $p_{rs} = 3$, $r_{rs} = 0$ et $d_{rs} = 7$. Dans ces conditions, $\Delta d_{max}(r, s, \mathcal{PI}(\mathcal{C}_0) = r_{u_{bc}} + p_{rs} - d_{u_{bc}} = 2$. L'insertion de (r, s) à la position $\mathcal{PI}(\mathcal{C}_0)$ a un impact de 2 unités de temps sur l'admissibilité.

Clique suffisante

Corollaire 1 *Une clique $\mathcal{C}$ compatible avec une opération (i, j) et telle que $l_\mathcal{C} \supseteq l_{ij}$ caractérise une position d'insertion $\mathcal{PI}(\mathcal{C})$ telle que la sélection dans $\mathcal{PI}(\mathcal{C})$ de e^k_{ij} 4-uplets différents $(k, l, p(k, l), f(k, l)) \in \mathcal{R}_{ij}$ pour chaque ressource $k \in \mathcal{R}_{ij}$ est une position d'insertion valide de (i, j). Il s'agit de l'ensemble des 4-uplets $(k, l, p(k, l), f(k, l))$ tels que $\exists u_\alpha \in \mathcal{C},\ k \in \mathcal{R}_{ij}$ et $e^{kl}_\alpha = 1$.*

On dit alors que $\mathcal{C}$ est une **clique suffisante** pour l'insertion de (i, j) à une position $\mathcal{PI} \subset \mathcal{PI}(\mathcal{C})$. $\Delta d_{max}(i, j, \mathcal{PI}(\mathcal{C}))$ est alors une borne supérieure de l'impact sur l'admissibilité de l'insertion de (r, s) dans une position d'insertion valide incluse dans $\mathcal{PI}(\mathcal{C})$.

Exemple de clique suffisante Dans l'exemple de la figure III.5, soit maintenant une opération (m, n) telle que $p_{mn} = 3, r_{mn} = 0, d_{mn} = 7$ et $l_{mn} = \{1, 1\}$. La clique $\mathcal{C}_0$ est maintenant un clique suffisante pour l'insertion de (m, n). $\Delta d_{max}(m, n, \mathcal{PI}(\mathcal{C}_0)) = 2$ est une borne supérieure de l'impact sur l'admissibilité de l'insertion de (m, n) dans une position incluse dans $\mathcal{PI}(\mathcal{C}_0)$. En effet, la position d'insertion $\mathcal{PI}s = \{(k_1, 1, a, c), (k_2, 1, a, c)\} \subset \mathcal{PI}(\mathcal{C}_0)$ est valide pour l'insertion de (m, n) et $\Delta d_{max}(m, n, \mathcal{PI}s) = 1$.

III.2.4.b) Procédure d'insertion d'une opération dans une clique suffisante

Par abus de langage on parle d'insertion d'une opération dans une clique d'arcs de type ressource au lieu d'insertion dans une position caractérisée par la clique. La figure III.8 décrit la procédure d'insertion InsèreClique d'une opération (i, j) d'occupation des ressources $l_{i,j}$ à une position d'insertion valide $\mathcal{PI}$ incluse dans la position d'insertion $\mathcal{PI}(\mathcal{C})$ associée à une clique suffisante $\mathcal{C}$. InsèreClique modifie le graphe $\mathcal{G}$ et définit l'occupation l^k_{ij} des lignes de charge de (i, j) sur chaque ressource $k \in \mathcal{R}_{ij}$. $\mathcal{PI}$ est sélectionnée arbitrairement dans $\mathcal{PI}(\mathcal{C})$ de la manière suivante. Il s'agit de créer au fur et à mesure les nouveaux arcs de type ressource et de modifier les occupations des arcs existant, voire de les supprimer si nécessaire. Les arcs $u_\alpha \in \mathcal{C}$ sont traités successivement. Pour chaque arc u_α, l_{reste} représente le nombre de lignes de charge auxquelles (i, j) n'est pas encore affectée. l_{inter} représente le nombre e^k_{inter} de lignes de charge occupées par u_α sur chaque ressource k qui seront allouées à (i, j). l^k_{inter} décrit précisément les lignes de charge qui seront affectées à (i, j) parmi les e^k_α lignes de charge caractérisées par u_α. La procédure sélectionne arbitrairement les e^k_{inter} premières. e^k_{inter} lignes de charge sont ainsi requises par (i,j) sur chaque ressource k entre $orig_\alpha$ et $dest_\alpha$. Un arc de type ressource $u_{\alpha ij}$ est créé entre $orig_\alpha$ et (i,j), d'occupation des ressources l_{inter}. Un autre arc de type ressource $u_{ij\alpha}$ est créé entre (i,j) et $dest_\alpha$ d'occupation des ressources l_{inter}. En fait, il est possible qu'un des deux arcs ait déjà été créé lors du traitement d'un autre arc $u_\beta \in \mathcal{C}$, les arcs de $\mathcal{C}$ étant traités successivement. Dans ce cas l'occupation des ressources de l'arc existant est augmentée de l_{inter}. L'occupation des ressources de l'arc u_α est diminuée de l_{inter}. Si $l_{inter} = l_\alpha$, toutes les lignes de charge caractérisées par u_α sont requises et l'arc u_α est détruit. Si au contraire $l_{inter} = 0$, aucune ligne de charge caractérisée par u_α n'est requise par (i,j). L'arc u_α est laissé intact dans $\mathcal{G}$ et aucun arc n'est créé entre $orig_\alpha$ et (i,j) ni entre (i,j) et $dest_\alpha$. Après le traitement de l'arc u_α, l_{reste} est diminuée de l_{inter} représentant les lignes de charge nouvellement allouées à (i,j). On passe ensuite à l'arc suivant de la clique $\mathcal{C}$. Lorsque tous les arcs ont été traités, l_{reste} doit être vide sinon la clique $\mathcal{C}$ n'est pas suffisante pour l'insertion de (i,j).

Cette procédure modifie le graphe $\mathcal{G}$ passé en argument. La complexité de l'algorithme est polynomiale en $\mathcal{O}(qME)$ où q est le nombre d'arcs de la clique, M est le nombre de ressources de l'atelier et $E = \max_{k=1}^{M} E_k$. En fait, un codage plus astucieux des occupations sous la forme d'une liste de couples (ressource, nombre de lignes de charge) permet d'obtenir un algorithme en $\mathcal{O}(qM_{ij}E_{ij})$ où M_{ij} est le nombre de ressources requises par (i,j) et $E_{ij} = \max_{k \in R_{ij}} e_{ij}^k$

III.2.4.c) Clique suffisante minimale

Dans le paragraphe III.2.4.a), on a montré que sous certaines conditions, une clique $\mathcal{C}$ de $\mathcal{G}u_{\mathcal{R}_{ij}}$ permet de caractériser une position d'insertion valide $\mathcal{PI}$ incluse dans la position $\mathcal{PI}(\mathcal{C})$ et telle que $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}))$ soit une borne supérieure de l'impact sur l'admissibilité de l'insertion de (i,j) dans $\mathcal{PI}$. La procédure InsèreClique décrite dans le paragraphe précédent permet d'insérer (i,j) à une position valide caractérisée par une clique suffisante $\mathcal{C}$ avec un impact sur l'admissibilité au plus égal à $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}))$.

Théorème 2 *Pour toute position d'insertion valide $\mathcal{PI}$ d'une opération (i,j), il existe au moins une clique suffisante $\mathcal{C}_s$ pour l'insertion de (i,j) telle que $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C})) = \Delta d_{max}(i,j,\mathcal{PI})$ et $\mathcal{PI} \subset \mathcal{PI}(\mathcal{C})$. Il s'agit de la clique formée des arcs $u_\alpha \in \mathcal{G}$ tels que $\exists k \in \mathcal{R}_{i,j}$ $\exists l \in 1,...,E_k$, $(k,l,orig_\alpha,dest_\alpha) \in \mathcal{PI}$.*

Preuve Soit la clique $\mathcal{C}$ composée des arcs $u_\alpha \in \mathcal{G}$ tels que $\exists k \in \mathcal{R}_{i,j}$ $\exists l \in 1,...,E_k$ tels que $(k,l,orig_\alpha,dest_\alpha) \in \mathcal{PI}$. $\mathcal{PI}(\mathcal{C})$ contient ainsi au moins toutes les lignes de charge contenues dans $\mathcal{PI}$. On a bien $\mathcal{PI} \subset \mathcal{PI}(\mathcal{C})$ et donc $l_\mathcal{C} \supseteq l_{ij}$. De plus, on a construit la clique $\mathcal{C}$ de telle sorte que $\forall p(k,l)$, $\forall f(k,l)$ tels que $(k,l,p(k,l),f(k,l)) \in \mathcal{PI}$, $\exists u_\alpha \in \mathcal{C}$ tel que $orig_\alpha = p(k,l)$ et $dest_\alpha = f(k,l)$. Par définition, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C})) = \Delta d_{max}(i,j,\mathcal{PI})$. □

Une telle clique est une **clique suffisante minimale** pour l'insertion de (i,j) dans une position $\mathcal{PI} \in \mathcal{PI}(\mathcal{C})$.

Proc INSÈRECLIQUE $((i,j), l_{ij}, \mathcal{C}, \mathcal{G}, \mathcal{R}, l_{ij}^k)$
  $l_{reste} \leftarrow l_{ij}$
  $\forall k \in \mathcal{R},\ l_{ij}^k \leftarrow \emptyset_l$
  Pour chaque arc $u_\alpha \in \mathcal{C} = \{u_1, ..., u_q\}$ **faire**
    $l_{inter} \leftarrow l_{reste} \cap l_\alpha$
    Pour chaque ressource $k \in \mathcal{R}$ **faire**
      $compteur \leftarrow 1$
      $l \leftarrow 1$
      Tant que $compteur \leq e_{inter}^k$ **faire**
        $e_{inter}^{kl} \leftarrow e_\alpha^{kl}$
        Si $e_\alpha^{kl} = 1$ **alors**
          $compteur \leftarrow compteur + 1$; $e_\alpha^{kl} = 0$
        Fin Si
        $l \leftarrow l + 1$
      FinTant que
    FinPour
    $l_{reste} \leftarrow l_{reste} \setminus l_{inter}$
    $l_\alpha \leftarrow l_\alpha \setminus l_{inter}$
    $\forall k \in \mathcal{R},\ l_{ij}^k \leftarrow l_{ij}^k \cup l_{inter}^k$
    Si $l_\alpha = \emptyset_l$ **alors**
      supprimer u_α de $\mathcal{G}$.
    FinSi
    Si $l_{inter} \neq \emptyset_l$ **alors**
      Si l'arc $u_{\alpha,i,j}$ entre $orig_\alpha$ et (i,j) existe **alors**
        mise à jour de : $l_{\alpha,i,j} \leftarrow l_{\alpha,i,j} \cup l_{inter}$; $\forall k \in \mathcal{R}, l_{\alpha,i,j}^k \leftarrow l_{\alpha,i,j}^k \cup l_{inter}^k$
      Sinon
        Créer l'arc $u_{\alpha,i,j}$ $l_{\alpha,i,j} \leftarrow l_{inter}$; $\forall k \in \mathcal{R},\ l_{\alpha,i,j}^k \leftarrow l_{inter}^k$
      FinSi
      Si l'arc $u_{i,j,\alpha}$ entre les opérations (i,j) et $dest_\alpha$ existe **alors**
        mise à jour de $l_{i,j,\alpha} \leftarrow l_{i,j,\alpha} \cup l_{inter}$; $\forall k \in \mathcal{R}, l_{i,j,\alpha}^k \leftarrow l_{i,j,\alpha}^k \cup l_{inter}^k$
      Sinon
        Créer l'arc $u_{i,j,\alpha}$ tel que $l_{i,j,\alpha} \leftarrow l_{inter}$; $\forall k \in \mathcal{R}, l_{i,j,\alpha}^k \leftarrow l_{inter}^k$.
      FinSi
    FinSi
    Si $l_{reste} = \emptyset_l$ **alors Fin**
    FinSi
  FinPour
  Si $l_{reste} \neq \emptyset_l$ **alors**
    l'occupation des ressources de $\mathcal{C}$ est insuffisante : **Erreur**
  FinSi
FinProc

FIG. III.8 – Algorithme de INSÈRECLIQUE

exemple Reprenons l'exemple de la figure III.5. Considérons l'opération (i,j) d'occupation des ressources $\{2,1\}$. $\mathcal{C}_0 = \{u_1, u_2\}$ est une clique suffisante minimale pour toute position d'insertion valide de (i,j) incluse dans $\mathcal{PI}(\mathcal{C}_0)$.

III.2.4.d) Clique suffisante optimale et transformation du problème de recherche d'une position d'insertion optimale

On a ainsi montré que pour toute position d'insertion valide $\mathcal{PI}$ de (i,j), il existe une clique suffisante $\mathcal{C}$ telle que l'insertion de (i,j) dans $\mathcal{C}$ par InsèreClique a un impact sur l'admissibilité égal à l'impact sur l'admissibilité de l'insertion de (i,j) dans $\mathcal{PI}$. En particulier, il existe une clique suffisante $\mathcal{C}_{opt}$ telle que $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_{opt})) = \Delta d_{max}(i,j,\mathcal{PI}_{opt})$, appelée **clique suffisante optimale**. L'énumération des cliques suffisantes de $\mathcal{G}u_{\mathcal{R}_{ij}}$ permet de trouver une position d'insertion optimale.

III.2.4.e) Description générale de la procédure de recherche d'une clique suffisante optimale

Dans [Le Gall, 1989] et [Billaut, 1993], une procédure d'exploration des cliques d'un graphe de compatibilité a été définie pour un problème légèrement différent, car seules des positions d'insertion admissibles étaient recherchées. Cette procédure en deux étapes peut être utilisée pour la recherche d'une clique suffisante optimale :

Etape 1 Génération du graphe de compatibilité $\mathcal{G}u_{\mathcal{R}_{ij}}$

Etape 2 Enumération exhaustive des cliques de $\mathcal{G}u_{\mathcal{R}_{ij}}$

La construction des sommets de $\mathcal{G}u_{\mathcal{R}_{ij}}$ nécessite l'énumération des arcs $u_\alpha \in \mathcal{G}$ pour lesquels $\exists k \in \mathcal{R}b_{ij}$ telle que $e_\alpha^k > 0$. La construction des arêtes de $\mathcal{G}u_{\mathcal{R}_{ij}}$ nécessite pour chaque paire d'arcs u_α, u_β du graphe le test de l'existence de chemin entre $dest_\alpha$ et $orig_\beta$ d'une part et entre $dest_\beta$ et $orig_\alpha$ d'autre part. Un algorithme de construction du graphe peut être réalisé en $\mathcal{O}(U^3)$ où U est le nombre d'arcs du graphe. Il peut être ainsi intéressant d'explorer les cliques sans générer explicitement ce graphe.

On peut remarquer que toute clique de $\mathcal{G}u_{\mathcal{R}_{ij}}$ est par définition incluse dans une clique maximale. La deuxième étape de l'algorithme peut ainsi être décomposée de la façon suivante :

Etape 2.1 Enumération des cliques maximales de $\mathcal{G}u_{\mathcal{R}_{ij}}$

Etape 2.2 Pour chaque clique maximale, recherche de la sous-clique suffisante $\mathcal{C}_s$ de $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s))$ minimal

Théorème 3 *Une clique $\mathcal{C}$ est d'occupation des ressources maximale sur $\mathcal{R}_{ij}$ si et seulement si $\mathcal{C}$ est une clique maximale de $\mathcal{G}u_{\mathcal{R}_{ij}}$*

Preuve Soit $\mathcal{C}$ une clique d'occupation des ressources maximale sur l'ensemble de ressources $\mathcal{R}_{ij}$. Soit un arc $u_\beta \notin \mathcal{C}$, une ressource k et une ligne de charge l de k tels que $k \in \mathcal{R}_{ij}$ et $e_\beta^{kl} = 1$. $\mathcal{C}$ étant d'occupation maximale des ressources, $\exists u_\alpha \in \mathcal{C}$, $e_\alpha^{kl} = 1$. Les opérations $orig_\alpha, orig_\beta$, $dest_\alpha$ et $dest_\beta$ étant affectées à la même ligne de charge, il existe dans $\mathcal{G}$ un chemin entre $dest_\alpha$ et $orig_\beta$ ou entre $dest_\beta$ et $orig_\alpha$. Les arcs u_α et u_β ne peuvent pas être compatibles. $\mathcal{C}$ est une clique maximale.

Soit maintenant $\mathcal{C}^{max}$ une clique maximale de $\mathcal{G}u_{\mathcal{R}_{ij}}$. Démontrons par l'absurde qu'elle est aussi d'occupation maximale des ressources (H). Supposons que $\exists k \in \mathcal{R}_{ij}$

et $\exists l \in 1, ..., E_k$, $e^{kl}_{\mathcal{C}^{max}} = 0$ (négation de H). Soit $\mathcal{O}(k,l)$ l'ensemble des opérations affectées à la ligne de charge l de la ressource k. Au minimum $\mathcal{O}(k,l) = \{O, H\}$. Parmi ces opérations soit l'ensemble $\mathcal{O}^{inf}(k,l)$ tels qu'il existe un chemin entre $o \in \mathcal{O}^{inf}(k,l)$ et une opération p telle que $\exists u_\alpha \in \mathcal{C}^{max}$, $orig_\alpha = p$. Soit l'ensemble $\mathcal{O}^{sup}(k,l)$ tels qu'il existe un chemin entre $o \in \mathcal{O}^{sup}(k,l)$ et une opération f telle que $\exists u_\beta \in \mathcal{C}^{max}$, $dest\beta = f$. $\mathcal{C}^{max}$ étant maximale, toute opération de $\mathcal{O}(k,l)$ appartient à un de ces deux ensembles. Les opérations de $\mathcal{O}(k,l)$ étant consécutives sur la ligne de charge, il existe un arc de type ressource u_d tel que $orig_d \in \mathcal{O}^{inf}(k,l)$ et $dest_d \in \mathcal{O}^{sup}(k,l)$ et $e^{kl}_d = 1$. Or cet arc de type ressource est compatible avec tous les arcs de $\mathcal{C}^{max}$, ce qui est en contradiction avec l'hypothèse que $\mathcal{C}^{max}$ est une clique maximale. Si $\mathcal{C}^{max}$ est maximale, il n'y a aucune ligne de charge l sur une ressource $k \in \mathcal{R}_{ij}$ telle que $e^{kl}_{\mathcal{C}^{max}} = 0$. $\mathcal{C}^{max}$ est d'occupation maximale des ressources sur $\mathcal{R}_{ij}$. □

L'énumération des cliques maximales de $\mathcal{G}u_{\mathcal{R}_{ij}}$ est équivalente à l'énumération des cliques d'occupation des ressources maximale sur l'ensemble $\mathcal{R}_{ij}$, c'est-à-dire les cliques $\mathcal{C}$ telles que $\forall k \in \mathcal{R}_{ij}$, $e^k_{\mathcal{C}} = E_k$. Comme on le verra dans le paragraphe suivant, les cliques d'occupation des ressources maximale peuvent être énumérées sans générer le graphe $\mathcal{G}u_{\mathcal{R}_{ij}}$.

On donne ainsi le principe de la recherche d'une clique suffisante optimale pour l'insertion de (i,j) d'occupation l_{ij} sur un ensemble de ressources $\mathcal{R}_{ij}$.

Etape 1 Exploration des cliques d'occupation des ressources maximale

Etape 2 Pour chaque clique d'occupation des ressources maximale, recherche de la meilleure sous-clique.

L'étape 1 est réalisée par la procédure ExplCliquesMaxDomin décrite dans le paragraphe III.2.5. L'étape 2 est réalisée par la procédure ExplSouscliquesDomin décrite dans le paragraphe III.2.6.

III.2.5 Procédure d'exploration des cliques maximales

On définit dans un premier temps des relations de dominance entre cliques maximales qui nous permettront d'éviter l'énumération de l'ensemble des cliques maximales. L'énumération des cliques maximales dominantes est suffisante pour l'obtention de la clique suffisante optimale, en supposant qu'on dispose d'une procédure qui recherche la meilleure sous-clique d'une clique donnée. La procédure d'exploration des cliques maximales dominantes part d'une clique maximale dominante initiale et explore l'ensemble des cliques maximales dominantes jusqu'à une clique dominante finale par une série de déplacements élémentaires.

Dans le paragraphe III.2.5.a), les relations de dominance entre cliques maximales sont énoncées. La génération d'une clique maximale voisine à partir d'une clique maximale donnée est défini dans le paragraphe III.2.5.b). La procédure de génération de la clique maximale dominante initiale et la caractérisation des cliques qu'elle domine sont présentées dans le paragraphe III.2.5.c). La procédure d'exploration des cliques maximales dominantes jusqu'à la clique maximale dominante finale est décrite dans le paragraphe III.2.5.d).

III.2.5.a) Relation de dominance entre cliques maximales

La relation de dominance entre cliques maximales s'appuie sur la relation de dominance entre sous-cliques suffisantes suivante :

Définition 8 *Une clique suffisante $\mathcal{C}'$ est dominée par toute clique suffisante $\mathcal{C}$ telle que $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C})) \leq \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}'))$.*

On définit alors la relation de dominance entre cliques maximales :

Définition 9 *Une clique maximale $\mathcal{C}$ domine une autre clique maximale $\mathcal{C}'$ si et seulement si $\forall \mathcal{C}'_s \subseteq \mathcal{C}'$, $\exists \mathcal{C}_s \subseteq \mathcal{C}$ telle que $\mathcal{C}_s$ domine $\mathcal{C}'_s$ au sens de la définition 8.*

Toute sous-clique de la clique maximale dominée est dominée au sens de la définition 8 par au moins une sous-clique de la clique maximale dominante. Autrement dit si on dispose d'une procédure qui trouve la meilleure sous-clique d'une clique donnée, la meilleure sous-clique suffisante de $\mathcal{C}$ domine la meilleure sous-clique suffisante de $\mathcal{C}'$.

III.2.5.b) Génération d'une clique maximale voisine

Soit une clique maximale $\mathcal{C}$ sur l'ensemble de ressources $\mathcal{R}_{ij}$. $l_{\mathcal{C}}$ est l'occupation maximale des ressources sur $\mathcal{R}_{ij}$. On définit deux types de modifications "horizontales" de cette clique appelés déplacements.

Déplacement vers la droite d'une clique On appelle déplacement vers la droite de $\mathcal{C}$ autour d'une opération o (noté dd_o) le remplacement dans $\mathcal{C}$ des arcs dont o est l'opération destination par l'ensemble des arcs dont o est l'opération origine et dont l'occupation des ressources a une intersection non vide avec $l_{\mathcal{C}}$:

$$\mathcal{C} \stackrel{dd_o}{\rightarrow} \mathcal{C}'$$

avec

$$\mathcal{C}' = dd_o(\mathcal{C}) = (\mathcal{C} \setminus \mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)) \cup \mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)$$

et $\mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)$ est le sous-ensemble des arcs u_α de $\mathcal{U}_{entr}(o)$ tel que $l_\alpha \cap l_{\mathcal{C}} \neq \emptyset_l$ et $\mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)$ est le sous-ensemble des arcs u_β de $\mathcal{U}_{sort}(o)$ tel que $l_\beta \cap l_{\mathcal{C}} \neq \emptyset_l$.

L'algorithme du déplacement vers la droite est en $\mathcal{O}(M_o^2)$ où M_o est le nombre maximum de ressources occupées par une opération.

Un déplacement est dit réalisable si l'ensemble des arcs qui en sont issus est une clique de $\mathcal{G}u_{\mathcal{R}}$, c'est-à-dire si ces arcs sont compatibles deux à deux.

Théorème 4 *Un déplacement vers la droite dd_o d'une clique $\mathcal{C}$ est réalisable si et seulement si $\forall u_\alpha \in \mathcal{C} \setminus \mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)$, il n'existe pas de chemin dans $\mathcal{G}$ entre $dest_\alpha$ et o.*

Preuve Si un tel chemin existe, alors les arcs issus de dd_o ne sont pas compatibles et ne forment pas une clique. Si un tel chemin n'existe pas, la première condition de compatibilité par rang entre chaque arc $u_\alpha \in \mathcal{C} \backslash \mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)$ et chaque arc $u_\beta \in \mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)$ est vérifiée. Il reste à donc prouver qu'il n'existe pas de chemin entre toute opération suivante sur les ressources de o et toute opération origine de chaque arc appartenant à la clique $\mathcal{C}$. Si un tel chemin existait, alors il existerait un chemin entre o et l'opération origine d'un arc de $\mathcal{C}$. $\mathcal{C}$ ne serait pas une clique. □

Théorème 5 *Tout déplacement réalisable vers la droite d'une clique maximale de $\mathcal{G}u_{\mathcal{R}_{ij}}$ est une clique maximale de $\mathcal{G}u_{\mathcal{R}_{ij}}$.*

Preuve L'union de l'occupation des ressources des arcs sortants d'une opération est par définition égale à l'occupation des ressources de l'opération. En particulier, le sous-ensemble $\mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)$ des arcs sortants de l'opération o_{dest} sur l'ensemble des ressources $\mathcal{R}_{ij}$ a une occupation des ressources $l_{\mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)}$ telle que $\forall r \in \mathcal{R}_{ij}$, $e^r_{\mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)} = e^r_o$. De même, l'union de l'occupation des ressources des arcs entrants d'une opération est égale à l'occupation des ressources de l'opération. En particulier, le sous-ensemble $\mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)$ a une occupation des ressources $l_{\mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)}$ telle que $\forall r \in \mathcal{R}_{ij}$, $e^r_{\mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)} = e^r_o$. L'ensemble des arcs sortants de l'opération o a donc une occupation des ressources supérieure ou égale à l'ensemble des arcs entrants de o et présents dans la clique $\mathcal{C}$. Si $\mathcal{C}$ est une clique maximale, et si le déplacement vers la droite est réalisable, alors $\mathcal{C}'$ est aussi une clique maximale. □

Déplacement vers la gauche d'une clique Soit o une opération origine commune à un sous-ensemble d'arcs de $\mathcal{C}$ On appelle déplacement vers la gauche de $\mathcal{C}$ autour de o (noté dg_o) le remplacement dans $\mathcal{C}$ des arcs dont o est l'opération origine par l'ensemble des arcs dont o est l'opération destination et dont l'occupation des ressources a une intersection non vide avec l'occupation des ressources de $\mathcal{C}$.

$$\mathcal{C} \overset{dg_o}{\to} \mathcal{C}'$$

avec

$$\mathcal{C}' = dg_o(\mathcal{C}) = (\mathcal{C} \setminus \mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)) \cup \mathcal{U}_{entr}^{\mathcal{R}_{ij}}(o)$$

Théorème 6 *Un déplacement vers la gauche dg_o d'une clique $\mathcal{C}$ est réalisable si et seulement si $\forall u_\alpha \in \mathcal{C} \setminus \mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o)$, il n'existe pas de chemin dans $\mathcal{G}$ entre $orig_\alpha$ et o*

Preuve symétrique du théorème 4. □

Théorème 7 *Tout déplacement réalisable vers la gauche d'une clique maximale de $Gu_{\mathcal{R}_{ij}}$ est une clique maximale de $Gu_{\mathcal{R}_{ij}}$.*

Preuve symétrique du théorème 5. □

Exemple illustratif Reprenons l'exemple de l'ordonnancement décrit dans la figure III.7. Dans la figure III.9, on peut effectuer des déplacements vers la gauche et vers la droite autour des opérations o_1 et o_2, ce qui permet d'obtenir les cliques $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_3$ et $\mathcal{C}_4$. Ces déplacements sont réalisables car il n'existe pas de chemin dans le graphe entre $o1$ et $o2$. Si on rajoute une contrainte de précédence entre o_1 et o_2 on ne peut plus effectuer le déplacement dd_{o2} à partir de la clique $\mathcal{C}_1$ ni le déplacement dg_{o_1} à partir de la clique $\mathcal{C}_3$. On vérifie dans le graphe de compatibilité de la figure III.7 que les arcs u_1 et u_4 ne sont pas compatibles.

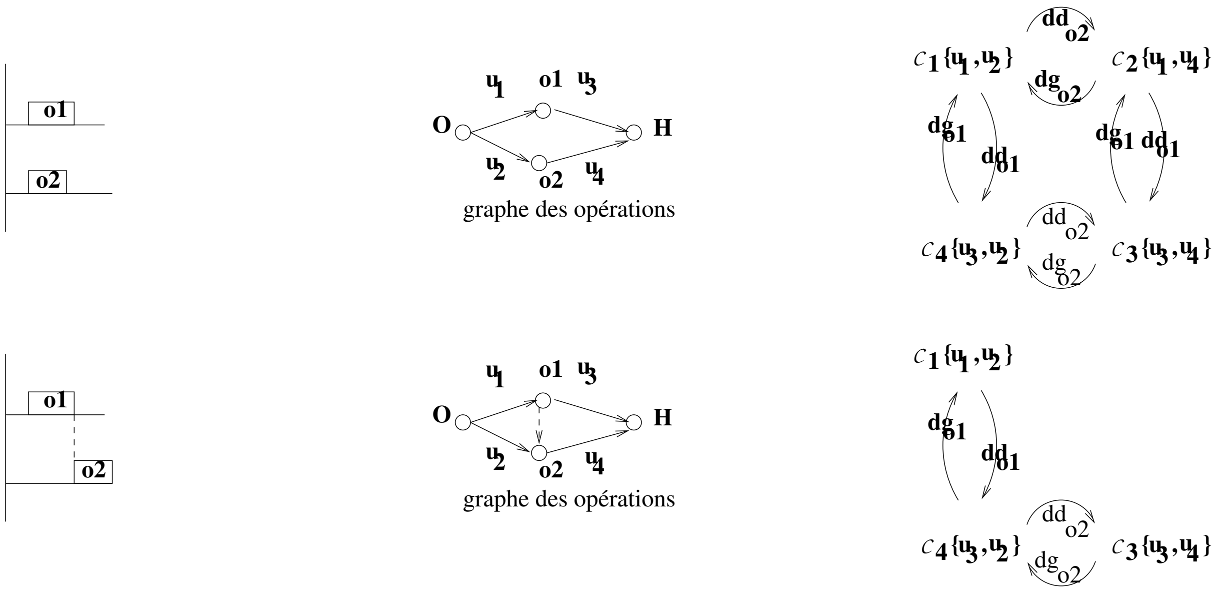


FIG. III.9 – Déplacements vers la gauche et vers la droite de cliques maximales

propriétés temporelles des déplacements Des relations temporelles entre arcs de type ressource adjacents sont à la base de la détermination de propriétés temporelles des déplacements.

Remarque 2 *Pour toute opération o telle que* $p_o > 0$, $\forall u_\alpha \in \mathcal{U}_{entr}(o)$ *et* $\forall u_\beta \in \mathcal{U}_{sort}(o)$, *on a* $r_\alpha < r_\beta$ *et* $d_\beta > d_\alpha$

La figure III.10 illustre ces deux relations temporelles pour deux arcs adjacents sur une ressource disjonctive.

En conséquence une clique $\mathcal{C}'$ obtenue par un déplacement vers la droite faisable d'une clique $\mathcal{C}$ est telle que $r_{\mathcal{C}'} \geq r_{\mathcal{C}}$ et $d_{\mathcal{C}'} \geq d_{\mathcal{C}}$.

Conditions suffisantes pour les déplacements réalisables On appelle *clique maximale initiale* $\mathcal{C}_{init}^{\mathcal{R}_{ij}}$ sur un ensemble de ressources $\mathcal{R}_{ij}$, l'ensemble des arcs de type ressource u_p de $\mathcal{G}$ tels que $orig_p = O$ (sommet origine du graphe) et $\exists k \in \mathcal{R}_{i,j} / e_p^k > 0$, c'est-à-dire

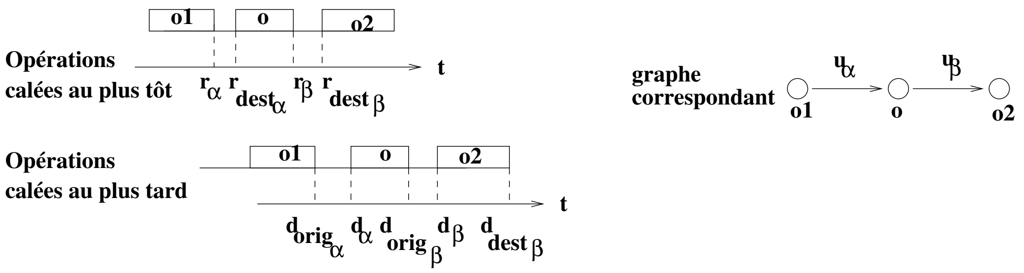


FIG. III.10 – Relations temporelles entre arcs adjacents

l'ensemble des premiers arcs sur ces ressources. Ces arcs forment bien une clique d'occupation maximale des ressources sur $\mathcal{R}_{ij}$.

On appelle *clique maximale finale* $\mathcal{C}_{fin}^{\mathcal{R}_{ij}}$, l'ensemble des arcs u_s de $\mathcal{G}$ tels que $dest_s = H$, et $\exists k \in \mathcal{R}_{i,j} / e_s^k > 0$, c'est-à-dire l'ensemble des derniers arcs sur ces ressources. Ces arcs forment bien une clique d'occupation maximale des ressources sur $\mathcal{R}_{ij}$.

Théorème 8 *Pour toute clique maximale $\mathcal{C}$ de $\mathcal{G}_{\mathcal{R}_{ij}}$, $\mathcal{C} \neq \mathcal{C}_{init}^{\mathcal{R}_{ij}}$, sont toujours réalisables les déplacements vers la gauche autour d'une opération origine de plus grande date de fin au plus tôt, de plus grande date de début au plus tôt, de plus grande date de fin au plus tard ou de plus grande date de début au plus tard parmi les opérations origine des arcs de $\mathcal{C}$.*

Pour toute clique maximale $\mathcal{C}$ de $\mathcal{G}_{\mathcal{R}_{ij}}$, $\mathcal{C} \neq \mathcal{C}_{fin}^{\mathcal{R}_{ij}}$, sont toujours réalisables les déplacements vers la droite autour d'une opération destination de plus petite date de début au plus tard, de plus petite date de fin au plus tard, de plus petite date de début au plus tôt ou de plus petite date de fin au plus tôt parmi les opérations destination des arcs de $\mathcal{C}$.

Preuve Soit une clique maximale $\mathcal{C}$. Soit une opération o_{orig} telle que $\exists u_\alpha \in \mathcal{C} / o_{orig} = orig_\alpha$ et $r_\alpha = \max_{u_\beta \in \mathcal{C}} r_\beta$. Il ne peut exister de chemin entre o_{orig} et aucune autre des opérations origines de la clique puisque o_{orig} se termine au plus tôt après toutes les autres. D'après le théorème 6, le déplacement vers la gauche autour de o_{orig} est valide.

Soit une opération o_{dest} telle que $\exists u_\alpha \in \mathcal{C} / o_{dest} = dest_\alpha$ et $d_\alpha = \min_{u_\beta \in \mathcal{C}} d_\beta$. Il ne peut exister de chemin entre aucune des autres opérations destinations de la clique et o_{dest} puisque o_{dest} commence au plus tard avant toutes les autres. D'après le théorème 4, le déplacement vers la droite autour de o_{dest} est valide.
Un raisonnement identique est applicable aux autres possibilités. □

La figure III.11 illustre ce théorème pour le déplacement vers la droite de la clique $\{u_1, u_2\}$. Ici, $o4$ a la plus petite date de début au plus tard. Le déplacement vers la droite autour de $o4$ est valide.

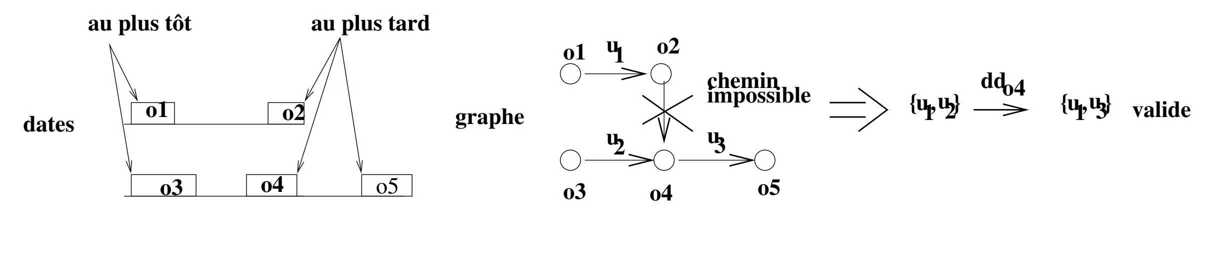


FIG. III.11 – Existence d'un déplacement vers la droite valide

III.2.5.c) Génération de la clique maximale dominante initiale

Le principe de génération de la clique maximale dominante initiale $\mathcal{C}_0$ est de partir de la clique maximale initiale gauche $\mathcal{C}_{init}^{\mathcal{R}_{ij}}$ dont la date de début au plus tôt est $r_{\mathcal{C}_{init}^{\mathcal{R}_{ij}}} = 0$, puis d'augmenter le plus faiblement possible cette date de début en déplaçant la clique vers la droite autour de l'opération destination de plus petite date de fin au plus tôt tant que cette date est inférieure ou égale à la date de début au plus tôt de (i,j).

La procédure est telle que :

- la condition d'arrêt se vérifie au bout d'un nombre fini d'itérations, chaque déplacement effectué étant faisable (proposition 1).
- La clique maximale obtenue est compatible avec l'opération (i,j) (proposition 2)
- La clique maximale obtenue domine un ensemble de cliques maximales dont on donne une caractérisation (proposition 3)

La procédure GÉNÈRECLIQUEMAXINIT est décrite dans la figure III.12. Les trois propositions sont énoncées et démontrées dans la suite du paragraphe.

Proc GÉNÈRECLIQUEMAXINIT $((i,j), l_{ij}, r_{ij}, \mathcal{G}, \mathcal{C}_0)$
 $\mathcal{C}_0 \leftarrow \mathcal{C}_{init}^{\mathcal{R}_{ij}}$
 $cpt \leftarrow 0$
 Tantque $(\min_{u_\alpha \in \mathcal{C}_{cpt}} r_{dest_\alpha} + p_{dest_\alpha} \leq r_{i,j})$ et $(\mathcal{C}_{cpt} \neq \mathcal{C}_{fin}^{\mathcal{R}_{ij}})$ **faire**
  $cpt \leftarrow cpt + 1$
  Soit l'opération $o_{dest} = dest_\alpha$ telle que $r_{dest_\alpha} + p_{dest_\alpha} = \min_{u_\beta \in \mathcal{C}_{cpt-1}} (r_{dest_\beta} + p_{dest_\beta})$
  $\mathcal{C}_{cpt} = dd_{o_{dest}}(\mathcal{C}_{cpt-1})$
 FinTantque
 $\mathcal{C}_0 \leftarrow \mathcal{C}_{cpt}$
FinProc

FIG. III.12 – Algorithme de GÉNÈRECLIQUEMAXINIT

Proposition 1 *La procédure* GÉNÈRECLIQUEMAXINIT *génère une clique maximale* $\mathcal{C}_0$ *telle que* $\forall u_\alpha \in \mathcal{C}_0$, $r_\alpha \leq r_{ij}$ *et* $r_{dest_\alpha} + p_{dest_\alpha} > r_{ij}$ *en un nombre fini d'itérations* *en parcourant un ensemble de cliques maximales contenant successivement tous les arcs ressources $u_\beta \in Gu_{\mathcal{R}}$ tels que $r_\beta \leq r_{i,j}$.*

Preuve
D'après le théorème 8, le déplacement vers la droite autour de l'opération o_{dest} est valide car elle possède la plus petite date de fin au plus tôt parmi les opérations destination des arcs de $\mathcal{C}_{cpt-1}$. $dd_{o_{dest}}(\mathcal{C}_{cpt-1})$ est donc une clique. D'après le théoreme 5, il s'agit aussi d'une clique maximale sur l'ensemble $\mathcal{R}_{ij}$.

Soit $\mathcal{O}_{min}(\mathcal{C}_{cpt-1})$ l'ensemble des opérations destination des arcs de $\mathcal{C}_{cpt-1}$ de plus petite date de fin au plus tôt. Par définition $\mathcal{C}_{cpt} = (\mathcal{C}_{cpt-1} \setminus \mathcal{U}^{\mathcal{R}_{ij}}_{entr}(o_{dest})) \cup \mathcal{U}^{\mathcal{R}_{ij}}_{sort}(o_{dest})$.

$\forall u_\beta \in \mathcal{U}^{\mathcal{R}_{ij}}_{sort}(o_{dest})$, $\forall u_\alpha \in \mathcal{U}^{\mathcal{R}_{ij}}_{entr}(o_{dest})$, on a : $r_{dest_\beta} + p_{dest_\beta} > r_{dest_\alpha} + p_{dest\alpha}$. D'où, si $\mathcal{O}_{min}(\mathcal{C}_{cpt})$ ne contient qu'une seule opération, $\min\limits_{u_\beta \in \mathcal{C}_{cpt-1}} (r_{dest_\beta} + p_{dest_\beta}) < \min\limits_{u_\beta \in \mathcal{C}_{cpt}} (r_{dest_\beta} + p_{dest_\beta})$ (1). Sinon, une autre opération de $\mathcal{O}_{min}(\mathcal{C}_{cpt-1})$ est sélectionnée pour la génération de $\mathcal{C}_{cpt+1}$ et la relation (1) sera vérifiée au bout de $|\mathcal{O}_{min}(\mathcal{C}_{cpt-1})|$ itérations.

Le graphe $\mathcal{G}$ étant sans cycle, tout arc supprimé ne sera jamais inclus à nouveau dans la clique d'exploration. Ce parcours est en fait un parcours exhaustif classique (voir par exemple [Prins, 1994]) des opérations o du graphe potentiels-tâches $\mathcal{G}$ telles que $r_o + p_o \leq r_{ij}$ ainsi que des arcs de type ressource issus de ces opérations suivant un front représenté par la clique courante $\mathcal{C}_{cpt}$. La procédure génère des cliques maximales $\mathcal{C}_{cpt}$ de $\min\limits_{u_r \in \mathcal{C}_{cpt}} (r_{dest_r} + p_{dest_r})$ croissant, ce qui prouve que la condition d'arrêt sera vérifiée en un nombre fini d'opérations borné supérieurement par le nombre d'arcs de type ressource $u_\alpha \in Gu_{\mathcal{R}}$ tels que $r_\alpha \leq r_{ij}$ □

Proposition 2 *La clique maximale $\mathcal{C}_0$ obtenue par la procédure* GÉNÈRECLIQUEMAXINIT *est compatible avec (i,j)*

Preuve Comme $\forall u_\alpha \in \mathcal{C}_0$, $r_{dest_\alpha} + p_{dest_\alpha} > r_{i,j}$ et $r_\alpha \leq r_{i,j}$, il ne peut exister de chemin ni entre une suivante de (i,j) et l'opération $orig_\alpha$, ni entre l'opération $dest_\alpha$ et une précédente de (i,j). □

Proposition 3 *La clique maximale $\mathcal{C}_0$ obtenue par la procédure* GÉNÈRECLIQUEMAXINIT *telle que $\forall u_\alpha \in \mathcal{C}_0$, $r_{dest_\alpha} + p_{dest_\alpha} > r_{i,j}$ et $r_{orig_\alpha} + p_{orig_\alpha} \leq r_{i,j}$ domine l'ensemble des cliques maximales $\mathcal{C}$ différentes de $\mathcal{C}_0$ telles que $r_{\mathcal{C}} \leq r_{i,j}$ et telles que $\forall u_p \in \mathcal{C}$, $\exists u_q \in \mathcal{C}_0$, $r_q \geq r_p$ qu'on appelle cliques maximales situées à gauche de $\mathcal{C}_0$.*

Preuve Soit $\mathcal{PI}(\mathcal{C}_0)$ la position d'insertion maximale définie par $\mathcal{C}_0$. Soit une clique suffisante $\mathcal{C}$. L'impact de l'insertion dans toute position $PI \subset \mathcal{PI}(\mathcal{C})$ sur l'admissibilité s'exprime par :

$$\Delta d_{max}(i,j,\mathcal{PI}) = \max(0, \max(r_{ij}, \max_{p(k,l) \in \mathcal{PI}} (r_{p(k,l)} + p_{p(k,l)})) + p_{ij}$$
$$- \min(d_{ij}, \min_{f(k,l) \in \mathcal{PI}} (d_{f(k,l)} - p_{f(k,l)})))$$

Si $\mathcal{C}$ est telle que $r_{\mathcal{C}} \leq r_{ij}$, $\forall (k,l,p(k,l),f(k,l)) \in \mathcal{PI}$, $r_{p(k,l)} + p_{p(k,l)} \leq r_{ij}$. Donc le terme $\max(r_{ij}, \max\limits_{p(k,l) \in \mathcal{PI}} (r_{p(k,l)} + p_{p(k,l)}))$ est égal à r_{ij}. Soit maintenant $\mathcal{C}$ une clique maximale différente de $\mathcal{C}_0$ et telle que $r_{\mathcal{C}} \leq r_{ij}$. Soit $\mathcal{PI} \subset \mathcal{PI}(\mathcal{C})$. Soit $\mathcal{PI}_0 \subset \mathcal{PI}(\mathcal{C}_0)$ contenant le même nombre de 4-uplets que $\mathcal{PI}$ et telle que $\forall k \in \mathcal{R}_{ij}$, $\forall l \in 1, ..., E_k$ si $(k, l, p(k,l), f(k,l)) \in \mathcal{PI}$ alors $\exists((k, l, p_0(k,l), f_0(k,l)) \in \mathcal{PI}_0$.

Si $\exists(k, l, p(k,l), f(k,l)) \in \mathcal{PI}$ et $(k, l, p(k,l), f(k,l)) \notin \mathcal{PI}_0$ alors pour exprimer la non compatibilité des arcs, il existe un chemin entre $f(k,l)$ et $p_0(k,l)$ tel que $(k, l, p_0(k,l), f_0(k,l)) \in \mathcal{PI}(\mathcal{C}_0)$. Il ne peut en effet exister de chemin entre $f_0(k,l)$ et $p(k,l)$ car dans ce cas on aurait $r_{\mathcal{C}} > r_{ij}$. Ceci implique que $d_{f(k,l)} - p_{f(k,l)} < d_{f_0(k,l)} - p_{f_0(k,l)}$ et donc que $\Delta d_{max}(i, j, \mathcal{PI}) < \Delta d_{max}(i, j, \mathcal{PI}_0)$.

Pour toute position d'insertion incluse dans $\mathcal{PI}(\mathcal{C})$ et non incluse dans $\mathcal{PI}(\mathcal{C}_0)$, il existe ainsi une position d'insertion incluse dans $\mathcal{PI}(\mathcal{C}_0)$ d'impact inférieur sur l'admissibilité. □

III.2.5.d) Procédure d'exploration des cliques maximales dominantes

Le principe de l'exploration des cliques maximales dominantes est de partir de la clique maximale dominante initiale $\mathcal{C}_0$ dont la date de fin au plus tard est $d_{\mathcal{C}_0}$, puis d'augmenter le plus faiblement possible cette date de fin en déplaçant la clique vers la droite autour de l'opération destination de plus petite date de début au plus tard tant que la date de fin au plus tard de la clique est strictement inférieure à la date de fin au plus tard de (i, j).

La procédure est telle que :

- la condition d'arrêt se vérifie au bout d'un nombre fini d'itération, chaque déplacement effectué étant faisable (proposition 4).
- La clique maximale obtenue est compatible avec l'opération (i, j) (proposition 5).
- La clique maximale obtenue domine un ensemble de cliques maximales dont on donne une caractérisation (proposition 6).
- L'ensemble des cliques maximales parcourues est dominant (proposition 7).

La procédure ExplCliquesMaxDomin est décrite dans la figure III.13. Les quatre propositions sont énoncées et démontrées dans la suite du paragraphe.

Proposition 4 *La procédure* ExplCliquesMaxDomin *génère une clique maximale $\mathcal{C}_K$ telle que $\forall u_\alpha \in \mathcal{C}_K$, $d_{orig_\alpha} - p_{orig_\alpha} < d_{ij}$ et $d_{\mathcal{C}_K} \geq d_{ij}$ en un nombre fini d'itérations en parcourant un ensemble de cliques maximales contenant successivement tous les arcs de type ressource $u_q \in Gu_{\mathcal{R}}$ à partir de $\mathcal{C}_0$ tels que $d_{dest_q} - p_{dest_q} < d_{i,j}$*

Preuve Identique à la preuve de la proposition 1. □

Proposition 5 *La clique maximale $\mathcal{C}_K$ obtenue par la procédure* ExplCliquesMaxDomin *est compatible avec (i, j).*

Preuve Symétrique de la preuve de la proposition 2. □

Proc ExplCliquesMaxDomin $((i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{G})$
    GénèreCliqueMaxInit $((i,j), l_{ij}, r_{ij}, \mathcal{G}, \mathcal{C}_0)$
    $cpt \leftarrow 0$
    Tantque $\mathcal{C}_{cpt} \neq \mathcal{C}_{fin}^{\mathcal{R}_{ij}}$ et $d_{\mathcal{C}} < d_{i,j}$ **faire**
        $cpt \leftarrow cpt + 1$
        Soit $o_{dest} = dest_\alpha$ telle que $d_\alpha = \min\limits_{u_\beta \in \mathcal{C}_{cpt-1}} d_\beta$
        $\mathcal{C}_{cpt} = dd_{o_{dest}}(\mathcal{C}_{cpt-1})$
    FinTantque
FinProc

Fig. III.13 – Algorithme de ExplCliquesMaxDomin

Proposition 6 *La clique maximale $\mathcal{C}_K$ obtenue par la procédure* ExplCliquesMaxDomin *telle que $d_{\mathcal{C}_K} \geq d_{i,j}$, $d_{\mathcal{C}_{K-1}} < d_{i,j}$ domine l'ensemble des cliques maximales $\mathcal{C}$ telles que $d_{\mathcal{C}} \geq d_{i,j}$, qu'on appelle cliques situées à droite de $\mathcal{C}_K$.*

Preuve Symétrique de la preuve de la proposition 3. □

Proposition 7 *Lorsqu'un arc est supprimé de la clique courante $\mathcal{C}_q$ par la procédure, toute clique le contenant est dominée par une sous clique de $\mathcal{C}_q$ ou par une clique déja visitée.*

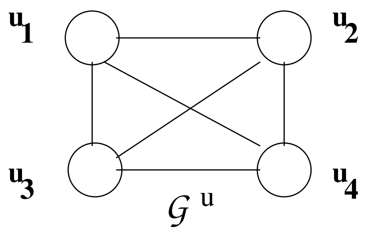
Preuve (Démonstration par récurrence)
Soit au rang 0 un arc $u_\alpha \in \mathcal{C}_0$ tel que $d_\alpha = \min\limits_{u_\beta \in \mathcal{C}_0} d_\beta$. Soit une clique $\mathcal{C}_s$ non incluse dans $\mathcal{C}_0$ et contenant u_α. Puisque $u_\alpha \in \mathcal{C}_s$, $d_{\mathcal{C}_s} \leq d_{\mathcal{C}_0}$. De plus puisque $r_{\mathcal{C}_0} \leq r_{ij}$, $\Delta d_{max}(i,j,\mathcal{C}_s) \geq \Delta d_{max}(i,j,\mathcal{C}_0)$. La proposition est vraie au rang 0. Admettons que la proposition est vraie pour tout rang $\leq q$. Soit au rang $q+1$ un arc $u_\alpha \in \mathcal{C}_{q+1}$ tel que $d_\alpha = \min\limits_{u_\beta \in \mathcal{C}_{q+1}} d_\beta$. Soit une clique $\mathcal{C}_s$ non incluse dans $\mathcal{C}_{q+1}$ et contenant u_α. Puisque $u_\alpha \in \mathcal{C}_s$, $d_{\mathcal{C}_s} \leq d_{\mathcal{C}_{q+1}}$. Pour obtenir Δd_{max} plus faible que celui de $\mathcal{C}_{q+1}$ on ne peut agir que sur le terme des dates de début au plus tôt. Soit u_α un arc non contenu dans $\mathcal{C}_{q+1}$ et tel que $r_\alpha < r_{\mathcal{C}_{q+1}}$. Cet arc a été déjà supprimé à un rang $q' < q+1$ et toute clique le contenant est dominée par $\mathcal{C}_{q'}$ ou par une clique déja visitée. □

III.2.6 Recherche de la meilleure sous-clique

Pour trouver la sous-clique suffisante d'impact minimal sur l'admissibilité, une procédure d'énumération est envisagée en III.2.6.a). Des règles de dominance énoncées en III.2.6.b) permettent de définir une procédure d'exploration dont l'algorithme est polynomial présentée en III.2.6.c). Dans le paragraphe III.2.6.d) on étudie l'influence de la durée de l'opération à insérer sur l'exploration.

III.2.6.a) Enumeration explicite des sous-cliques

Soit une clique à q arcs. Un telle clique contient 2^q sous-cliques. L'énumération explicite des sous-cliques a été définie dans [Le Gall, 1989] dans un contexte cumulatif et reprise dans [Billaut, 1993] dans le contexte étudié dans ce paragraphe. Cette énumération, basée sur un algorithme de recherche d'une clique maximale décrit dans [Esquirol, 1987], est en $\mathcal{O}(2^q)$.



arc	r	d	l_α
u_1	16	26	{2,1}
u_2	14	24	{2,1}
u_3	12	28	{1,1}
u_4	7	19	{1,3}

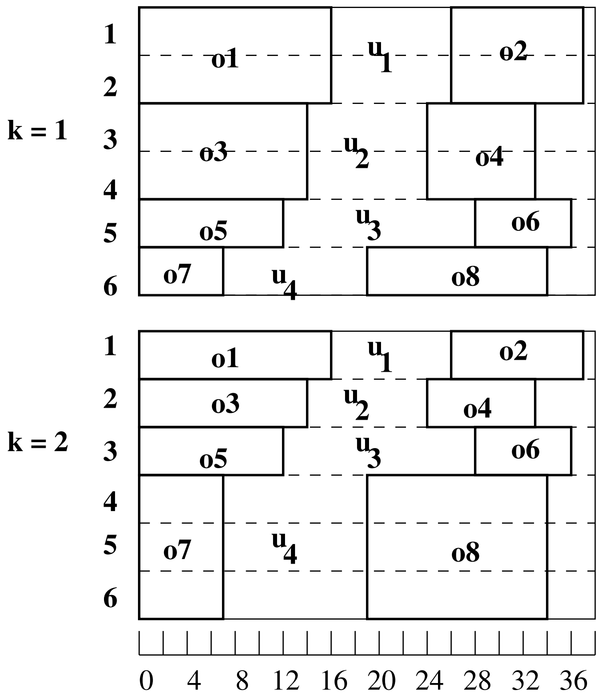


FIG. III.14 – Clique d'occupation maximale sur deux ressources

Par exemple, soit la clique d'occupation des ressources maximale C_0 représentée dans la figure III.14. L'opération (i,j) à insérer est telle que $l_{i,j} = (2,2)$, $p_{ij} = 12$, $r_{ij} = 0$ et $d_{ij} = 30$. Pour trouver la meilleure sous clique, on devrait explorer au maximum 16 sous-cliques.

III.2.6.b) Règles de dominance entre cliques

Des relations d'ordre entre arcs ressource induites par les dates de début au plus tôt d'une part, et les dates de fin au plus tard d'autre part, sont à la base des règles de dominance pour la recherche d'une position d'insertion suffisante optimale décrites ici. Elles s'inspirent de règles de dominance entre ensembles d'intervalles déterminées dans [Torres, 1996].

Soit une clique $\mathcal{C}$ d'ordre q. Soit l'ensemble $\mathcal{U}_{\max}$ des arcs de $\mathcal{C}$ de plus grande date de début au plus tôt : $\forall u_\alpha \in \mathcal{U}_{\max}$, $r_\alpha = \max_{\beta=1,\ldots,q} r_\beta$.
Soit l'ensemble $\mathcal{U}_{\min}$ des arcs de $\mathcal{C}$ de plus petite date de fin au plus tard : $\forall u_\alpha \in \mathcal{U}_{\min}$, $d_\alpha = \min_{\beta=1,\ldots,q} d_\beta$.

Théorème 9 *Une clique $\mathcal{C}$ suffisante pour l'insertion d'une opération (i,j) est dominée par une sous clique $\mathcal{C}_s \subset \mathcal{C}$ si et seulement si : (A) $l_{\mathcal{C}_s} \supseteq l_{ij}$ et (B) soit $\mathcal{C}_s$ ne contient aucun arc de $\mathcal{U}_{\max}$ (C) soit $\mathcal{C}_s$ ne contient aucun arc de $\mathcal{U}_{\min}$.*

Preuve :

Si (A) est vérifiée, alors $\mathcal{C}_s$ est aussi une clique suffisante pour l'insertion de (i,j).

- Si (B) est vérifiée alors on peut affirmer que $r_{\mathcal{C}_s} < r_{\mathcal{C}}$ car $\forall u_\alpha \in \mathcal{U}_{\max}$, $r_{\mathcal{C}} = r_\alpha$. Si $\mathcal{U}_{\max} \neq \mathcal{U}_{\min}$ alors $d_{\mathcal{C}_s} = d_{\mathcal{C}}$. Sinon, $d_{\mathcal{C}_s} > d_{\mathcal{C}}$ car $\forall u_\alpha \in \mathcal{U}_{\min}$, $d_{\mathcal{C}} = d_\alpha$ et $\mathcal{C}_s$ ne contient pas non plus d'arcs de $\mathcal{U}_{\min}$.
- Si $\mathcal{C}_s$ ne contient aucun arc de $\mathcal{U}_{\min}$ (C) alors on peut affirmer que $d_{\mathcal{C}_s} > d_{\mathcal{C}}$ car $\forall u_\alpha \in \mathcal{U}_{\min}, d_{\mathcal{C}} = d_\alpha$. Si $\mathcal{U}_{\min} \neq \mathcal{U}_{\max}$ alors $r_{\mathcal{C}_s} = r_{\mathcal{C}}$. Sinon $r_{\mathcal{C}_s} < r_{\mathcal{C}}$ car $\forall u_\alpha \in \mathcal{U}_{\max}, r_{\mathcal{C}} = r_\alpha$ et $\mathcal{C}_s$ ne contient pas non plus d'arcs de $\mathcal{U}_{\max}$.

Si l'hypothèse (A) est valide et si l'hypothèse (B) ou (C) est valide, on a bien $\mathcal{C}_s$ suffisante et dans les deux cas $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s)) \leq \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}))$.

Considérons que $\mathcal{C}_s \subset \mathcal{C}$ domine $\mathcal{C}$. Ceci implique directement (A) car une clique dominante est obligatoirement suffisante (voir définition 8). De plus, d'après cette définition, on a $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s)) \leq \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}))$, ce qui ne peut être vérifié que si une des hypothèses (B) ou (C) est vérifiée. □

III.2.6.c) Procédure d'exploration des sous-cliques dominantes

A partir des règles de dominance exprimées dans le théoreme 9, une procédure d'exploration des sous-cliques dominantes alternative à l'énumération explicite des sous-cliques est proposée.

On définit un arbre d'exploration dont chaque noeud est une sous-clique dominante. Le noeud initial contient la clique suffisante de départ. A chaque noeud s on associe ainsi la clique $\mathcal{C}_s$ et les ensembles $\mathcal{U}_{max,s}$ et $\mathcal{U}_{min,s}$. A partir du noeud s on génère le noeud $s1$ appelé “fils gauche” de s correspondant à la clique obtenue en enlevant de $\mathcal{C}_s$ tous les arcs de $\mathcal{U}_{max,s}$ et le noeud $s2$ appelé “fils droit” de s correspondant à la clique obtenue en enlevant de $\mathcal{C}_s$ tous les arcs de $\mathcal{U}_{min,s}$. La clique $\mathcal{C}_{s1}$ domine la clique $\mathcal{C}_s$ sauf si $l_{\mathcal{C}_{s1}} \not\supseteq l_{ij}$. De même, la clique $\mathcal{C}_{s2}$ domine la clique $\mathcal{C}_s$ sauf si $l_{\mathcal{C}_{s2}} \not\supseteq l_{ij}$. On peut remarquer d'autre part que le fils gauche $s21$ de $s2$ est confondu avec le fils droit $s12$ de $s1$ puisque $s21$ et $s12$ correspondent à la clique incluse dans $\mathcal{C}_s$ ne contenant ni $\mathcal{U}_{max,s}$ ni $\mathcal{U}_{min,s}$. On peut ainsi décomposer l'arbre d'exploration en niveaux ν et ne générer que les fils gauche de chaque noeud à l'exception du dernier fils droit de chaque niveau, dont les deux fils doivent être générés. On cesse de développer un noeud lorsque la clique associée ne contient qu'un seul arc puisqu'on ne peut lui trouver aucune sous-clique dominante ou lorsque la clique associée a une occupation des ressources qui n'inclut plus au sens strict l'occupation des ressources de l'opération. En effet, toute sous-clique est dans ce cas insuffisante. De plus lorsque sur un noeud donné $\mathcal{U}_{max,s} = \mathcal{U}_{min,s}$ on a une seule sous-clique dominante. Dans ce cas là, le fils gauche du noeud suivant n'a pas besoin d'être développé puisqu'on retrouverait le fils gauche du noeud précédent. Cette propriété permet de réduire encore le nombre de sous-cliques visitées.

L'algorithme de ExplSousCliquesDomin décrit dans la figure III.15 permet de réaliser l'exploration des sous-cliques dominantes selon les principes énoncés.

Proc ExplSousCliquesDomin $(\mathcal{C}, (i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{C}_{opt}, \Delta d_{max}^{opt})$
  $\mathcal{C}^{max} \leftarrow \mathcal{C}$ triée par ordre décroissant des r_α
  $\mathcal{C}^{min} \leftarrow \mathcal{C}$ triée par ordre croissant des d_α
  $\mathcal{L}_1 \leftarrow \{(\mathcal{C}^{\max}, \mathcal{C}^{\min}, l_{\mathcal{C}})\}$
  $\Delta d_{max}^{opt} \leftarrow \infty$; $\nu \leftarrow 1$; $STOP \leftarrow FAUX$
  Tantque $(\mathcal{L}_\nu \neq \emptyset)$ et $(STOP = FAUX)$ **faire**
    $\mathcal{L}_{\nu+1} \leftarrow \emptyset$; $s' \leftarrow \emptyset$
    Pour chaque noeud $s = (\mathcal{C}_s^{max}, \mathcal{C}_s^{min}, l_{\mathcal{C}_s}) \in \mathcal{L}_\nu$ et **Tantque** $STOP = FAUX$ **faire**
      Enlever $(\mathcal{C}_s^{max}, \mathcal{C}_s^{min}, l_{\mathcal{C}_s})$ de $\mathcal{L}_\nu$.
      Déterminer $\mathcal{U}_{\max,s}$ et $\mathcal{U}_{\min,s}$ et $l_{\mathcal{C}_s}$
      Si $l_{\mathcal{C}_s} \supseteq l_{ij}$ **alors**
        Calculer $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s))$
        Si $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s)) < \Delta d_{max}^{opt}$ **alors**
          $\Delta d_{max}^{opt} \leftarrow \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s))$
          $\mathcal{C}_{opt} \leftarrow \mathcal{C}_s$
          Si $\Delta d_{max}^{opt} = 0$ **alors**
            $STOP \leftarrow VRAI$
          FinSi
        FinSi
        Génération du fils gauche s_1
        Si $\mathcal{U}_{\max,s'}$ et $\mathcal{U}_{\min,s'}$ du noeud précédent sont distincts **alors**
          $\mathcal{C}_{s1}^{max} \leftarrow \mathcal{C}_s^{max} \setminus \mathcal{U}_{max,s}$
          $\mathcal{C}_{s1}^{min} \leftarrow \mathcal{C}_s^{min} \setminus \mathcal{U}_{max,s}$
          $\forall u_\alpha \in \mathcal{U}_{max,s}$, $l_{\mathcal{C}_{s1}} \leftarrow l_{\mathcal{C}_s} \setminus l_\alpha$
          ajouter $(\mathcal{C}_{s1}^{max}, \mathcal{C}_{s1}^{min}, l_{\mathcal{C}_{s1}})$ à la fin de la liste $\mathcal{L}_{\nu+1}$
        FinSi
        Si s est le dernier de élément de $\mathcal{L}_\nu$ **alors**
          Génération du fils droit s_2
          $\mathcal{C}_{s2}^{max} \leftarrow \mathcal{C}_s^{max} \setminus \mathcal{U}_{\min,s}$
          $\mathcal{C}_{s1}^{min} \leftarrow \mathcal{C}_s^{min} \setminus \mathcal{U}_{\min,s}$
          $\forall u_\alpha \in \mathcal{U}_{\min,s}$, $l_{\mathcal{C}_{s2}} \leftarrow l_{\mathcal{C}_s} \setminus l_\alpha$
          ajouter $(\mathcal{C}_{s2}^{max}, \mathcal{C}_{s2}^{min}, l_{\mathcal{C}_{s2}})$ à la fin de la liste $\mathcal{L}_{\nu+1}$
        FinSi
      FinSi
    $s' \leftarrow s$
    FinPour
    $\nu \leftarrow \nu + 1$
  FinTantque
FinProc

Fig. III.15 – *Algorithme de* ExplSousCliquesDomin

On peut déterminer une borne supérieure du nombre de cliques dominantes parcourues par cette procédure. Chaque niveau ν correspond à l'enlèvement dans les cliques du niveau précédent d'au moins un arc. Pour une clique contenant q arcs, on ne peut avoir que q niveaux. De plus, à chaque niveau ν on ajoute un noeud par rapport au niveau précédent puisque seul le dernier fils droit a deux fils, tous les autres noeuds ne possédant qu'un fils. Le niveau $\nu = 1$ ayant 1 noeud, on a ainsi au maximum $\sum_{\nu=1}^{q} \nu = \frac{q(q+1)}{2}$ noeuds parcourus. Des suppressions et insertions dans des listes étant effectuées à chaque noeud, la complexité de l'algorithme est en $\mathcal{O}(q^3)$.

La figure III.16 montre l'arbre de parcours des cliques-dominantes pour l'exemple de la figure III.14. Les branches représentées en tirets correspondent aux fils dominants mais non générés. Les branches représentées en lignes continues correspondent aux fils dominants générés. W est la largeur de la clique correspondant au noeud.

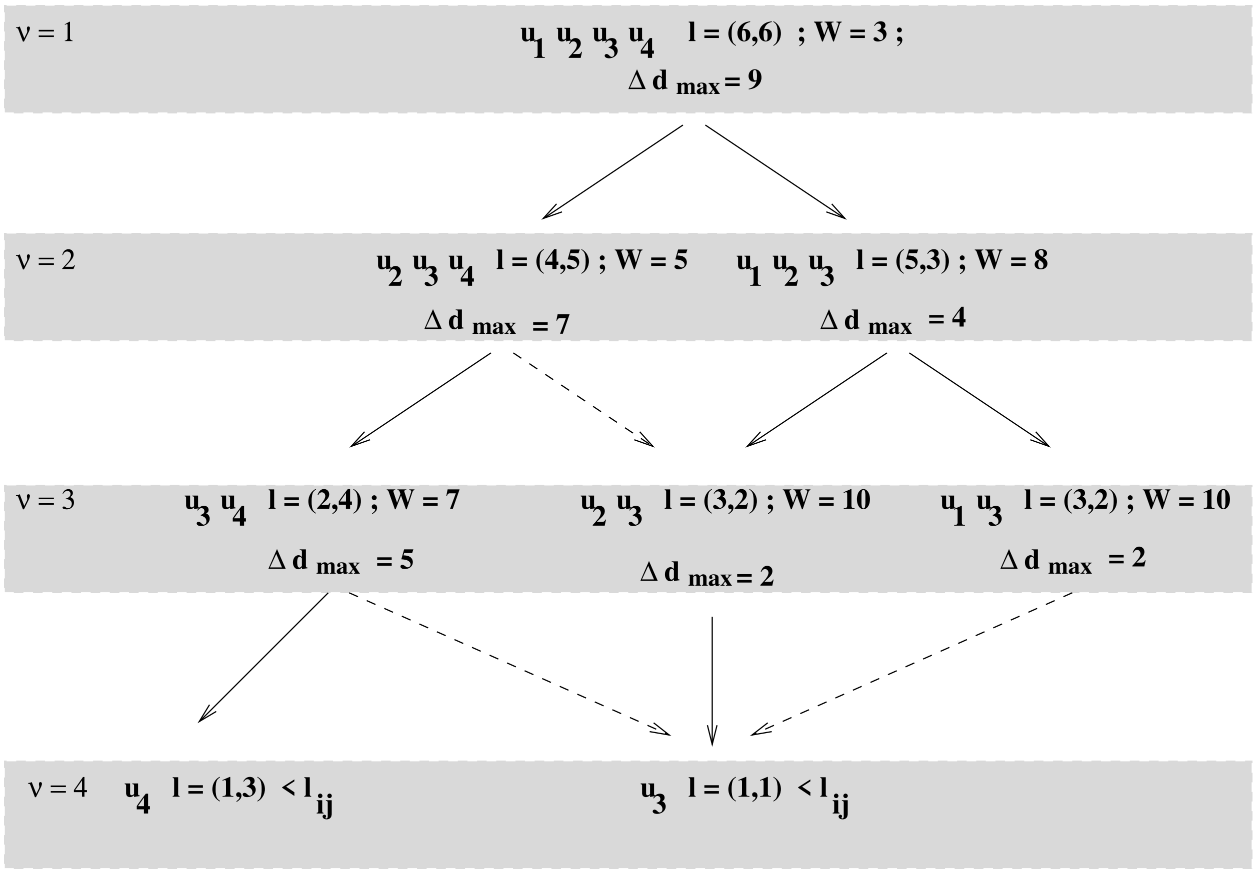


FIG. III.16 – Exploration des sous-cliques dominantes

Si l'arbre ne contenait que des noeuds s tels que $\mathcal{U}_{max,s} \neq \mathcal{U}_{min,s}$ il aurait 10 noeuds. Or les cliques $\{u_2, u_3\}$ et $\{u_1, u_3\}$ sont telles que $\mathcal{U}_{\max} = \mathcal{U}_{\min} = \{u_3\}$ ce qui ramène l'arbre à 8 noeuds. Les cliques minimisant l'impact sur l'admissibilité sont ici $\{u_2, u_3\}$ et $\{u_1, u_3\}$. On explore dans ce cas deux fois moins de noeuds que la procédure d'énumération explicite pour une clique initiale à 4 arcs.

III.2.6.d) Influence de la durée de l'opération

On peut remarquer que l'arbre d'exploration est le même quelle que soit la durée de l'opération à insérer. Toutefois, comme on recherche un impact minimal sur l'admissibilité, l'exploration peut s'arréter plus rapidement si la durée de l'opération est plus petite. La largeur de la clique W donne la durée maximale qu'une opération peut avoir sans décaler les opérations suivantes. Prenons $p_{ij} = 8$. La clique $\{u_1, u_2, u_3\}$ de largeur $W = 8$ donne $\Delta d_{max} = 0$. L'exploration s'arrète ainsi au niveau 2.

III.2.7 Intégration des deux procédures

La procédure CliqueOpt dont l'algorithme est décrit dans la figure III.17 intègre les procédures ExplCliquesMaxDomin et ExplSousCliquesDomin. Pour chaque clique maximale dominante générée par l'exploration selon le principe de ExplCliquesMax-Domin , la procédure ExplSousCliquesDomin est appelée pour trouver la sous-clique optimale. Une condition d'arrêt supplémentaire est ajoutée lors de l'exploration des sous-cliques d'une clique maximale dominante $\mathcal{C}_{cpt}$: la disparition dans la sous-clique courante des arcs ajoutés à $\mathcal{C}_{cpt-1}$ par le déplacement vers la droite. En effet, les sous-cliques ne contenant pas ces arcs ont déjà été explorées lors de la recherche de la sous-clique optimale de $\mathcal{C}_{cpt-1}$. L'algorithme de CliqueOpt est en $\mathcal{O}(q^3 u M_o^2)$ où q est le nombre maximal d'arcs dans une clique, u le nombre d'arcs ressources du graphe (non généré) $\mathcal{G}u_{\mathcal{R}_{ij}}$ et M_o le nombre maximum de ressources occupées par une opération. A partir de la clique suffisante optimale trouvée $\mathcal{C}_{opt}$ d'impact Δd_{max}^{opt} sur l'admissibilité, l'utilisation de la procédure InsèreClique permet d'effectuer l'insertion.

III.2.8 Exemple complet

Soit une opération (i, j) de durée $p_{ij} = 4$, de date de début au plus tôt $r_{ij} = 5$, de date de fin au plus tard $d_{ij} = 15$ et d'occupation des ressources l_{ij} telle que $e_{ij}^R = 2$. R est une ressource cumulative à 3 lignes de charge. Les opérations ordonnancées sur R sont représentées dans la figure III.18. En haut de la figure se trouve une représentation partielle du graphe $\mathcal{G}$. Seules les opérations ordonnancées sur R et les arcs ressources les liant sont représentés. Au milieu de la figure, un diagramme de Gantt des opérations de R calées au plus tôt donne les dates de début au plus tôt des opérations déja ordonnancées sur $\mathcal{G}$. En bas de la figure, un diagramme de Gantt des opérations de R calées au plus tard donne les dates de début au plus tard de ces opérations.

Les cliques explorées par GénèreCliqueMaxInit sont les cliques suivantes :

$\mathcal{C}_0 = \{u_{Oa}, u_{Ob}\}$, $\min\limits_{u_\alpha \in \mathcal{C}_0} r_{dest_\alpha} + p_{dest_\alpha} = 2$, puis $\mathcal{C}_1 = \{u_{ab}, u_{Ob}\}$, $\min\limits_{u_\alpha \in \mathcal{C}_1} r_{dest_\alpha} + p_{dest_\alpha} = 4$ et enfin $\mathcal{C}_2 = \{u_{bc}, u_{bd}, u_{be}\}$ avec $\min\limits_{u_\alpha \in \mathcal{C}_2} r_{dest_\alpha} + p_{dest_\alpha} = 6 > r_{ij}$. L'exploration débute ainsi avec la clique initiale dominante $\mathcal{C}_0 = \{u_{bc}, u_{bd}, u_{be}\}$.

Les cliques explorées par ExplCliquesMaxDomin et ExplSousCliquesDomin sont les cliques suivantes :

$cpt = 0$ $\mathcal{C}_0 = \{u_{bc}, u_{bd}, u_{be}\}$, $d_{\mathcal{C}_0} = 6$, $\Delta d_{max}(i, j, \mathcal{PI}(\mathcal{C}_0)) = 3$
Fils gauche $\mathcal{C}_{01} = \emptyset$
Fils droit $\mathcal{C}_{02} = \{u_{bc}, u_{bd}\}$, $e_{\mathcal{C}_{02}}^R = 2$, $\Delta d_{max}(i, j, \mathcal{PI}(\mathcal{C}_{02})) = 2$

Proc CliqueOpt $((i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{G}, \mathcal{C}_{opt}, \Delta d_{max}^{opt})$
  GénèreCliqueMaxInit $((i,j), l_{ij}, r_{ij}, \mathcal{G}, \mathcal{C}_0)$
  ExplSousCliquesDomin $(\mathcal{C}_0, (i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{C}_{opt}, \Delta d_{max}^{opt})$
  Si $\Delta d_{max}^{opt} = 0$ **alors**
    $STOP \leftarrow VRAI$
  Sinon
    $STOP \leftarrow FAUX$
  FinSi
  $cpt \leftarrow 0$;
  Tantque $\mathcal{C}_{cpt} \neq \mathcal{C}_{fin}^{\mathcal{R}_{ij}}$ et $d_{\mathcal{C}} < d_{i,j}$ et $STOP = FAUX$ **faire**
    $cpt \leftarrow cpt + 1$
    Soit $o_{dest} = dest_\alpha$ telle que $d_\alpha = \min_{u_\beta \in \mathcal{C}_{cpt-1}} d_\beta$
    $\mathcal{C}_{cpt} = dd_{o_{dest}}(\mathcal{C}_{cpt-1})$
    ExplSousCliquesDominPart [1] $(\mathcal{C}_0, (i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o_{dest}), \mathcal{C}_{opt}(cpt), \Delta d_{max}^{opt}(cpt))$
    Si $\Delta d_{max}^{opt}(cpt) < \Delta d_{max}^{opt}$ **alors**
      $\Delta d_{max}^{opt} \leftarrow \Delta d_{max}^{opt}(cpt)$
      $\mathcal{C}_{opt} \leftarrow \mathcal{C}_{opt}(cpt)$
      Si $\Delta d_{max}^{opt} = 0$ **alors**
        $STOP \leftarrow VRAI$
      FinSi
    FinSi
  FinTantque
FinProc

[1] : ExplSousCliquesDominPart est identique à ExplSousCliquesDomin à la différence qu'un noeud correspondant à une clique ne contenant aucun arc de l'ensemble $\mathcal{U}_{sort}^{\mathcal{R}_{ij}}(o_{dest})$ n'est pas développé.

Fig. III.17 – Algorithme de CliqueOpt

$cpt = 1$ $\mathcal{C}_1 = \{u_{bc}, u_{bd}, u_{ef}\}$, $d_{\mathcal{C}_1} = 7$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_1)) = 4$
  Fils gauche $\mathcal{C}_{11} = \mathcal{C}_{01}$
  Fils droit $\mathcal{C}_{12} = \{u_{ef}\}$, $e_{\mathcal{C}_{12}}^R = 1 < e_{ij}^R$

$cpt = 2$ $\mathcal{C}_2 = \{u_{cg}, u_{df}, u_{ef}\}$, $d_{\mathcal{C}_2} = 9$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_2)) = 2$
  Fils gauche $\mathcal{C}_{21} = \{u_{df}\}$, $e_{\mathcal{C}_{21}}^R = 1 < e_{ij}^R$
  Fils droit $\mathcal{C}_{22} = \{u_{cg}\}$, $e_{\mathcal{C}_{22}}^R = 1 < e_{ij}^R$

$cpt = 3$ $\mathcal{C}_3 = \{u_{cg}, u_{fg}, u_{fh}\}$, $d_{\mathcal{C}_3} = 10$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_3)) = 2$
  Fils gauche $\mathcal{C}_{31} = \mathcal{C}_{21}$
  Fils droit $\mathcal{C}_{32} = \{u_{fh}\}$, $e_{\mathcal{C}_{32}}^R = 1 < e_{ij}^R$

$cpt = 4$ $\mathcal{C}_4 = \{u_{gk}, u_{gi}, u_{fh}\}$, $d_{\mathcal{C}_4} = 11$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_4)) = 3$
  Fils gauche $\mathcal{C}_{41} = \mathcal{C}_{31}$
  Fils droit $\mathcal{C}_{42} = \{u_{gk}, u_{gi}\}$, $e_{\mathcal{C}_{42}}^R = 2$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_{42})) = 1$

$cpt = 5$ $\mathcal{C}_5 = \{u_{gk}, u_{gi}, u_{hi}\}$, $d_{\mathcal{C}_5} = 13$, $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_5)) = 1$
  Fils gauche $\mathcal{C}_{51} = \emptyset$

Fils droit $\mathcal{C}_{52} \subset \mathcal{C}_4$

$cpt = 6$ $\mathcal{C}_6 = \{u_{gk}, u_{ik}, u_{ij}\}$, $d_{\mathcal{C}_6} = 16 > d_{ij}$ (**Stop**), $\Delta d_{max}(i, j, \mathcal{PI}(\mathcal{C}_6)) = 2$
Fils gauche $\mathcal{C}_{61} = \mathcal{C}_{41}$
Fils droit $\mathcal{C}_{62} = \emptyset$

La dernière clique optimale explorée est $\mathcal{C}_5 = \{u_{gk}, u_{gi}, u_{hi}\}$ d'impact sur l'admissibilité de 1 unité.

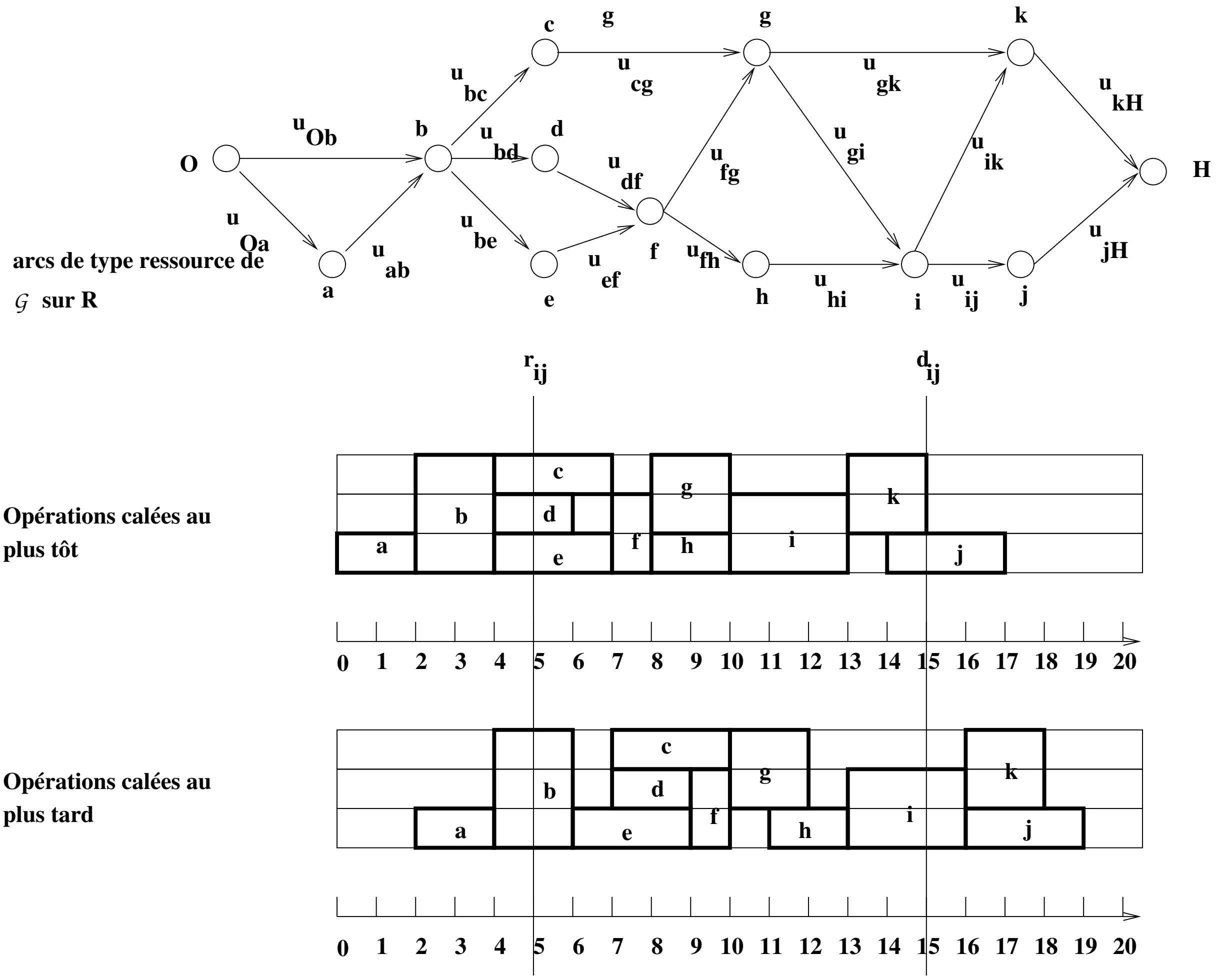


FIG. III.18 – Exemple d'ordonnancement sur une ressource cumulative

III.3 Prise en compte de l'affectation dans le problème d'insertion

III.3.1 Définition du problème

Dans le modèle présenté au paragraphe II.2, toute opération possède un ensemble de m_{ij} modes étendus. Pour chaque mode étendu m un ensemble de pools $\mathcal{P}_{ij}^m$ est défini.

Une ressource doit être sélectionnée dans chaque pool $P_{ij}^{mr} \in \mathcal{P}_{ij}^{m}$. Le nombre de lignes de charge e_{ij}^{k} que peut utiliser l'opération (i,j) sur une ressource k dans le mode étendu m, est compris entre une borne inférieure e_{ij-}^{mk} et une borne supérieure e_{ij+}^{mk}. La sélection d'un mode étendu, d'une ressource dans chaque pool du mode étendu sélectionné et d'un nombre de lignes de charge sur chaque ressource sélectionnée fixe l'occupation des ressources de l'opération. L'ensemble des occupations des ressources possibles correspondant à un ensemble donné de modes étendus peut être énuméré. L'occupation des ressources l_{ij} d'une opération (i,j) peut ainsi prendre $\sum_{m=1}^{m_{ij}} \prod_{r=1}^{M_{ij}^{m}} \sum_{k \in P_{ij}^{mr}} (e_{ij+}^{mk} - e_{ij-}^{mk} + 1)$ valeurs différentes. On remarque que ce nombre croit linéairement en fonction du nombre de modes étendus m_{ij} mais exponentiellement en fonction du nombre de pools M_{ij}^{m}. Soit $\mathcal{L}_{ij}^{m}$ l'ensemble des occupations des ressources possibles pour (i,j) dans le mode m. Une solution pour trouver une position d'insertion optimale associée à une opération (i,j) dans ce contexte est d'énumérer l'ensemble des occupations des ressources possibles et pour chacune d'elles, d'appliquer la procédure de recherche d'une clique suffisante optimale CliqueOpt présentée au paragraphe III.2. Pour chaque occupation des ressources l_{ij} on rappelle que la durée de l'opération $p_{ij} = p_{ij}^{mge_{ij}^{g}}$ dépend de la ressource g sélectionnée dans le pool guidant $Pg_{ij}^{m} \in \mathcal{P}_{ij}^{m}$ et du nombre de lignes de charge e_{ij}^{g} utilisé par (i,j) sur g. Une durée unique est ainsi associée à chaque occupation des ressources l_{ij} et on est bien dans le champ d'application de la procédure CliqueOpt. La durée associée à une occupation des ressources possible l_{ij} est notée $p_{ij}(l_{ij})$.

Dans les cas où le nombre de pools et de ressources dans chaque pool sont importants, il est primordial de limiter l'énumération. La procédure présentée ici permet de restreindre l'énumération aux seuls modes étendus. Elle est basée sur l'exploration des cliques maximales dominantes sur l'ensemble de toutes les ressources possibles pour l'opération dans le mode étendu considéré. Pour chacune de ces cliques maximales dominantes, l'exploration des sous-cliques dominantes est effectuée suivant le principe de ExplSousCliquesDomin sauf que le problème d'affectation est résolu localement à chaque noeud de l'arbre d'exploration.

III.3.2 Procédure optimale d'insertion à mode étendu fixé en alternative à l'énumération des occupations

Pour un mode étendu m fixé, une borne supérieure de l'occupation des ressources de l'opération l_{ij+} et une borne inférieure l_{ij-} peut être calculée :

$$l_{ij+} = \{e_{ij+}^{m1}, ..., e_{ij+}^{mM}\} \tag{III.12}$$
$$l_{ij-} = \{e_{ij-}^{m1}, ..., e_{ij-}^{mM}\} \tag{III.13}$$

On rappelle que e_{ij+}^{mk} (e_{ij-}^{mk}) est le nombre maximum (minimum) de lignes de charge utilisées par (i,j) sur la ressource k dans le mode étendu m.

Cette valeur est non nulle pour toute ressource k de chaque pool P_{ij}^{mr} où $r = 1, ..., M_{ij}^{m}$. Soit $\mathcal{R}_{ij}^{m+} = \bigcup_{r=1,...,M_{ij}^{m}} P_{ij}^{mr}$ l'ensemble de toutes les ressources utilisables par

(i,j) exécutée dans le mode étendu m. Toute occupation des ressources l_{ij} possible pour (i,j) dans le mode étendu m est telle que $l_{ij} \subseteq l_{ij+}$.

Soit une opération fictive (i,s) d'occupation des ressources l_{ij+} possédant les mêmes relations de précédence et de succession dans g_i que (i,j). Soit $\{\mathcal{C}_q^+\}$ l'ensemble des cliques maximales dominantes explorées par ExplCliquesMaxDomin appliquée à (i,s). Soit l_α une des valeurs possibles de l_{ij} dans le mode étendu m ($l_\alpha \in \mathcal{L}_{ij}^m$). Soit $\{\mathcal{C}_{q'}\}$ l'ensemble des cliques maximales dominantes explorées par ExplCliquesMaxDomin appliquée à (i,j) d'occupation des ressources l_α.

Théorème 10 *Toute clique de l'ensemble $\{\mathcal{C}_{q'}\}$ est incluse dans une clique de l'ensemble $\{\mathcal{C}_q^+\}$ et toute clique de l'ensemble $\{\mathcal{C}_q^+\}$ inclut une clique de l'ensemble $\{\mathcal{C}_{q'}\}$.*

Preuve (Démonstration par récurrence) La clique initiale $\mathcal{C}_{init}^{\mathcal{R}_{ij}^{m+}}$ dans la procédure GénèreCliqueMaxInit constituée des arcs de type ressource dont l'origine est le sommet O et d'occupation maximale des ressources sur $\mathcal{R}_{ij}^{m+}$ inclut bien la clique correspondante $\mathcal{C}_{init}^{\mathcal{R}_{ij}}$ constituée des arcs de type ressource dont l'origine est le sommet O et d'occupation maximale des ressources sur $\mathcal{R}_{ij} \subset \mathcal{R}_{ij}^{m+}$. Soit une certaine étape q de la procédure GénèreCliqueMaxInit appliquée à partir de $\mathcal{C}_{init}^{\mathcal{R}_{ij}^{m+}}$. Supposons qu'il existe une étape q' de la procédure GénèreCliqueMaxInit appliquée à partir de $\mathcal{C}_{init}^{\mathcal{R}_{ij}}$ telle que $\mathcal{C}_{q'} \subset \mathcal{C}_q^+$. Soit maintenant l'ensemble $\mathcal{U}_{min}^+$ des arcs de type ressource supprimés de $\mathcal{C}_q^+$ à l'étape q. Soit l'ensemble $\mathcal{U}_{\min}$ des arcs de type ressource supprimés de $\mathcal{C}_{q'}$ à l'étape q'. On a soit $\mathcal{U}_{min}^+ \supset \mathcal{U}_{\min}$ (les deux étant des sous-ensembles de l'ensemble des arcs entrant dans une même opération $o_d est$) , auquel cas $\mathcal{C}_{q'+1} \subset \mathcal{C}_{q+1}^+$ (déplacement autour de la même opération o_{dest} dans les deux cas), soit $\mathcal{U}_{min}^+ \cap \mathcal{U}_{\min} = \emptyset$, auquel cas $\mathcal{C}_{q'} \subset \mathcal{C}_{q+1}^+$. □

Cette observation permet d'affirmer que la recherche d'une sous-clique optimale pour une opération (i,j) d'occupation l_{ij} (à l'aide de la procédure ExplSousCliqueDomin) peut s'effectuer aussi bien à partir de l'ensemble $\{\mathcal{C}_q^+\}$ qu'à partir de l'ensemble $\{\mathcal{C}_{q'}\}$. On peut donc transformer l'énumération de la façon suivante :

- Explorer les cliques maximales dominantes (ExplCliquesMaxDomin) sur l'ensemble de ressources $\mathcal{R}_{ij}^{m+}$ pour l'opération fictive (i,s)
- Pour chaque clique maximale explorée $\mathcal{C}^+$, énumérer l'ensemble des occupations de $\mathcal{L}_{ij}^m$. Pour chaque occupation possible l_{ij}, rechercher la sous-clique optimale (ExplSousCliqueMaxDomin) de $\mathcal{C}^+$ pour l'opération (i,j).

Une seconde observation permet d'éviter l'énumération des occupations : *la structure de l'arbre d'exploration des sous-cliques de $\mathcal{C}^+$ par* ExplSousCliquesDomin *est la même quelle que soit l'occupation des ressources et la durée de (i,j) correspondante.* En effet, l'ensemble $\mathcal{U}_{\max}$ des arcs de type ressource de $\mathcal{C}^+$ de plus grande date de début au plus tôt qui détermine le fils gauche d'un noeud donné et l'ensemble $\mathcal{U}_{\min}$ des arcs de type ressource de $\mathcal{C}^+$ de plus petite date de fin au plus tard qui détermine le fils droit ne dépendent que de $\mathcal{C}$ et non de (i,j) (voir remarque du paragraphe III.2.6.d)). Soit deux occupations des ressources différentes $l_\alpha \in \mathcal{L}_{ij}^m$ et $l_\beta \in \mathcal{L}_{ij}^m$. Les explorations des sous cliques de $\mathcal{C}^+$ pour l_α et l_β se distinguent toutefois à chaque noeud commun s correspondant à la sous-clique $\mathcal{C}_s$ par :

- si $p_{ij}(l_\alpha) \geq p_{ij}(l_\beta)$ alors $\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s),p_{ij}(l_\alpha)) \geq \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_s),p_{ij}(l_\beta))$
- si $l_\alpha \not\subseteq l_{\mathcal{C}_s}$ alors l'exploration associé à l_α ne continue pas pour les fils du noeud s, ce qui n'est pas forcément le cas pour l_β puisque $l_\alpha \neq l_\beta$.

On observe ainsi qu'au noeud correspondant à la sous-clique $\mathcal{C}_{opt}$ de $\mathcal{C}^+$ de plus petit impact sur l'admissibilité l'occupation des ressources l_{opt} est telle que pour toute autre occupation des ressources $l_\alpha \in \mathcal{L}_{ij}^m$, $l_\alpha \neq l_{opt}$, $l_\alpha \subseteq l_{\mathcal{C}_{opt}}$:
$\Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_{opt}),p_{ij}(l_opt)) \leq \Delta d_{max}(i,j,\mathcal{PI}(\mathcal{C}_{opt}),p_{ij}(l_\alpha))$,
c'est-à-dire $p_{ij}(l_{opt}) \leq p_{ij}(l_\alpha)$.

Une manière alternative d'obtenir $\mathcal{C}_{opt}$ et l_{opt} est ainsi d'explorer les sous-cliques dominantes de la clique $\mathcal{C}^+$ en une seule fois en déterminant à chaque noeud s de l'arbre exploré, l'occupation des ressources l_{min} telle que $p_{ij}(l_{min}) = \min_{l_\alpha \in \mathcal{L}_{ij}^m, l_{ij} \subseteq l_{\mathcal{C}_s}} p_{ij}(l_\alpha)$. L'exploration s'arrête à un noeud $\mathcal{C}_s$ si aucune occupation possible pour (i,j) n'est incluse dans $l_{\mathcal{C}_s}$.

La procédure OccupDuréeMin décrite dans la figure III.19 détermine une telle occupation des ressources l_{min} d'une clique $\mathcal{C}$ en examinant chaque ressource guidante g et le nombre maximum de lignes de charge occupées par $\mathcal{C}$ sur g. La ressource guidante donnant la plus petite durée est sélectionnée pour l'occupation des ressources l_{min} et les ressources annexes sont choisies arbitrairement. La procédure renvoie une occupation des ressources vide si la clique $\mathcal{C}$ n'est pas suffisante. Son algorithme est en $\mathcal{O}(n_1 n_2)$ où $n_1 = |\mathcal{R}_{ij}^{m+}|$ et $n_2 = \max_{r=1,\ldots,M_{ij}^m} |P_{ij}^{mr}|$.

On donne l'algorithme de la procédure CliqueOccupOpt qui intègre les procédures ExplCliquesMaxDomin, ExplSousCliquesDomin et OccupDuréeMin dans la figure III.20. La procédure trouve l'occupation l_{opt} et la clique suffisante $\mathcal{C}_{opt}$ optimales pour l'insertion de (i,j) dans le mode étendu m. L'algorithme ainsi défini est en $\mathcal{O}(q^3 u n_1^3 n_2)$ où n_1 est le nombre de ressources utilisables par (i,j), n_2 est le nombre maximal de ressources dans un pool de (i,j), q le nombre maximal d'arcs dans une clique et u le nombre d'arcs de type ressource du graphe $\mathcal{G}$. En comparaison, l'algorithme appliquant CliqueOpt pour chaque occupation des ressources possible est en $\mathcal{O}(q^3 u M_o^2 n_2^{M_{ij}^m})$. Plus l'aspect multi-ressource est important, plus l'algorithme proposé devient intéressant par rapport à l'énumération car le facteur $n_2^{M_{ij}^m}$ augmente très rapidement.

III.4 Prise en compte des opérations de préparation dans le problème d'insertion

On suppose dans ce paragraphe que l'opération d'exécution (i,j) à insérer nécessite un ensemble de ressources $\mathcal{R}_{ij}$ composé d'un ensemble de ressources principales disjonctives $\mathcal{R}b_{ij}$ devant être préparées suivant les règles définies dans le paragraphe II.2 et d'un ensemble de ressources complémentaires $\mathcal{R}c_{ij}$ ne nécessitant pas de préparation. Comme dans le paragraphe III.2, on considère que le problème de sélection du mode étendu m de

Proc OCCUPDURÉEMIN $(\mathcal{C}, (i,j), M_{ij}^m, \mathcal{P}_{ij}^m, l_{ij+}, l_{ij-}, l_{min}, p_{min})$
    $p_{min} \leftarrow \infty$
    $l_{min} \leftarrow \emptyset_l$
    Pour $k \in \mathcal{R}_{ij}^{m+}$ **faire**
        Si $e_{\mathcal{C}}^k > e_{ij-}^{mk}$ **alors**
            $e \leftarrow \min(e_{\mathcal{C}}^k, e_{ij+}^{mk})$
            Si $k \in Pg_{ij}^m$ **et** $p_{ij}^{mke} < p_{min}$ **alors**
                $p_{min} \leftarrow p_{ij}^{mke}$; $e_{min}^k = e$; $\forall q \in Pg_{ij}^m, q \neq k, e_{min}^q \leftarrow 0$
            Sinon Si $k \notin Pg_{ij}^m$ **alors**
                    Soit $Pk_{i,j} \in \mathcal{P}_{ij}^m, k \in Pk_{ij}$
                    Si $\forall q \in Pk_{ij}$, $e_{min}^q = 0$ **alors** $e_{min}^k \leftarrow e$.
            Finsi
        Finsi
    Finpour
    Pour $r \in 1...M_{ij}^m$ **faire**
        Si $\forall k \in P_{ij}^{mr}$, $e_{min}^k = 0$ **alors** $p_{min} \leftarrow \infty$; $l_{min} \leftarrow \emptyset_l$
    Finpour
Finproc

FIG. III.19 – *Algorithme de* OCCUPDURÉEMIN

l'opération, des ressources dans chaque pool du mode m et du nombre de lignes de charge sur chaque ressource sélectionnée est résolu. L'occupation des ressources $l_{ij} = \{e_{ij}^1, ..., e_{ij}^M\}$ et la durée p_{ij} sont fixées.

On suppose en outre dans ce paragraphe que les opérations de préparation des ressources principales ne nécessitent pas de ressources complémentaires. Pour tout mode étendu de montage μ, pour tout mode étendu de démontage δ, pour tout mode étendu de changement de type τ, $\mathcal{P}c_{sm}^{\mu} = \mathcal{P}c_{sd}^{\delta} = \mathcal{P}c_{sct}^{\tau} = \emptyset$. On n'a donc pas de problème de résolution de conflit d'utilisation des ressources complémentaires de préparation. La durée de toute opération de préparation est complètement déterminée par les ressources principales préparées. En effet, la ressource guidante de préparation est obligatoirement une ressource principale.

Dans le paragraphe III.4.1, des hypothèses restrictives sur les valeurs temps de préparation à la base des règles de dominance énoncées par la suite sont données. Dans le paragraphe III.4.2, une position d'insertion est définie, les modifications du graphe $\mathcal{G}$ suite à l'insertion sont présentés et leur impact sur l'admissibilité est calculé. La caractérisation d'une position d'insertion et les principes de la méthode de recherche d'une position d'insertion optimale sont décrits en III.4.3. Cette méthode nécessite une extension de la procédure d'exploration des cliques maximales dominantes EXPLCLIQUESMAXDOMIN du paragraphe III.2.5.d) décrite en III.4.4. Ce paragraphe présente une amélioration de l'approche heuristique proposée dans [Artigues et Roubellat, 1997d].

Proc CLIQUEOCCUPOPT $((i,j), M_{ij}^m, \mathcal{P}_{ij}^m, l_{ij+}, l_{ij-}, r_{ij}, d_{ij}, \mathcal{G}, \mathcal{C}_{opt}, \Delta d_{max}^{opt}, l_{opt})$
 GÉNÈRECLIQUEMAXINIT $((i,j), l_{ij+}, r_{ij}, \mathcal{G}, \mathcal{C}_0)$
 EXPLSOUSCLIQUESDOMINOCCUP[1] $(\mathcal{C}_0, (i,j), M_{ij}^m, \mathcal{P}_{ij}^m, l_{ij+}, l_{ij-}, r_{ij}, d_{ij}, \mathcal{C}_{opt}, \Delta d_{max}^{opt}, l_{opt})$
 Si $\Delta d_{max}^{opt} = 0$ **alors**
  $STOP \leftarrow VRAI$
 Sinon
  $STOP \leftarrow FAUX$
 FinSi
 $cpt \leftarrow 0$;
 Tantque $\mathcal{C}_{cpt} \neq \mathcal{C}_{fin}^{\mathcal{R}_{ij}^{m+}}$ et $d_{\mathcal{C}} < d_{i,j}$ et $STOP = FAUX$ **faire**
  $cpt \leftarrow cpt + 1$
  Soit $o_{dest} = dest_\alpha$ telle que $d_\alpha = \min_{u_\beta \in \mathcal{C}_{cpt-1}} d_\beta$
  $\mathcal{C}_{cpt} = dd_{o_{dest}}(\mathcal{C}_{cpt-1})$
  EXPLSOUSCLIQUESDOMINPARTOCCUP[1] $(\mathcal{C}_0, (i,j), M_{ij}^m, \mathcal{P}_{ij}^m, l_{ij+}, l_{ij-}, r_{ij}, d_{ij},$
   $\mathcal{U}_{sort}^{\mathcal{R}_{ij}^{m+}}(o_{dest}), \mathcal{C}_{opt}(cpt), \Delta d_{max}^{opt}(cpt), l_{opt}(cpt))$
  Si $\Delta d_{max}^{opt}(cpt) < \Delta d_{max}^{opt}$ **alors**
   $\Delta d_{max}^{opt} \leftarrow \Delta d_{max}^{opt}(cpt)$
   $\mathcal{C}_{opt} \leftarrow \mathcal{C}_{opt}(cpt)$
   $l_{opt} \leftarrow l_{opt}(cpt)$
   Si $\Delta d_{max}^{opt} = 0$ **alors**
    $STOP \leftarrow VRAI$
   FinSi
  FinSi
 FinTantque
FinProc

[1] : EXPLSOUSCLIQUESDOMINOCCUP et EXPLSOUSCLIQUESDOMINPARTOCCUP sont respectivement identiques à EXPLSOUSCLIQUESDOMIN et EXPLSOUSCLIQUESDOMINPART, à la différence que l'occupation l_{ij} et la durée p_{ij} sont calculées à chaque noeud en appliquant la procédure OCCUPDURÉEMIN.

FIG. III.20 – Algorithme de CLIQUEOCCUPOPT

III.4.1 Inégalités valides sur les temps de préparation

Si les temps de préparation sont quelconques, il est très difficile de dégager des règles de dominance pour restreindre l'énumération des possibilités d'insertion. On définit un ensemble d'inégalités valides qui restreignent les possibilités du modèle mais qui sont d'une part réalistes pour la plupart des ateliers considérés et qui, d'autre part, permettent d'obtenir des propriétés intéressantes dans les algorithmes de résolution.

On suppose que *l'inégalité triangulaire* est vérifiée pour les temps de changement de type. Cette inégalité (voir par exemple [Monma et Potts, 1989]) signifie qu'il n'est jamais moins rapide pour une ressource principale isolée ou pour une association, de passer directement d'un type d'opérations d'exécution A à un autre type B que de passer par un type intermédiaire C. Dans le modèle considéré, cette inégalité n'est vérifiée que si les opérations de changement de type concernées utilisent la même ressource guidante g et le même nombre de lignes de charge e sur g :

$$p_{sct}^{\tau ABge} \leq p_{sct}^{\tau ACge} + p_{sct}^{\tau CBge} \qquad \forall \tau, \forall A, B \in \mathcal{T} \times \mathcal{T}, \forall g \in Pg_{sct}^{\tau}, \forall e \in e_{sct-}^{\tau g}, ..., e_{sct+}^{\tau g} \quad \text{(III.14)}$$

Une deuxième inégalité suppose qu'entre deux opérations d'exécution utilisant la même association de ressources principales il existe une sélection des ressources guidantes mises en jeu telle qu'il n'est pas moins rapide d'effectuer le changement de type sur les ressources associées que de démonter l'association, d'effectuer le changement de type sur chaque ressource isolée et de remonter l'association. Soit τ_1 un mode de changement de type concernant la ressource principale k. Soit τ_2 un mode étendu de changement de type concernant une association a telle que $|a| > 1$ et $k \in a$. Soit μ_2, le mode étendu de montage concernant a. Soit δ_2 le mode étendu de démontage de a.

Quels que soient la ressource guidante de changement de type g et le nombre de lignes de charge e sélectionnés pour le mode étendu τ_2, il existe une ressource guidante de montage g_{sm} et un nombre de lignes de charge e_{sm} du mode étendu μ_2, une ressource guidante de changement de type g_{sct} et un nombre de lignes de charge e_{sct} du mode étendu τ_1, une ressource guidante de démontage g_{sd} et un nombre de lignes de charge e_{sd} du mode étendu δ_2 tels que :

$$p_{sct}^{\tau_2 ABge} \leq p_{sm}^{\mu_2 g_{sm} e_{sm}} + p_{sct}^{\tau_1 ABg_{sct} e_{sct}} + p_{sd}^{\delta_2 g_{sd} e_{sd}}, \qquad \forall A, B \in \mathcal{T} \times \mathcal{T} \qquad \text{(III.15)}$$

Une troisième inégalité suppose qu'il existe une sélection des ressources guidantes de changement de type mises en jeu telle qu'il n'est pas moins rapide de changer de type une ressource isolée k qu'une association a contenant la ressource k. Soit une ressource principale k. Soit τ_1 le mode étendu de changement de type de k. Soit a une association telle que $|a| > 1$ et $k \in a$. Soit τ_2 le mode étendu de changement de type associé à a. Quels que soient la ressource guidante g et le nombre de lignes de charge e sélectionnés pour le mode étendu τ_1, il existe une ressource guidante g_2 et un nombre de lignes de charge e_2 du mode étendu τ_2 tels que :

$$p_{sct}^{\tau_1 ABge} \leq p_{sct}^{\tau_2 ABg_2 e_2} \qquad \forall A, B \in \mathcal{T} \times \mathcal{T} \qquad \text{(III.16)}$$

III.4.2 Graphe à insérer et impact sur l'admissibilité

Pour caractériser une position d'insertion, la notion d'opérations d'exécution consécutives est définie comme suit :

Définition 10 *Deux opérations d'exécution p et f sont dites* consécutives *sur la ressource principale k si aucune autre opération d'exécution n'est séquencée entre p et f sur k.*

Cette définition n'exclut pas la présence d'opérations de préparation entre p et f sur k.

La définition 4 d'une position d'insertion $\mathcal{PI}$ par un ensemble de 4-uplets $\{(k, l, p(k, l), f(k, l))\}$ et les conditions nécessaires et suffisantes de validité (définition 5) d'une position d'insertion sur un ensemble de ressources définies au paragraphe III.2.2.b) sont encore valables en considérant la définition 10.

On peut décomposer toute position d'insertion valide $\mathcal{PI}$ de (i,j) en une position d'insertion $\mathcal{PI}(b)$ sur les ressources principales et une position d'insertion $\mathcal{PI}(c)$ sur les ressources complémentaires telles que $\mathcal{PI} = \mathcal{PI}(b) \cup \mathcal{PI}(c)$. L'insertion de (i,j) dans une position d'insertion supprime d'une part les opérations de préparation ordonnancées entre chaque paire d'opérations $p(k,1)$ et $f(k,1)$ telle que $(k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$. D'autre part cette insertion génère entre chaque paire d'opérations $p(k,1)$ et $f(k,1)$ ainsi définie une séquence d'opérations incluant l'opération d'exécution (i,j) et éventuellement des opérations de préparation. Pour mesurer l'impact sur l'admissibilité de l'insertion de (i,j) à la position $\mathcal{PI}$ on doit considérer à la fois :

- l'insertion de (i,j) sur les ressources complémentaires d'exécution à la position $\mathcal{PI}(c)$,
- l'insertion de la séquence d'opérations sur les ressources principales à la position $\mathcal{PI}(b)$.

Pour mesurer précisément cet impact, on considère les modifications du graphe $\mathcal{G}$ causées par ces deux composantes. Les modifications nécessaires dans le graphe $\mathcal{G}$ sur chaque ligne de charge l de chaque ressource complémentaire c concernant les arcs de type ressource entre $p(c,l)$ et $f(c,l)$, entre $p(c,l)$ et (i,j) et entre (i,j) et $p(c,l)$ ont été précisément décrites au paragraphe III.2.2.c). Soit r_{ij}^{*c} et d_{ij}^{*c} la date de début au plus tôt et la date de fin au plus tard de (i,j) imposées par les opérations déjà présentes sur les ressources complémentaires (voir les équations III.5, III.6 et III.7). On pose :

$$r_{ij}^{*c} = \max_{(k,l,p(k,l),f(k,l)) \in \mathcal{PI}(c)} (r_{p(k,l)} + p_{p(k,l)}) \tag{III.17}$$

$$d_{ij}^{*c} = \min_{(k,l,p(k,l),f(k,l)) \in \mathcal{PI}(c)} (d_{f(k,l)} - p_{f(k,l)}) \tag{III.18}$$

Pour décrire les modifications causées par l'insertion sur les ressources principales à la position $\mathcal{PI}(b)$, on représente la séquence d'opérations à insérer par un graphe $\mathcal{G}_{ij}^*$. Les principes de construction de ce graphe ont été présentés dans [Artigues et Roubellat, 1997c] et sont rappelés ici. Quatre configurations possibles de préparation entre les ensembles de paires $(p(k,1), f(k,1))$, $k \in \mathcal{R}b_{ij}$ sont considérées et sont représentées respectivement dans les figures III.21,III.22,III.23 et III.24. Pour chaque configuration, les modifications transforment le graphe $\mathcal{G}$ en un graphe $\mathcal{G}^*$ correspondant à l'inclusion dans $\mathcal{G}$ du graphe $\mathcal{G}_{ij}^*$. Dans les figures, les sommets et arcs supprimés dans $\mathcal{G}$ par l'insertion sont représentés en ligne brisée. Les sommets et arcs de $\mathcal{G}_{ij}^*$ sont représentés en pointillés. Les données (dates, arcs, ...) associées au graphe $\mathcal{G}^*$ sont notées avec un $(^*)$. Les ressources principales $k \in \mathcal{R}b_{ij}$ étant disjonctives, on peut employer indifféremment $p(k,1)$ et $p(k)$ d'une part et $f(k,1)$ et $f(k)$ d'autre part. Pour chaque configuration, on calcule la date de début au plus tôt r_{ij}^{*b} et la date de fin au plus tard d_{ij}^{*b} à partir des dates des opérations d'exécution présentes sur les ressources principales et en fonction des contraintes de succession des opérations de préparation et d'exécution sur les ressources principales décrites en II.2.2.d). L'impact de l'insertion de (i,j) dans la position valide $\mathcal{PI}$ sur l'admissibilité s'exprime alors par :

$$\Delta d_{max}(i,j,\mathcal{PI}) = \max(0, \max(r_{ij}, r_{ij}^{*c}, r_{ij}^{*b}) + p_{ij} - \min(d_{ij}, d_{ij}^{*c}, d_{ij}^{*b})) \tag{III.19}$$

i) Cette configuration concerne l'insertion de (i, j) au sein d'une association existante. Cette configuration est elle-même divisée en trois configurations (i1, i2 et i3).

i1) La première configuration est l'insertion entre deux opérations p et f utilisant la même association que (i, j), c'est-à-dire l'ensemble de ressources principales $\mathcal{R}b_{i,j}$. La

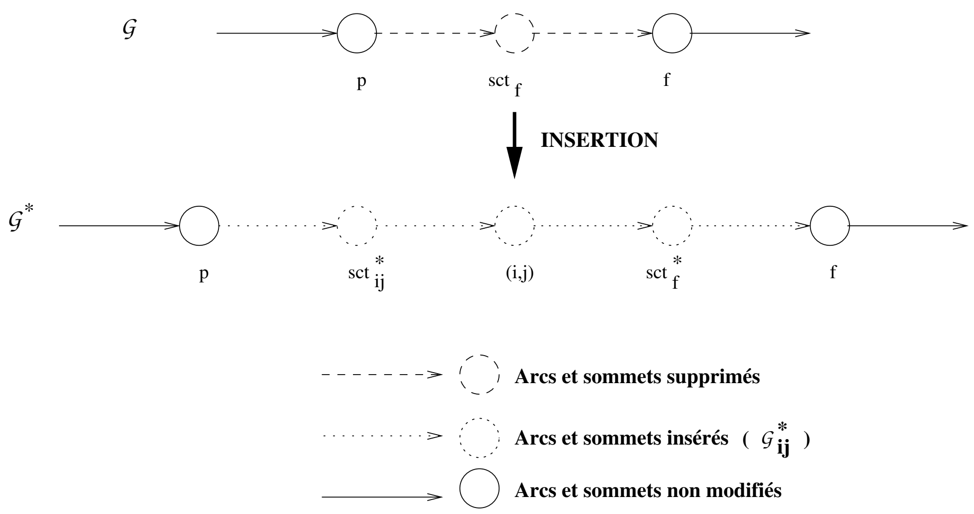


FIG. III.21 – Modifications dans $\mathcal{G}$ dues à l'insertion sur des ressources associées

position d'insertion $\mathcal{PI}(b)$ est alors telle que $\forall k \in \mathcal{R}b_{ij}$, $p(k) = p$ et $f(k) = f$. Avant l'insertion, une seule opération peut être présente entre p et f : une opération de changement de type sct_f si $RT_p \neq RT_f$. C'est le cas du graphe $\mathcal{G}$ représenté en haut de la figure III.21. Un unique mode de changement de type $\tau(\mathcal{R}b_{ij})$ correspondant à l'association $\mathcal{R}b_{ij}$ donne la durée $p_{sct_f} = p_{sct}^{\tau(\mathcal{R}b_{ij})RT_pRT_fg(\mathcal{R}b_{ij})1}$ où $g(\mathcal{R}b_{ij})$ est la ressource guidante du mode étendu de changement de type contenue ici dans $\mathcal{R}b_{ij}$ puisque les opérations de préparation n'ont pas de ressources complémentaires (voir paragraphe II.2.2.c)).

Dans ce contexte, le graphe $\mathcal{G}^*_{ij}$ représenté en bas de la figure III.21 contient au plus trois opérations : l'opération de changement de type sct^*_{ij} de durée $p_{sct}^{\tau(\mathcal{R}b_{ij})RT_pRT_{ij}g(\mathcal{R}b_{ij})1}$ et d'occupation des ressources $l_{sct^*_{ij}} = lb_{ij}$, nécessaire si $RT_{ij} \neq RT_p$, l'opération (i, j) de durée p_{ij}, l'opération de changement de type sct^*_f de durée $p_{sct}^{\tau(\mathcal{R}b_{ij})RT_{ij}RT_fg(\mathcal{R}b_{ij})1}$ et d'occupation des ressources $l_{sct^*_f} = lb_{ij}$ nécessaire si $RT_{ij} \neq RT_f$. On a ainsi d'après les contraintes II.3, II.4 et II.5 :

$$r^{*b}_{ij} = r_p + p_p + p_{sct}^{\tau(\mathcal{R}b_{ij})RT_pRT_{ij}g(\mathcal{R}b_{ij})1} \tag{III.20}$$

$$d^{*b}_{ij} = d_f - p_f - p_{sct}^{\tau(\mathcal{R}b_{ij})RT_{ij}RT_fg(\mathcal{R}b_{ij})1} \tag{III.21}$$

D'après l'inégalité triangulaire associée au mode $\tau(\mathcal{R}b_{ij})$ (voir paragraphe III.4.1), l'insertion de (i, j) ne peut faire commencer l'opération f plus tôt car on a $p_{sct_f} \leq p_{sct^*_{ij}} + p_{sct^*_f}$. L'insertion de (i, j) ne peut pas améliorer l'admissibilité !

i2) La deuxième configuration correspond à l'insertion de (i, j) avant une opération d'exécution f qui est en tête d'une séquence d'opérations d'exécution utilisant l'association $\mathcal{R}b_{ij}$. La position d'insertion $\mathcal{PI}(b)$ est telle que $\forall k \in \mathcal{R}b_{ij}$, $f(k) = f$ et $\exists k, k' \in \mathcal{R}b_{ij} \times \mathcal{R}b_{ij}$ telles que $p(k) \neq p(k')$. Les opérations de préparation présentes

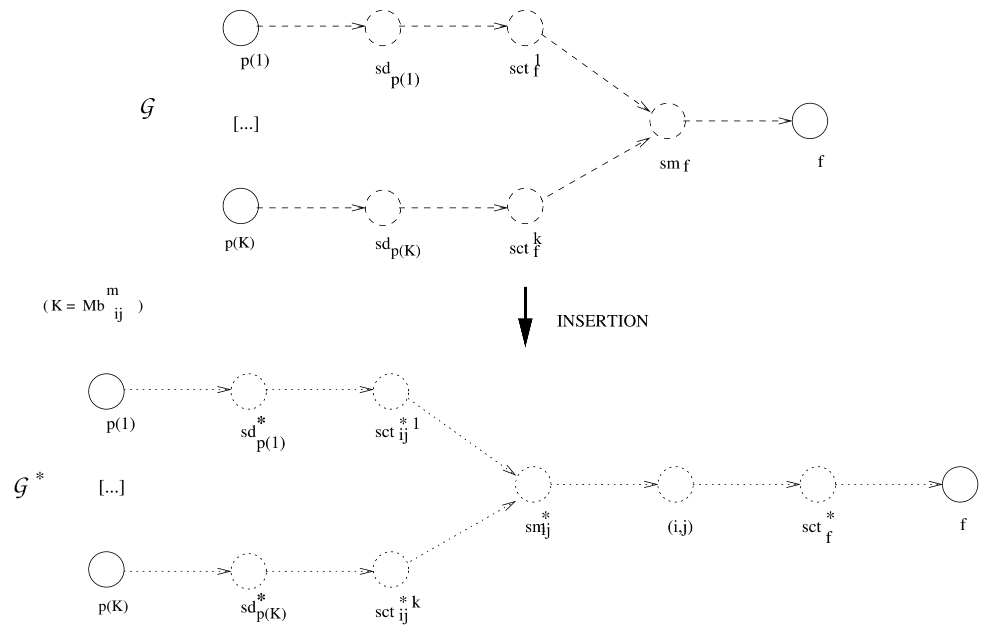


FIG. III.22 – Modifications dans $\mathcal{G}$ dues à l'insertion au début d'une association

dans $\mathcal{G}$ entre chaque opération $p(k)$ et l'unique opération f sont représentées en haut de la figure III.22 (sous-graphe en ligne brisée). f étant la première opération d'exécution utilisant l'association $\mathcal{R}b_{ij}$, elle est précédée d'une opération de montage sm_f. Un unique mode étendu de montage $\mu(\mathcal{R}b_{ij})$ associé à $\mathcal{R}b_{ij}$ donne la durée $p_{sm_f} = p_{sm}^{\mu(\mathcal{R}b_{ij})g(\mathcal{R}b_{ij})1}$ où $g(\mathcal{R}b_{ij})$ est la ressource guidante du mode étendu de montage contenue dans $\mathcal{R}b_{ij}$. Avant d'être associée au sein de l'ensemble $\mathcal{R}b_{ij}$, chaque ressource k est, après l'exécution de $p(k)$, mise à l'état isolée par l'opération de démontage $sd_{p(k)}$ de l'association $\mathcal{R}b_{p(k)}$. Un unique mode étendu de démontage $\delta(\mathcal{R}b_{p(k)})$ associé à $\mathcal{R}b_{p(k)}$ donne la durée $p_{sd_{p(k)}} = p_{sd}^{\delta(\mathcal{R}b_{p(k)})g(\mathcal{R}b_{p(k)})1}$ où $g(\mathcal{R}b_{p(k)})$ est la ressource guidante du mode étendu de démontage contenue dans $\mathcal{R}b_{p(k)}$. Si $RT_{p(k)} \neq RT_f$, la ressource k isolée subit l'opération de changement de type sct_f^k avant d'être montée. Un unique mode de changement de type $\tau(k)$ associé à la ressource isolée k donne la durée $p_{sct_f^k} = p_{sct}^{\tau(k)RT_{p(k)}RT_f k1}$ où k est obligatoirement la ressource guidante.

Le graphe $\mathcal{G}_{ij}^*$ à insérer et le graphe résultant $\mathcal{G}^*$ sont représentés au bas de la figure III.22. Après (i, j) on trouve, comme dans la configuration **i1**, une unique opération de changement de type sct_f^* nécessaire uniquement si $RT_{ij} \neq RT_f$. Avant (i, j), le graphe $\mathcal{G}_{ij}^*$ a la même structure que le sous-graphe supprimé dans $\mathcal{G}$, puisque (i, j) se retrouve en première position de l'association $\mathcal{R}b_{ij}$. Les opérations de démontage $sd_{p(k)}^*$ sont identiques aux opérations $sd_{p(k)}$ car elles ne dépendent que de l'association à démonter qui demeure $\mathcal{R}b_{p(k)}$. Chaque opération sct_f^k est remplacée par une opération sct_{ij}^{*k} si $RT_{ij} \neq RT_{p(k)}$. La durée de sct_{ij}^{*k} est $p_{sct_{ij}^{*k}} = p_{sct}^{\tau(k)RT_{p(k)}RT_{ij}k1}$. L'opération de montage sm_f est remplacée par l'opération identique sm_{ij}^* car (i,j) et f utilisent la même association. On a ainsi d'après les contraintes II.6, II.7, II.8, II.9 et II.10 :

$$r_{ij}^{*b} = \max_{k \in \mathcal{R}b_{ij}} (r_{p(k)} + p_{p(k)} + p_{sd}^{\delta(\mathcal{R}b_{p(k)})g(\mathcal{R}b_{p(k)})1} + p_{sct}^{\tau(k)RT_{p(k)}RT_{ij}k1}) + p_{sm}^{\mu(\mathcal{R}b_{ij})g(\mathcal{R}b_{ij})1} \quad \text{(III.22)}$$

d_{ij}^{*b} est déterminé comme dans la configuration **i1** (équation III.21). On montre que l'insertion de (i,j) ne peut provoquer un décalage à gauche de la date de début au plus tôt de f. En effet, sur chaque ressource principale $k \in \mathcal{R}b_{ij}$, par rapport à la situation avant l'insertion, les temps démontage sont inchangés ($p_{sd^*_{p(k)}} = p_{sd_{p(k)}}$). Le temps de montage de $\mathcal{R}b_{ij}$ est le même ($p_{sm^*_{ij}} = p_{sm_f}$). On applique l'inégalité associée aux modes étendus $\tau(k)$ et τ, où τ est le mode étendu de changement de type correspondant à l'association $\mathcal{R}b_{ij}$ (voir paragraphe III.4.1). On obtient
$p_{sct}^{\tau(k)RT_{ij}RT_fk1} \leq p_{sct}^{\tau RT_{ij}RT_fg1}$ où $g \in \mathcal{R}b_{ij}$. (A)
En appliquant l'inégalité triangulaire associée au mode étendu $\tau(k)$, on obtient :

$$p_{sct}^{\tau(k)RT_{p(k)}RT_fk1} \leq p_{sct}^{\tau(k)RT_{p(k)}RT_{ij}k1} + p_{sct}^{\tau(k)RT_{ij}RT_fg1}.$$

D'où, d'après (A),

$$p_{sct}^{\tau(k)RT_{p(k)}RT_fk1} \leq p_{sct}^{\tau(k)RT_{p(k)}RT_{ij}k1} + p_{sct}^{\tau RT_{ij}RT_fg1}.$$

On a $p_{sct_f^k} \leq p_{sct_{ij}^{*k}} + p_{sct_f^*}$, ce qui implique l'impossibilité d'améliorer l'admissibilité par l'insertion de (i,j).

i3) La troisième configuration, symétrique à la précédente, correspond à l'insertion de (i,j) après une opération d'exécution p qui est à la fin d'une séquence d'opérations d'exécution utilisant l'association $\mathcal{R}b_{ij}$. La position d'insertion $\mathcal{PI}(b)$ est telle que $\forall k \in \mathcal{R}b_{ij}$, $p(k) = p$ et $\exists k, k' \in \mathcal{R}b_{ij} \times \mathcal{R}b_{ij}$ telles que $f(k) \neq f(k')$. Les opérations de préparation présentes dans $\mathcal{G}$ entre l'unique opération p et chaque opération $f(k)$ sont représentées en haut de la figure III.23 (sous-graphe en ligne brisée). p étant la dernière opération de l'association $\mathcal{R}b_{ij}$, elle est suivie d'une opération de démontage sd_p. Un unique mode étendu de démontage $\delta(\mathcal{R}b_{ij})$ associé à $\mathcal{R}b_{ij}$ donne la durée $p_{sd_p} = p_{sd}^{\delta(\mathcal{R}b_{ij})g(\mathcal{R}b_{ij})1}$ où $g(\mathcal{R}b_{ij})$ est la ressource guidante du mode étendu de démontage contenue dans $\mathcal{R}b_{ij}$. Si $RT_{f(k)} \neq RT_p$, la ressource k isolée subit l'opération de changement de type $sct_{f(k)}^k$ avant d'être montée. Le mode unique de changement de type $\tau(k)$ associé à la ressource isolée k donne la durée $p_{sct_{f(k)}} = p_{sct}^{\tau(k)RT_pRT_{f(k)}k1}$ où k est obligatoirement la ressource guidante. Avant d'exécuter l'opération $f(k)$, chaque ressource isolée k est associée à l'ensemble $\mathcal{R}b_{f(k)}$ par l'opération de montage $sm_{f(k)}$ de l'association $\mathcal{R}b_{f(k)}$. Un unique mode étendu de montage $\mu(\mathcal{R}b_{f(k)})$ donne la durée $p_{sm_{f(k)}} = p_{sm}^{\mu(\mathcal{R}b_{f(k)})g(\mathcal{R}b_{f(k)})1}$ où $g(\mathcal{R}b_{f(k)})$ est la ressource guidante du mode étendu de montage contenue dans $\mathcal{R}b_{f(k)}$.

Le graphe $\mathcal{G}_{ij}^*$ à insérer et le graphe résultant $\mathcal{G}^*$ sont représentés au bas de la figure III.23. Avant (i,j) on trouve une unique opération de changement de type sct_{ij}^* nécessaire

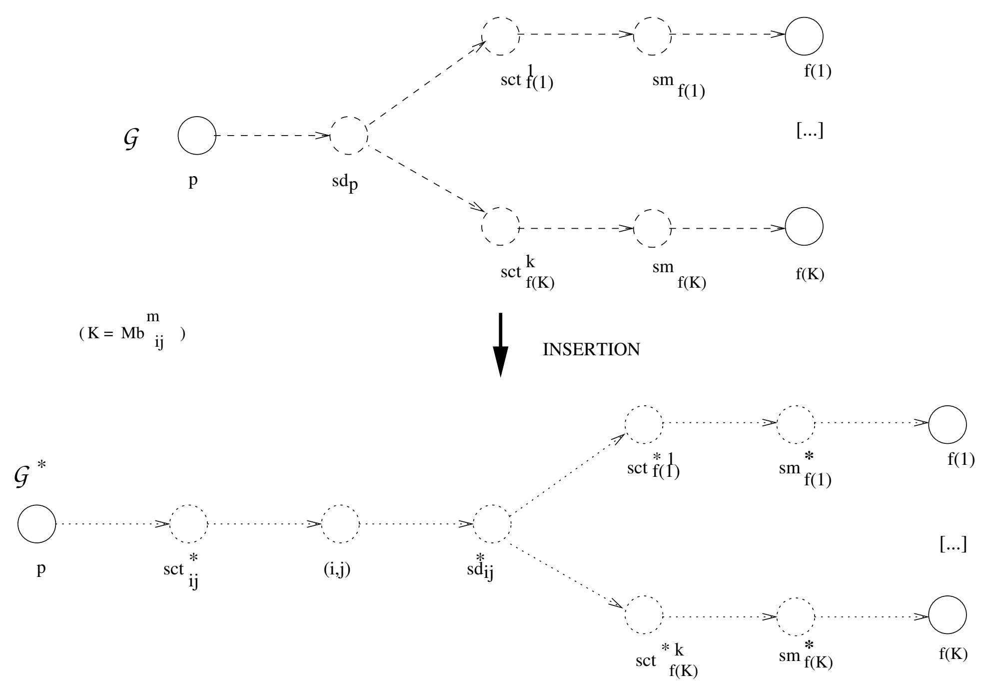


FIG. III.23 – Modifications dans $\mathcal{G}$ dues à l'insertion à la fin d'une association

uniquement si $RT_{ij} \neq RT_p$, comme dans la configuration **i1**. Après (i,j), le graphe $\mathcal{G}^*_{ij}$ a la même structure que le sous-graphe supprimé dans $\mathcal{G}$, puisque (i,j) se retrouve en dernière position de l'association $\mathcal{R}b_{ij}$. Les opérations de montage $sm^*_{f(k)}$ sont identiques aux opérations $sm_{f(k)}$ car elles ne dépendent que de l'association à monter qui demeure $\mathcal{R}b_{f(k)}$. Chaque opération $sct^k_{f(k)}$ est remplacée par une opération $sct^{*k}_{f(k)}$ si $RT_{ij} \neq RT_{f(k)}$. La durée de $sct^{*k}_{f(k)}$ est $p_{sct^{*k}_{f(k)}} = p_{sct}^{\tau(k)RT_{ij}RT_{f(k)}k1}$. L'opération de démontage sd_p est remplacée par l'opération identique sd^*_{ij} car (i,j) et p utilisent la même association. On a ainsi d'après les contraintes II.6, II.7, II.8, II.9 et II.10 :

$$d^{*b}_{ij} = \min_{k \in \mathcal{R}b_{ij}} (d_{f(k)} - p_{f(k)} - p_{sm}^{\mu(\mathcal{R}b_{f(k)})g(\mathcal{R}b_{f(k)})1} - p_{sct}^{\tau(k)RT_{ij}RT_{f(k)}k1}) - p_{sd}^{\delta(\mathcal{R}b_{ij})g(\mathcal{R}b_{ij})1} \quad \text{(III.23)}$$

r^{*b}_{ij} est déterminé comme dans la configuration **i1** (équation III.20). On montre d'une façon symétrique à la configuration **i2**, que l'insertion de (i,j) ne peut provoquer de décalage à gauche de la date de début au plus tôt d'une opération $f(k)$, $k \in \mathcal{R}b_{ij}$.

ii) Cette dernière configuration correspond à l'insertion à une position où l'association $\mathcal{R}b_{ij}$ n'est pas constituée. La position d'insertion $\mathcal{PI}$ est telle que $\exists k, k' \in \mathcal{R}b_{ij} \times \mathcal{R}b_{ij}$ telles que $p(k) \neq p(k')$ et $\exists k'', k''' \in \mathcal{R}b_{ij} \times \mathcal{R}b_{ij}$ telles que $f(k'') \neq f(k''')$. Deux catégories de ressources principales doivent être considérées dans le graphe $\mathcal{G}$, représentées en haut de la figure III.24 :

Cas 1) Soit $k \in \mathcal{R}b_{ij}$ telle que $p(k)$ et $f(k)$ ne sont pas consécutives sur l'ensemble de leurs ressources principales, i.e. elles n'utilisent pas la même association. Dans ce cas,

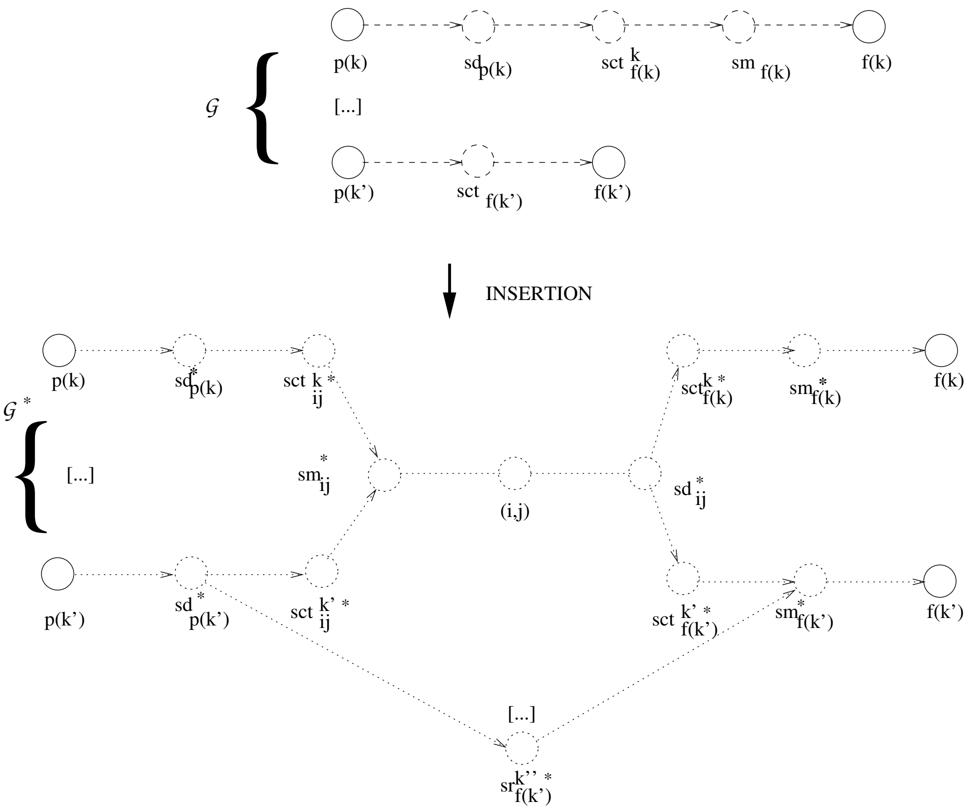


FIG. III.24 – Modifications dans $\mathcal{G}$ dues à l'insertion à une position valide quelconque

on trouve sur k une opération de démontage $sd_{p(k)}$, une opération de changement de type $sct^k_{f(k)}$ dans le cas où $RT_{p(k)} \neq RT_{f(k)}$ et une opération de montage $sm_{f(k)}$.

Cas 2) Soit $k' \in \mathcal{R}b_{ij}$ telle que $p(k')$ et $f(k')$ sont consécutives sur l'ensemble de leurs ressources principales, i.e. elles utilisent la même association, différente de $\mathcal{R}b_{ij}$. Dans ce cas une unique opération de changement de type $sct_{f(k')}$ est présente entre $p(k')$ et $f(k')$ dans le cas où $RT_{p(k')} \neq RT_{f(k')}$.

L'ensemble des opérations de préparation présentes entre chaque paire $(p(k), f(k))$ et $(p(k'), f(k'))$ est supprimé.

Le graphe $\mathcal{G}^*_{ij}$ à insérer, représenté dans le bas de la figure III.24, est décomposable en deux parties. La partie antérieure à (i, j) est identique à la partie antérieure à (i, j) du graphe de la configuration **i2**. La partie postérieure à (i, j) est identique à la partie postérieure à (i, j) du graphe de la configuration **i3**. La seule différence intervient pour les couples d'opérations $p(k')$ et $f(k')$ du cas 2 ci-dessus. L'opération de changement de type $sct_{f(k')}$ est exécutée sur l'ensemble de ressources $\mathcal{R}b_{p(k')} = \mathcal{R}b_{f(k')}$. Sur chaque ressource $k'' \in \mathcal{R}b_{p(k')}$ telle que $k'' \notin \mathcal{R}b_{ij}$, on doit en effet remplacer $sct_{f(k')}$ par une opération de changement de type $sct^{*k''}_{f(k)}$ entre $sd^*_{p(k')}$ et $sm^*_{f(k')}$ de durée $p_{sct}^{\tau(k')RT_{p(k'')}RT_{f(k')}k''1}$. r^{*b}_{ij} est calculé comme dans la configuration **i2** (équation III.22). d^{*b}_{ij} est calculé comme dans la configuration **i3** (équation III.23). Comme pour les configurations précédentes, en utilisant les inégalités valides sur les temps de préparation, on peut montrer que les dates de début au plus tôt des opérations $f(k)$, $k \in \mathcal{R}b_{ij}$ ne peuvent pas être décalées vers la gauche par l'insertion de l'opération (i,j). De plus, chaque opération $sct^{*k''}_{f(k')}$ avec $k'' \notin \mathcal{R}b_{ij}$ est telle que $p_{sct^{*k''}_{f(k')}} \leq p_{sct_{f(k')}}$. L'opération de changement de type $sct^{*k''}_{f(k')}$ sur la ressource isolée k'' ordonnancée entre $p(k')$ et $f(k')$ sur k'' ne peut ainsi avoir d'impact sur l'admissibilité.

complexité de la détermination du sous-graphe $\mathcal{G}^*_{ij}$ Soit une position d'insertion $\mathcal{PI}(b)$. L'algorithme de construction du sous-graphe $\mathcal{G}^*_{ij}$, de calcul des dates d^{*b}_{ij} r^{*b}_{ij}, et de l'impact sur l'admissibilité associé à partir de $\mathcal{PI}(b)$ est en $\mathcal{O}(K)$ où $K = |\mathcal{R}b_{ij}|$.

III.4.3 Caractérisation d'une position d'insertion et principes de la méthode de recherche d'une position d'insertion optimale

On montre dans le paragraphe III.4.3.a) qu'on peut encore caractériser une position d'insertion par une clique d'arcs de type ressource, par l'intermédiaire de la définition d'un arc fictif. Un exemple est donné en III.4.3.b). Sur la base de cette caractérisation les principes de la méthode de recherche d'une position d'insertion optimale sont données en III.4.3.c).

III.4.3.a) Arc fictif caractérisant une position d'insertion sur les ressources principales

Dans le paragraphe III.2.4.a), on a montré qu'une clique d'arcs de type ressource de $\mathcal{G}$ telle que $l_{\mathcal{C}} \geq l_{ij}$ caractérise une position d'insertion suffisante pour une opération (i,j) ne nécessitant pas de préparation. L'objectif de ce paragraphe est d'étendre cette caractérisation à une opération (i,j) nécessitant de la préparation. On peut remarquer que les modifications occasionnées dans $\mathcal{G}$ décrites dans le paragraphe III.4.2 (contexte avec préparation) sont plus profondes que celles décrites dans le paragraphe III.2.2.c) (contexte sans préparation). En effet, des sommets de $\mathcal{G}$ représentant des opérations de préparation entre chaque paire d'opérations d'exécution $p(k)$ et $f(k)$ sont supprimés. Un arc de type ressource u_α de $\mathcal{G}$ tel que $e^k_\alpha > 0$ où $k \in \mathcal{R}b_{ij}$ peut avoir comme origine ou destination une opération de préparation à supprimer et ne peut ainsi directement caractériser une position d'insertion.

On considère une position d'insertion valide $\mathcal{PI}(b)$ sur les ressources principales $\mathcal{R}b_{ij}$. Pour caractériser $\mathcal{PI}(b)$, on peut déterminer un graphe dans lequel un arc de type ressource peut directement caractériser une position d'insertion sur les ressources principales. Dans chaque graphe $\mathcal{G}^*_{ij}$ correspondant à l'une des quatre configurations décrites au paragraphe III.4.2, l'opération (i,j) possède une unique opération (de préparation) précédente et une unique opération (de préparation) suivante. Soit le graphe $\tilde{\mathcal{G}}_{ij}$ issu de la suppression de l'opération (i,j) dans le graphe $\mathcal{G}^*_{ij}$. Cette suppression se traduit par la génération d'un arc de type ressource $\tilde{u}(i,j)$ entre l'unique opération précédente et l'unique opération suivante.

Soit le graphe $\tilde{\mathcal{G}}$ issu des suppressions nécessaires à l'insertion de (i,j) dans $\mathcal{G}$ et de l'insertion du graphe $\tilde{\mathcal{G}}_{ij}$. Cette manipulation consiste en fait à préparer les ressources principales pour recevoir l'opération (i,j) sans insérer (i,j). Dans $\tilde{\mathcal{G}}$, l'arc $\tilde{u}(i,j)$ est associé à la date de début au plus tôt r^{*b}_{ij}, à la date de fin au plus tard d^{*b}_{ij} et l'occupation des ressources lb_{ij}. Il caractérise la position d'insertion $\mathcal{PI}(b)$ de (i,j). Toute clique $\tilde{\mathcal{C}}$ d'arcs de type ressource de $\tilde{\mathcal{G}}$ contenant cet arc $\tilde{u}(i,j)$ et telle que

- $l_{\tilde{\mathcal{C}}} \supseteq l_{ij}$ et
- $\tilde{\mathcal{C}}$ est compatible avec (i,j)

est une clique suffisante pour l'insertion de (i,j).

Soit $\mathcal{PI}' = \mathcal{PI}'(b) \cup \mathcal{PI}'(c)$ une autre position d'insertion. Si $\mathcal{PI}'(b) = \mathcal{PI}(b)$ alors on peut caractériser la position d'insertion $\mathcal{PI}'$ par une clique d'arcs de type ressource du graphe $\tilde{\mathcal{G}}$ défini ci-dessus. Sinon, on peut caractériser la position d'insertion $\mathcal{PI}'$ par une clique d'arcs de type ressource d'un graphe $\tilde{\mathcal{G}}'$ différent de $\tilde{\mathcal{G}}$ et correspondant à la position d'insertion $\mathcal{PI}'(b)$.

III.4.3.b) exemple

Dans la figure III.25, une opération d'exécution a est ordonnancée sur deux ressources principales $k1$ et $k2$ et sur les trois lignes de charges d'une ressource complémentaire $c3$ à 3 lignes de charge.

a et b sont consécutives sur $k1$. b utilise une autre ressource principale non représentée sur la figure. a et b sont aussi consécutives sur une ligne de charge de la ressource complémentaire $c3$. a et c sont consécutives sur $k2$. c utilise une autre ressource principale non représentée sur la figure. Cet ordonnancement est représenté par le graphe $\mathcal{G}$ où on voit apparaître notamment les arc u_{ab}, u_{af} et u_{ag} d'occupations des ressources $l_{ab} = l_{af} = l_{ag} = \{0,0,1\}$.

On souhaite insérer dans $\mathcal{G}$ une opération (i,j) telle que $e_{ij}^{k1} = 1$, $e_{ij}^{k2} = 1$ et $e_{ij}^{c3} = 2$. Cette opération utilise les mêmes ressources principales que a.

Soit la position d'insertion $\mathcal{PI}1 = \{(k1,1,a,b),(k2,1,a,c),(c3,1,a,b),(c3,2,a,f)\}$. On peut décomposer cette position d'insertion en $\mathcal{PI}1(b) = \{(k1,1,a,b),(k2,1,a,c)\}$ et $\mathcal{PI}1(c) = \{(c3,1,a,b),(c3,2,a,f)\}$. A $\mathcal{PI}1(b)$ correspond le graphe $\tilde{\mathcal{G}}1$ obtenu par les modifications correspondant à la configuration **i3** du paragraphe III.4.2. L'arc $\tilde{u}_1(i,j)$ caractérisant la position d'insertion $\mathcal{PI}1(b)$ a pour origine l'opération de remise à l'état initial sct_{ij}^* et pour destination l'opération de démontage sd_{ij}^*. Les arcs $\tilde{u}_{ab}, \tilde{u}_{af}$ et $\tilde{u}_{ag}$ sont d'occupation des ressources inchangée par rapport aux arcs u_{ab}, u_{af} et u_{ag}. L'ensemble d'arcs de type ressource $\tilde{\mathcal{C}}1 = \{\tilde{u}_1(i,j), \tilde{u}_{ab}, \tilde{u}_{af}\}$ caractérise la position d'insertion $\mathcal{PI}1$. On peut vérifier sur $\tilde{\mathcal{G}}1$ que les conditions de compatibilité par rang des arcs de $\tilde{\mathcal{C}}1$ exprimées dans la définition 6 sont vérifiées. $\tilde{\mathcal{C}}1$ est donc une clique. De plus $l_{\tilde{\mathcal{C}}1} = (1,1,2) = l_{ij}$. Enfin $\tilde{\mathcal{C}}$ est compatible avec (i,j). $\mathcal{PI}1$ est donc une position d'insertion valide de (i,j).

Soit la position d'insertion $\mathcal{PI}2 = \{(k1,1,a,b),(k2,1,a,c),(c3,2,a,f),(c3,3,a,g)\}$. Comme on a $\mathcal{PI}2(b) = \mathcal{PI}1(b)$, on peut caractériser $\mathcal{PI}2$ par une clique d'arcs de type ressource de $\tilde{\mathcal{G}}1$. L'ensemble d'arcs de type ressource $\tilde{\mathcal{C}}2 = \{\tilde{u}_1(i,j), \tilde{u}_{ab}, \tilde{u}_{af}, \tilde{u}_{ag}\}$ caractérise la position $\mathcal{PI}2$. $\tilde{\mathcal{C}}2$ est une clique. De plus $l_{\tilde{\mathcal{C}}2} = (1,1,3) > l_{ij}$. Enfin $\tilde{\mathcal{C}}2$ est compatible avec (i,j). $\mathcal{PI}2$ est donc une position d'insertion suffisante de (i,j).

Soit la position d'insertion $\mathcal{PI}3 = \{(k1,1,a,b),(k2,1,c,d),(c3,2,a,f),(c3,3,a,g)\}$. On a $\mathcal{PI}3(b) = \{(k1,1,a,b),(k2,1,c,d)\} \neq \mathcal{PI}1(b)$. On ne peut pas caractériser $\mathcal{PI}3$ par une clique d'arcs de type ressource de $\tilde{\mathcal{G}}1$. A $\mathcal{PI}3(b)$ correspond le graphe $\tilde{\mathcal{G}}3$ obtenu par les modifications correspondant à la configuration **ii** du paragraphe III.4.2. L'arc $\tilde{u}_3(i,j)$ caractérisant la position $\mathcal{PI}3(b)$ a dans ce cas pour origine l'opération de montage smt^*_{ij} et pour destination l'opération de démontage sd^*_{ij}. Les arcs $\tilde{u}_{ab}, \tilde{u}_{af}$ et $\tilde{u}_{ag}$ sont toujours d'occupation des ressources inchangée par rapport aux arcs u_{ab}, u_{af} et u_{ag}. On montre de la même manière que l'ensemble d'arcs de type ressource $\tilde{\mathcal{C}}3 = \{\tilde{u}_3(i,j), \tilde{u}_{af}, \tilde{u}_{ag}\}$ est une clique compatible avec (i,j) et d'occupation des ressources $l_{\tilde{\mathcal{C}}3} = l_{ij}$. $\mathcal{PI}3$ est donc une position d'insertion valide de (i,j).

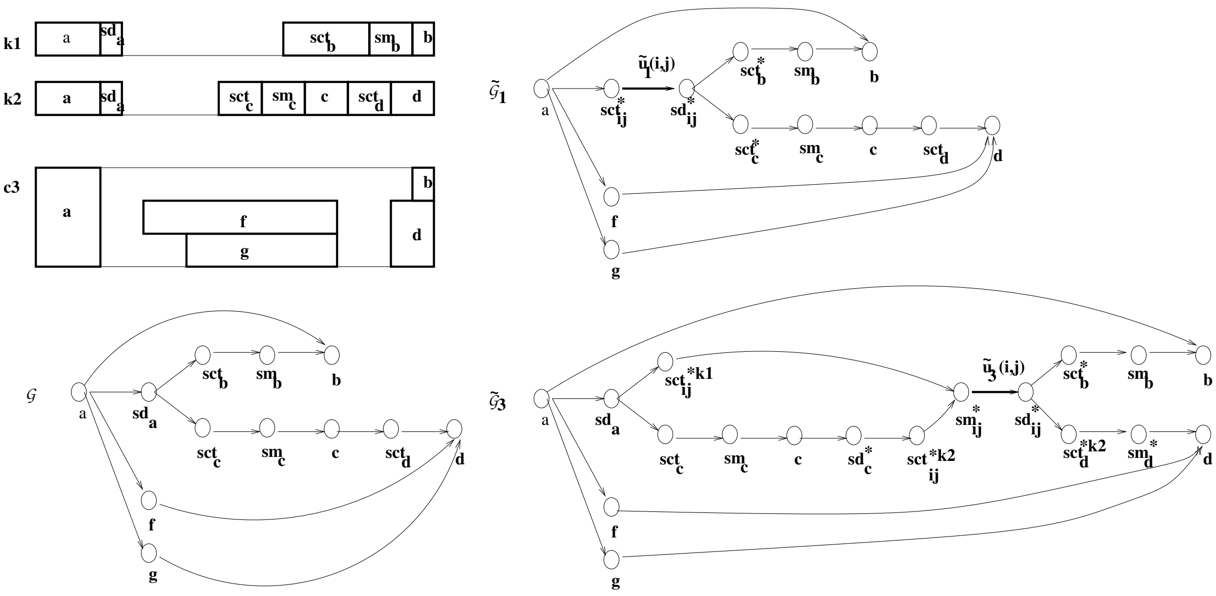


FIG. III.25 – Graphes correspondant à des positions d'insertion sur les ressources principales

III.4.3.c) Méthode de recherche d'une position d'insertion optimale

La recherche d'une position d'insertion optimale de $\mathcal{G}$ est équivalente à la recherche d'une clique suffisante optimale qu'on peut déterminer de la façon suivante. Pour chaque position d'insertion $\mathcal{PI}(b)$ possible pour (i,j) sur ses ressources principales $\mathcal{R}b_{ij}$, on peut générer le graphe $\tilde{\mathcal{G}}$ correspondant et énumérer les cliques maximales d'arcs de type ressource de chaque graphe $\tilde{\mathcal{G}}$ contenant l'arc $\tilde{u}(i,j)$. On peut appliquer à chaque clique maximale la procédure d'exploration des sous-cliques dominantes EXPLSOUSCLIQUES-DOMIN.

On peut remarquer qu'il n'est en fait pas nécessaire de générer les graphes $\tilde{\mathcal{G}}$ pour obtenir une clique suffisante $\tilde{\mathcal{C}}$. En effet, les opérations de préparation n'étant séquencées que sur les ressources principales, les modifications du graphe $\mathcal{G}$ pour créer le graphes $\tilde{\mathcal{G}}$ ne modifient que les arcs de type ressource d'occupation des ressources non nulle sur chaque ressource principale $k \in \mathcal{R}b_{ij}$ et situées sur le chemin entre chaque paire d'opération $(p(k,1), f(k,1))$ telle que $(k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)$. Autrement dit, tout arc u_α de type ressource de $\mathcal{G}$ et tel que $\exists c \in \mathcal{R}c$, $e^c_\alpha > 0$ a la même occupation des ressources dans $\tilde{\mathcal{G}}$ et dans $\tilde{\mathcal{G}}'$. Une clique suffisante $\tilde{\mathcal{C}}$ est parfaitement définie à partir de $\mathcal{G}$ sans générer $\tilde{\mathcal{G}}$ par :

- une clique $\mathcal{C}(c)$ du graphe $\mathcal{G}u_{\mathcal{R}c_{ij}}$ de date de début au plus tôt $r_{\mathcal{C}(c)}$, de date de fin au plus tard $d_{\mathcal{C}(c)}$ et d'occupation $l_{\mathcal{C}(c)} \geq lc_{ij}$,
- une position d'insertion $\mathcal{PI}(b)$ valide sur $\mathcal{R}b_{ij}$ et définissant l'arc fictif $\tilde{u}(i,j)$ de date de début au plus tôt r^{*b}_{ij}, de date de fin au plus tard d^{*b}_{ij} et d'occupation lb_{ij}

La condition à vérifier pour que $\tilde{\mathcal{C}}$ soit une clique du graphe $\tilde{\mathcal{G}}u_{\mathcal{R}_{ij}}$ est la compatibilité entre les positions d'insertions $\mathcal{PI}(b)$ et $\mathcal{PI}(\mathcal{C}) = \mathcal{PI}(c)$.

On associe à $\tilde{\mathcal{C}}$ une date de début au plus tôt $r_{\tilde{\mathcal{C}}} = \max(r^{*b}_{ij}, r_{\mathcal{C}(c)})$, une date de fin au plus tard $d_{\tilde{\mathcal{C}}} = \min(d^{*b}_{ij}, d_{\mathcal{C}(c)})$, une occupation $l_{\tilde{\mathcal{C}}} = l_{ij}$ et un impact sur l'admissibilité égal à $\Delta d_{max}(i,j,\mathcal{PI}(\tilde{\mathcal{C}})) = \max\{0, \max(r_{ij}, r_{\tilde{\mathcal{C}}}) + p_{ij} - \min(d_{ij}, d_{\tilde{\mathcal{C}}})\}$.

On donne ainsi le principe de la recherche d'une clique suffisante optimale pour l'insertion de (i,j) d'occupation l_{ij} sur un ensemble de ressources $\mathcal{R}_{ij}$.

Etape 1 Exploration des cliques d'occupation des ressources maximale

Etape 2 Pour chaque clique d'occupation des ressources maximale, recherche de la sous-clique d'impact minimal sur l'admissibilité en appliquant directement la procédure d'exploration des sous-cliques dominantes EXPLSOUSCLIQUESDOMIN.

L'Etape 1, ainsi que la procédure de recherche d'une clique suffisante optimale CLIQUEPRÉPAOPT sont détaillées dans le paragraphe III.4.4.

III.4.4 Procédure d'exploration des cliques maximales dominantes et procédure de recherche d'une clique suffisante optimale

On considère les relations de dominance entre cliques maximales énoncées dans le paragraphe III.2.5.a). On propose une procédure d'exploration des cliques maximales selon le principe énoncé en III.2.5 dans un contexte sans préparation. La procédure d'exploration des cliques maximales dominantes part d'une clique maximale dominante initiale et explore l'ensemble des cliques maximales dominantes jusqu'à une clique dominante finale par une série de déplacements élémentaires. La génération d'une clique maximale voisine à partir d'une clique maximale donnée est définie dans le paragraphe III.4.4.a). La procédure de génération de la clique maximale dominante initiale et la caractérisation des cliques qu'elle domine sont présentées dans le paragraphe III.4.4.b). La procédure d'exploration des cliques maximales dominantes jusqu'à la clique maximale dominante finale et la procédure de recherche d'une clique suffisante optimale CLIQUEPRÉPAOPT sont décrites dans le paragraphe III.4.4.c).

III.4.4.a) Génération d'une clique maximale voisine

Comme dans le paragraphe III.2.5.d) une clique maximale $\tilde{\mathcal{C}}_q$ est obtenue par déplacement vers la droite de la clique maximale précédente $\tilde{\mathcal{C}}_{q-1}$. Soit une clique maximale $\tilde{\mathcal{C}}$

caractérisant une position d'insertion $\mathcal{PI} = \mathcal{PI}(b) \cup \mathcal{PI}(c)$. Pour définir le déplacement vers la droite de $\tilde{\mathcal{C}}$ autour d'une opération o_{dest}, selon la nature de o_{dest}, trois cas sont à considérer :

1. S'il existe un arc $\tilde{u}_\alpha$ de $\tilde{\mathcal{C}}$ différent de $\tilde{u}(i,j)$ tel que $o_{dest} = dest_\alpha$, alors il s'agit d'un déplacement vers la droite sur les ressources complémentaires défini comme dans le paragraphe III.2.5.b) :
$$\tilde{\mathcal{C}} \stackrel{dd_{o_{dest}}}{\rightarrow} \tilde{\mathcal{C}}'$$
avec
$$\tilde{\mathcal{C}}' = dd_{o_{dest}}(\tilde{\mathcal{C}}) = (\tilde{\mathcal{C}} \setminus \tilde{\mathcal{U}}^{\mathcal{R}c_{ij}}_{entr}(o_{dest})) \cup \tilde{\mathcal{U}}^{\mathcal{R}c_{ij}}_{sort}(o_{dest})$$
où $\tilde{\mathcal{U}}^{\mathcal{R}c_{ij}}_{entr}(o_{dest})$ est le sous-ensemble des arcs u_α de $\tilde{\mathcal{G}}$ entrant dans o_{dest} tel que $l_\alpha \cap lc_{ij} \neq \emptyset_l$ et $\mathcal{U}^{\mathcal{R}c_{ij}}_{sort}(o_{dest})$ est le sous-ensemble des arcs u_β de $\tilde{\mathcal{G}}$ sortant de o_{dest} tel que $l_\beta \cap lc_{ij} \neq \emptyset_l$.

2. Si o_{dest} est une opération d'exécution telle que $\exists (k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)$, $o_{dest} = f(k,1)$ alors le déplacement vers la droite de $\tilde{\mathcal{C}}'$ correspond à une nouvelle position d'insertion $\mathcal{PI}'(b)$ sur les ressources principales. $\mathcal{PI}'(b)$ est obtenue à partir de $\mathcal{PI}(b)$ en conservant les 4-uplets $(k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)$ tels que $f(k,1) \neq o_{dest}$ et en remplaçant chaque 4-uplet $(k', 1, p(k',1), f(k',1)) \in \mathcal{PI}(b)$ tel que $f(k',1) = o_{dest}$ par le 4-uplet $(k', 1, o_{dest}, f'(k'))$ tel que o_{dest} et $f'(k')$ sont consécutives sur k'. A partir de cette nouvelle position d'insertion $\mathcal{PI}'(b)$, on obtient le graphe $\mathcal{G}^{*'}_{ij}$ et le graphe $\tilde{\mathcal{G}}'_{ij}$ contenant les opérations de préparation à insérer dans $\mathcal{G}$. Le graphe $\tilde{\mathcal{G}}'$ est obtenu par l'insertion de $\tilde{\mathcal{G}}'_{ij}$ dans $\mathcal{G}$. Cette position d'insertion de (i,j) sur les ressources principales est caractérisée à nouveau par un unique arc $\tilde{u}'(i,j)$.
$$\tilde{\mathcal{C}}' = dd_{o_{dest}}(\tilde{\mathcal{C}}) = (\tilde{\mathcal{C}} \setminus \{\tilde{u}(i,j)\}) \cup \{\tilde{u}'(i,j)\}$$

3. Une même opération o_{dest} peut correspondre aux deux hypothèses. Dans ce cas, on doit à la fois déplacer la clique vers la droite sur les ressources complémentaires autour de o_{dest} et changer la position d'insertion sur les ressources principales.

Pour une clique $\tilde{\mathcal{C}}$ donnée, l'algorithme de réalisation d'un déplacement vers la droite est en $\mathcal{O}(M_o^2)$ où M_o est un majorant du nombre de ressources occupées par une opération.

Un déplacement vers la droite est dit réalisable si l'ensemble des arcs de type ressource qui en sont issus constitue une clique maximale, c'est-à-dire si ces arcs sont compatibles deux à deux et si $l_{\tilde{\mathcal{C}}'} = l_{\tilde{\mathcal{C}}}$. On donne des conditions nécessaires et suffisantes pour qu'un déplacement vers la droite soit réalisable :

Théorème 11 *Un déplacement vers la droite $dd_{o_{dest}}$ d'une clique $\tilde{\mathcal{C}}$ est réalisable si et seulement si $\forall u_\alpha \in \tilde{\mathcal{C}} \setminus (\tilde{\mathcal{U}}^{\mathcal{R}c_{ij}}_{entr}(o_{dest}) \cup \tilde{u}_{ij})$ et $\forall (k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)$ tel que $f(k,1) \neq o_{dest}$, il n'existe pas de chemin dans $\mathcal{G}$ entre $dest_\alpha$ et o_{dest} (1) ni entre $f(k,1)$ et o_{dest} (2)*

Preuve Si o_{dest} est telle que $\forall(k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$, $o_{dest} \neq f(k,1)$ alors on reste dans le graphe $\tilde{\mathcal{G}}$. On connaît les conditions nécessaires et suffisantes de faisabilité d'un tel déplacement énoncées dans le théorème 4. Pour se référer à la preuve de ce théorème, on fait la remarque suivante: dans tout graphe possible $\tilde{\mathcal{G}}_{ij}$, l'ensemble des chemins issus de l'opération destination de $\tilde{u}(i,j)$ notée $dest_{\tilde{u}(i,j)}$ passe par une opération $f(k,1)$ telle que $(k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$. S'il n'existe pas de chemin dans $\mathcal{G}$ entre $f(k,1)$ et o_{dest} alors il n'existe pas non plus de chemin dans $\tilde{\mathcal{G}}$ entre $dest_{\tilde{u}(i,j)}$ et $f(k,1)$. Réciproquement, s'il existe un chemin dans $\mathcal{G}$ entre $f(k,1)$ et o_{dest} alors il existe aussi un chemin entre $dest_{\tilde{u}(i,j)}$ et o_{dest} dans $\tilde{\mathcal{G}}$. Ainsi la condition (2) équivaut à la non existence d'un chemin entre $dest_{\tilde{u}(i,j)}$ et o_{dest}. Si (1) et (2) sont vérifiées, on est exactement dans les conditions d'application du théorème 4. D'autre part, le théorème 5 assure que $\tilde{\mathcal{C}}'$ est une clique maximale.

Si o_{dest} est telle que $\exists(k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$, $o_{dest} = f(k,1)$, le déplacement vers la droite génère une nouvelle position d'insertion sur les ressources principales $\mathcal{PI}'(b)$. L'arc $\tilde{u}(i,j)$ est remplacé par un arc $\tilde{u}'(i,j)$. Si o_{dest} est en plus destination d'un arc $u_\alpha \in \tilde{\mathcal{C}}$ différent de $\tilde{u}_{ij}$, on se déplace aussi sur les ressources complémentaires. Dans ce cas d'après le théorème 4, (1) est une condition nécessaire et suffisante de validité du déplacement vers la droite de la clique $\tilde{\mathcal{C}}\backslash\{\tilde{u}(i,j)\}$. D'après le théorème 5, la clique obtenue $\tilde{\mathcal{C}}' \setminus \{\tilde{u}'(i,j)\}$ est aussi une clique maximale sur l'ensemble des ressources $\mathcal{R}c_{ij}$. Il suffit maintenant de montrer que (1) et (2) sont des conditions nécessaires et suffisantes pour que $\mathcal{PI}'(b)$ soit une position d'insertion valide pour (i,j) sur les ressources principales et que l'arc $\tilde{u}'(i,j)$ est compatible avec les arcs de $\tilde{\mathcal{C}}' \setminus \{\tilde{u}'(i,j)\}$.

Montrons que (2) est une condition nécessaire et suffisante pour que $\mathcal{PI}'(b)$ soit valide, $\mathcal{PI}(b)$ étant supposée valide. Si (2) n'est pas vérifiée, alors il existe un chemin entre une opération $f'(k,1)$ telle que $(k,1,p'(k,1),f'(k,1)) \in \mathcal{PI}'(b)$ avec $p'(k,1) \neq o_{dest}$ et l'opération $p'(k',1) = o_{dest}$ telle que $(k',1,p'(k',1),f'(k',1)) \in \mathcal{PI}'(b)$. $\mathcal{PI}'(b)$ n'est donc pas une position d'insertion valide. Si (2) est vérifiée, alors il n'existe pas de chemin entre toute opération $f'(k,1)$ telle que $(k,1,p'(k,1),f'(k,1)) \in \mathcal{PI}'(b)$ avec $p'(k,1) \neq o_{dest}$ et toute opération $p'(k',1)$ telle que $(k',1,p'(k',1),f'(k',1)) \in \mathcal{PI}'(b)$. Il n'existe pas non plus de chemin entre $f'(k,1)$ telle que $(k,1,p'(k,1),f'(k,1)) \in \mathcal{PI}'(b)$ avec $p'(k,1) = o_{dest}$ et toute opération $p'(k',1)$ telle que $(k',1,p'(k',1),f'(k',1)) \in \mathcal{PI}'(b)$. En effet, ceci implique l'existence d'un chemin entre o_{dest} et $p'(k',1) = p(k',1)$, ce qui est impossible puisque $\mathcal{PI}(b)$ est valide. $\mathcal{PI}'(b)$ étant une position d'insertion valide sur les ressources principales, l'occupation des ressources de l'arc $\tilde{u}'_{ij}$ caractérisant $\mathcal{PI}'(b)$ est égale à lb_{ij}. $\tilde{\mathcal{C}}'$ est ainsi d'occupation maximale des ressources sur $\mathcal{R}_{ij}$.

Montrons que l'arc $\tilde{u}'_{ij}$ caractérisant la position d'insertion valide $\mathcal{PI}'(b)$ est compatible avec les autres arcs u_α de $\tilde{\mathcal{C}}'$ si et seulement si (1) est vérifiée. On observe que pour tout graphe $\tilde{\mathcal{G}}'_{ij}$, tous les chemins passant par l'opération origine de l'arc $\tilde{u}'(i,j)$, notée $orig_{\tilde{u}'(i,j)}$ proviennent d'une opération $p'(k,1)$ telle que $(k,1,p'(k,1),f'(k,1)) \in \mathcal{PI}'(b)$. Il existe de plus un chemin passant par o_{dest}. (1) équivaut à la non existence d'un chemin entre $dest_\alpha$ et $orig_{\tilde{u}'(i,j)}$. Il ne peut pas exister de chemin entre $dest_{\tilde{u}'(i,j)}$ et $orig_\alpha$ car dans ce cas il existerait un chemin entre o_{dest} et $orig_\alpha$ et $\tilde{\mathcal{C}}$ ne serait pas une clique. $\square$.

Exemple de déplacement vers la droite Dans l'exemple de la figure III.25, on considère la clique suffisante $\tilde{\mathcal{C}}_0 = \{\tilde{u}_1(i,j), \tilde{u}_{ab}, \tilde{u}_{af}, \tilde{u}_{ag}\}$. Cette clique correspond à la position d'insertion sur les ressources principales $\mathcal{PI}1(b) = \{(k1,1,a,b),(k2,1,a,c)\}$. $\tilde{\mathcal{G}}1$ est le graphe correspondant. On considère tout d'abord le déplacement vers la droite autour de f. La condition (1) de faisabilité est vérifiée puisqu'aucun arc u_α de $\tilde{\mathcal{C}}_0 \backslash \{\tilde{u}_1(i,j)\}$ n'est tel qu'il existe un chemin entre $dest_\alpha$ (b ou g) et f. La condition (2) de faisabilité est vérifiée puisqu'aucune opération $f(k,1)$ telle que $(k,1,p(k,1),f(k,1)) \in \mathcal{PI}1(b)$ n'est telle qu'il existe un chemin entre $f(k,1)$ (ici b ou c) et f. f est tel que $\forall (k,1,p(k,1),f(k,1)) \in \mathcal{PI}1(b)$ $f(k,1) \neq f$. Aussi, le déplacement vers la droite conserve la position d'insertion $\mathcal{PI}1(b)$ et le graphe $\tilde{\mathcal{G}}1$. On obtient la clique $\mathcal{C}'_0 = \{\tilde{u}_1(i,j), \tilde{u}_{ab}, \tilde{u}_{fd}, \tilde{u}_{ag}\}$.

On considère maintenant le déplacement vers la droite de $\tilde{\mathcal{C}}_0$ autour de c. On peut vérifier comme dans le cas précédent que les conditions (1) et (2) sont vérifiées. Le déplacement est donc réalisable. Contrairement au cas précédent, c est tel que $\exists (k,1,p(k,1),f(k,1)) \in \mathcal{PI}1(b)$ tel que $f(k,1) = c$. En effet le 4-uplet $(k2,1,a,c)$ de $\mathcal{PI}1(b)$ vérifie cette condition. Aussi, le déplacement modifie la position d'insertion et le 4-uplet précédent est remplacé par $(k2,1,c,d)$. On retrouve la position d'insertion sur les ressources principales $\mathcal{PI}3(b) = \{(k1,1,a,b),(k2,1,c,d)\}$ dont le graphe correspondant $\tilde{\mathcal{G}}3$ est représenté. On obtient la clique maximale $\mathcal{C}3 = \{\tilde{u}3(i,j), \tilde{u}_{af}, \tilde{u}_{ag}\}$.

Relations temporelles entre cliques voisines Soit $\tilde{\mathcal{C}}$ une clique maximale sur $\mathcal{R}_{ij}$ contenant un arc $\tilde{u}_{ij}$ correspondant à une position d'insertion $\mathcal{PI}(b)$ sur les ressources principales $\mathcal{R}b_{ij}$ et une clique $\mathcal{C}(c)$ d'occupation maximale sur les ressources complémentaires $\mathcal{R}c_{ij}$. Soit $\tilde{\mathcal{C}}'$ la clique obtenue par un déplacement vers la droite de $\mathcal{C}$ autour d'une opération o_{dest}. $\tilde{\mathcal{C}}'$ contient un arc $\tilde{u}'_{ij}$ correspondant à une position d'insertion $\mathcal{PI}'(b)$ sur les ressources principales $\mathcal{R}b_{ij}$ et une clique $\mathcal{C}'(c)$ d'occupation maximale sur les ressources complémentaires $\mathcal{R}c_{ij}$.

Si $\forall (k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$, $o_{dest} \neq f(k,1)$ alors $\tilde{\mathcal{C}}'$ est telle que $\tilde{u}'_{ij} = \tilde{u}_{ij}$ et on sait que $r_{\mathcal{C}'(c)} > r_{\mathcal{C}(c)}$ et que $d_{\mathcal{C}'(c)} > d_{\mathcal{C}(c)}$ d'après la conséquence de la remarque 2. D'où $r_{\tilde{\mathcal{C}}'} > r_{\tilde{\mathcal{C}}}$ et $d_{\tilde{\mathcal{C}}'} > d_{\tilde{\mathcal{C}}}$.

Peut-on établir de telles relations si $\exists (k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$, $o_{dest} = f(k,1)$? Dans ce cas, on a $\tilde{u}'_{ij} \neq \tilde{u}_{ij}$. Peut-on établir des relations systématiques entre r^{*b}_{ij} et $r^{*b'}_{ij}$ d'une part, et entre d^{*b}_{ij} et $d^{*b'}_{ij}$ d'autre part ?

On donne pour cela dans le théorème 12 un ensemble de relations temporelles entre deux 4-uplets d'opérations d'exécution adjacents. On dit que deux 4-uplets $(k,l,p(k,l),f(k,l))$ et $(k,l,p'(k,l),f'(k,l))$ sont adjacents sur la ligne de charge l de la ressource k si $f(k,l) = p'(k,l)$ ou si $f'(k,l) = p(k,l)$. En effet, le déplacement vers la droite d'une clique $\tilde{\mathcal{C}}$ caractérisant la position d'insertion $\mathcal{PI}(b)$ sur les ressources principales, autour d'une opération $f(k,l)$ telle que $(k,l,p(k,l),f(k,l)) \in \mathcal{PI}(b)$ revient à remplacer un (ou plusieurs) 4-uplet(s) par un (ou plusieurs) 4-uplet(s) adjacent(s) sur une ressource principale k. Aussi, on va comparer dans tous les cas possibles de préparation (voir paragraphe III.4.2) la contribution d'un 4-uplet $(k,1,a,b) \in \mathcal{PI}$ dans les dates r^{*b}_{ij} et d^{*b}_{ij} à un 4-uplet adjacent $(k,1,b,c) \in \mathcal{PI}'$ dans les dates $r^{*b'}_{ij}$ et $d^{*b'}_{ij}$. Ces relations temporelles sont basées sur les inégalités valides présentées en III.4.1. Les temps de préparation ajoutés dans les relations concernant les dates de début au plus tôt correspondent aux temps nécessaires entre a et (i,j) si on insère (i,j) dans le 4-uplet $(k,1,a,b)$, ce qui correspond aux termes de gauche des relations, et entre b et (i,j), si on insère (i,j) dans le 4-uplet adjacent $(k,1,b,c)$, ce qui correspond aux termes de droite des relations. Les temps de préparation ajoutés dans les relations concernant les dates de fin au plus tard correspondent aux temps nécessaires entre $(i,j))$ et b si on insère (i,j) dans le 4-uplet $(k,1,a,b)$, ce qui correspond aux termes de droite des relations, et entre (i,j) et c, si on insère (i,j) dans le 4-uplet adjacent $(k,1,b,c)$, ce qui correspond aux termes de gauche des relations. Dans toutes ces relations la ressource guidante de préparation (g ou g') est une des ressources principales de l'association concernée, déterminée par le mode étendu de cette association.

Théorème 12 *Soient deux 4-uplets adjacents $(k,1,a,b)$ et $(k,1,b,c)$ et une opération (i,j) à insérer sur l'ensemble de ressources principales $\mathcal{R}b_{ij}$ telles que $k \in \mathcal{R}b_{ij}$. Les relations suivantes sont vérifiées :*

$$\begin{aligned}
r_b + p_b + p_{sd}^{\delta(\mathcal{R}b_b)g1} + p_{sct}^{\tau(k),RT_bRT_{ij}k1} &> r_a + p_a + p_{sd}^{\delta(\mathcal{R}b_a)g'1} + p_{sct}^{\tau(k),RT_aRT_{ij}k1} && \text{(III.24)}\\
r_b + p_b + p_{sd}^{\delta(\mathcal{R}b_b)g1} + p_{sct}^{\tau(k),RT_bRT_{ij}k1} &> r_a + p_a + p_{sct}^{\tau(\mathcal{R}b_a),RT_aRT_{ij}g'1} && \text{(III.25)}\\
r_b + p_b + p_{sct}^{\tau(\mathcal{R}b_b),RT_bRT_{ij}g1} &> r_a + p_a + p_{sd}^{\delta(\mathcal{R}b_a)g'1} + p_{sct}^{\tau(k),RT_aRT_{ij}k1} && \text{(III.26)}\\
r_b + p_b + p_{sct}^{\tau(\mathcal{R}b_b),RT_bRT_{ij}g1} &> r_a + p_a + p_{sct}^{\tau(\mathcal{R}b_a),RT_aRT_{ij}g'1} && \text{(III.27)}
\end{aligned}$$

$$\begin{aligned}
d_c - p_c - p_{sm}^{\mu(\mathcal{R}b_c)g1} - p_{sct}^{\tau(k),RT_{ij}RT_ck1} &> d_b - p_b - p_{sm}^{\mu(\mathcal{R}b_b)g'1} - p_{sct}^{\tau(k),RT_{ij}RT_bk1} && \text{(III.28)}\\
d_c - p_c - p_{sm}^{\mu(\mathcal{R}b_c)g1} - p_{sct}^{\tau(k),RT_{ij}RT_ck1} &> d_b - p_b - p_{sct}^{\tau(\mathcal{R}b_b),RT_{ij}RT_bg'1} && \text{(III.29)}\\
d_c - p_c - p_{sct}^{\tau(\mathcal{R}b_c),RT_{ij}RT_cg1} &> d_b - p_b - p_{sm}^{\mu(\mathcal{R}b_b)g'1} - p_{sct}^{\tau(k),RT_{ij}RT_bk1} && \text{(III.30)}\\
d_c - p_c - p_{sct}^{\tau(\mathcal{R}b_c),RT_{ij}RT_cg1} &> d_b - p_b - p_{sct}^{\tau(\mathcal{R}b_b),RT_{ij}RT_bg'1} && \text{(III.31)}
\end{aligned}$$

Preuve Démontrons la relation III.25. On rappelle que $\mathcal{R}b_b$ est l'ensemble des ressources principales de l'opération d'exécution b. $\delta(\mathcal{R}b_b)$ est le mode étendu de démontage de l'association $\mathcal{R}b_b$. $p_{sd}^{\delta(\mathcal{R}b_b)g1}$ est le temps de démontage de $\mathcal{R}b_b$ sur la ressource guidante de démontage g. On a $g \in \mathcal{R}b_b$ puisque $\mathcal{P}c_{sct}^{\delta(\mathcal{R}b_b)} = \emptyset$: il n'y a pas de ressources complémentaires de démontage. $\tau(k)$ est le mode de changement de type de la ressource isolée k. $p_{sct}^{\tau(k),RT_bRT_{ij}k1}$ est le temps de changement de type de la ressource isolée k du type RT_b de l'opération b vers le type RT_{ij} de l'opération (i,j). k est la ressource guidante de changement de type puisqu'il n'y a pas de ressources complémentaires de changement de type: $\mathcal{P}c_{sct}^{\tau(k)} = \emptyset$. Deux configurations sont à envisager pour a et b, opérations d'exécution successives sur la ressource principale k.

- Si $\mathcal{R}b_a = \mathcal{R}b_b$ et si a et b sont consécutives sur toutes les ressources principales de $\mathcal{R}b_a$ alors le montage et le démontage sont inutiles entre a et b. La relation suivante est ainsi vérifiée dans $\mathcal{G}$:

  $$r_b \geq r_a + p_a + p_{sct}^{\tau(\mathcal{R}b_a)RT_aRT_bg'1} \;\; avec \; g' \in \mathcal{R}b_a$$

  Or, $p_b > 0$ et $\mathcal{R}b_a = \mathcal{R}b_b$, d'où :

  $$r_b + p_b + p_{sd}^{\delta(\mathcal{R}b_b)g1} > r_a + p_a + p_{sd}^{\delta(\mathcal{R}b_a)g1} + p_{sct}^{\tau(\mathcal{R}b_a)RT_aRT_bg'1} \;\; avec \; (g,g') \in \mathcal{R}b_a \times \mathcal{R}b_a$$

On sait de plus d'après la troisième inégalité valide énoncée dans le paragraphe III.4.1, que $p_{sct}^{\tau(\mathcal{R}b_a)RT_aRT_bg'1} \geq p_{sct}^{\tau(k)RT_aRT_bk1}$ puisque $k \in \mathcal{R}b_a$, d'où :

$$r_b + p_b + p_{sd}^{\delta(\mathcal{R}b_b)g1} > r_a + p_a + p_{sd}^{\delta(\mathcal{R}b_a)g1} + p_{sct}^{\tau(k)RT_aRT_bk1} \; avec \; g \in \mathcal{R}b_a$$

Enfin l'inégalité triangulaire énoncée dans le paragraphe III.4.1 impose que :

$$p_{sct}^{\tau(k)RT_aRT_{ij}k1} \leq p_{sct}^{\tau(k)RT_aRT_bk1} + p_{sct}^{\tau(k)RT_bRT_{ij}k1}(A)$$

On obtient ainsi la relation III.25 cherchée.

– Si a et b ne sont pas consécutives sur l'ensemble de leurs ressources principales, alors le démontage de l'association $\mathcal{R}b_a$ et le montage de l'association $\mathcal{R}b_b$ sont nécessaires entre a et b. La relation suivante est ainsi vérifiée :

$$r_b \geq r_a + p_a + p_{sd}^{\delta(\mathcal{R}b_a)g1} + p_{sct}^{\tau(k)RT_aRT_bk1} + p_{sm}^{\mu(\mathcal{R}b_b)g''1} \; avec \; g \in \mathcal{R}b_a, g'' \in \mathcal{R}b_b$$

où $\mu(\mathcal{R}b_b)$ est le mode de montage de l'association $\mathcal{R}b_b$. Comme $p_{sm}^{\mu(\mathcal{R}b_b)g''1} \geq 0$, $p_{sd}^{\delta(\mathcal{R}b_b)g'1} \geq 0$, $p_b > 0$ et l'inégalité triangulaire (A) est vérifiée, on obtient ainsi la relation III.25 cherchée.

Des arguments similaires s'appuyant sur les inégalités énoncées dans le paragraphe III.4.1 peuvent être utilisés pour démontrer les autres inégalités. □

Ces relations montrent que pour toutes les configurations possibles entre 4-uplets adjacents, la date de début au plus tôt de (i, j) après insertion dans la clique $\tilde{\mathcal{C}}'$ est supérieure ou égale à la date de début au plus tôt de (i, j) après insertion dans la clique $\tilde{\mathcal{C}}$. La date de fin au plus tard de (i, j) pour le respect de l'admissibilité si on l'insère dans $\tilde{\mathcal{C}}'$ est supérieure ou égale à la date de fin au plus tard de (i, j) pour le respect de l'admissibilité si on l'insère dans la clique $\tilde{\mathcal{C}}$.

Conditions suffisantes pour les déplacements réalisables On montre avec des arguments similaires à la preuve du théorème 11 que les conditions suffisantes de faisabilité d'un déplacement énoncées au paragraphe III.2.5.b) dans le théorème 8 sont valides dans le contexte considéré ici.

III.4.4.b) Génération d'une clique maximale dominante initiale

Dans le paragraphe III.2.5.c), la procédure GÉNÈRECLIQUEMAXINIT part de la clique initiale $\mathcal{C}_{init}^{\mathcal{R}_{ij}}$ et applique une série de déplacements vers la droite réalisables autour de l'opération destination de plus petite date de fin au plus tôt. D'après la proposition 2 du paragraphe III.2.5.c), la condition d'arrêt utilisée est aussi suffisante pour assurer la compatibilité de la clique $\mathcal{C}_0$ ainsi obtenue avec l'opération (i, j). Le contre-exemple suivant montre que la clique initiale obtenue en adaptant la procédure GÉNÈRECLIQUEMAXINIT à la préparation n'est pas dominante en présence d'opérations de préparation et que la recherche de la clique initiale dominante nécessite l'utilisation des conditions nécessaires et suffisantes de compatibilité avec (i, j) énoncées au paragraphe III.2.4.a).

Contre-exemple On considère l'ordonnancement sur les ressources disjonctives $R1$ et $R2$ représenté sur la figure III.26. On considère que $R2$ est une ressource principale et $R1$ une ressource complémentaire. Un mode unique de changement de type est défini pour la ressource $R2$. Aucune ressource complémentaire de changement de type n'est requise. La durée de changement de type est donnée en fonction du type initial et du type final par la matrice représentée sur la figure. Sur $R2$ une opération $(2,1)$ telle que $p_{21} = 1$ et $RT_{21} = X$ est suivie d'une opération $(3,1)$ telle que $p_{31} = 1$ et $RT_{31} = Y$. Avant $(2,1)$, une opération de préparation sct_{21} de durée 1 est séquencée avant $(2,1)$ pour passer du type initial I au type X. Une opération de changement de type sct_{31} de durée 1 est nécessaire pour passer du type X au type Y. Les dates de début au plus tôt de ces opérations peuvent être lues sur la figure. Sur $R1$ une opération $(1,1)$ de date de début au plus tôt $r_{11} = 2$ et de durée $p_{11} = 1$ est séquencée. On considère l'opération $(1,2)$ de durée $p_{12} = 3$ et de type $RT_{12} = Z$ à insérer sur la ressource $R2$ avec $prec^1_{12} = \{(1,1)\}$. On a ainsi $r_{12} = 3$. Adaptons la procédure GÉNÈRECLIQUEMAXINIT à ce problème. On considère la position d'insertion initiale $\mathcal{PI}_O(b) = \{(R2, 1, O, (2,1))\}$. $(2,1)$ est telle que $r_{21}+p_{21} = 2$, donc $r_{21}+p_{21} < r_{12}$. La condition d'arrêt n'est pas vérifiée et ces relations temporelles ne garantissent pas qu'il n'existe pas de chemin entre $(2,1)$ et $(1,1)$. Considérons toutefois qu'un tel chemin n'existe pas et tentons d'insérer $(1,2)$ à la position $\mathcal{PI}_O(b)$. Le graphe $\tilde{\mathcal{G}}_{12}$ est formé de deux opérations : sct^*_{12} nécessaire pour passer du type initial I au type Z et sct^*_{21} nécessaire pour passer du type Z au type X. L'arc de type ressource $\tilde{u}_{12}$ relie ces deux opérations et caractérise la position d'insertion de $(1,2)$: $\tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init} = \{\tilde{u}_{12}\}$. La durée de l'opération sct^*_{12} étant de 3, on a $r^{*b}_{12} = 3$. De plus on a $d^{*b}_{12} = d_{21} - p_{21} - p_{sct^*_{21}} = 6$. $(1,2)$ n'ayant pas de ressources complémentaires on a ainsi $\Delta d_{max}(1, 2, \tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init}) = r^{*b}_{12} + p_{12} - d^{*b}_{12} = 0$.

Le déplacement vers la droite de $\tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init}$ donne la position d'insertion $\{(R2, 1, (2,1), (3,1))\}$ avec $r_{31} + p_{31} = 4 > r_{12}$. La condition d'arrêt de GÉNÈRECLIQUEMAXINIT est donc vérifiée et cette relation temporelle assure qu'il n'existe pas de chemin entre $(3,1)$ et $(1,1)$. La clique correspondant à cette position d'insertion serait la clique initiale retenue $\tilde{\mathcal{C}}_0$ par la procédure adaptée de GÉNÈRECLIQUEMAXINIT. Or par un calcul similaire au précédent on trouve que l'insertion de $(1,2)$ dans $\tilde{\mathcal{C}}_0$ a un impact sur l'admissibilité de $\Delta d_{max}(1, 2, \tilde{\mathcal{C}}_0) = 1 > \Delta d_{max}(1, 2, \tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init})$. $\tilde{\mathcal{C}}_0$ n'est pas dominante dans cet exemple. Par contre, si on rajoute une relation de précédence entre $(2,1)$ et $(1,1)$ (en considérant par exemple que $(2,1)$ est aussi ordonnancée sur R_1 juste avant $(1,1)$), la clique $\tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init}$ n'est plus compatible avec $(1,2)$, bien qu'on trouve toujours $\Delta d_{max}(1, 2, \tilde{\mathcal{C}}^{\mathcal{R}_{12}}_{init}) = 0$.

Description de la procédure Soit $\mathcal{C}^{\mathcal{R}c_{ij}}_{init}(c)$ la clique maximale initiale sur l'ensemble $\mathcal{R}c_{ij}$ des ressources complémentaires utilisées par (i,j). Soit $\mathcal{PI}_{init}(b)$ la position initiale d'insertion sur l'ensemble $\mathcal{R}b_{ij}$ des ressources principales utilisées par (i,j). Soit $\tilde{u}_{ijO}$ l'arc caractérisant la position $\mathcal{PI}_{init}(b)$. Soit $\tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{init} = \{\tilde{u}_{ijO}\} \cup \mathcal{C}^{\mathcal{R}c_{ij}}_{init}(c)$. La procédure GÉNÈRECLIQUEMAXINITPRÉPA proposée génère une clique maximale initiale $\tilde{\mathcal{C}}_0$ dominante en explorant une suite de cliques maximales en partant de la clique $\tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{init}$. De la même façon que la procédure GÉNÈRECLIQUEMAXINIT augmente progressivement la valeur $r_{\mathcal{C}_q}$, GÉNÈRECLIQUEMAXINITPRÉPA augmente la valeur $r_{\tilde{\mathcal{C}}_q}$ de telle sorte qu'à

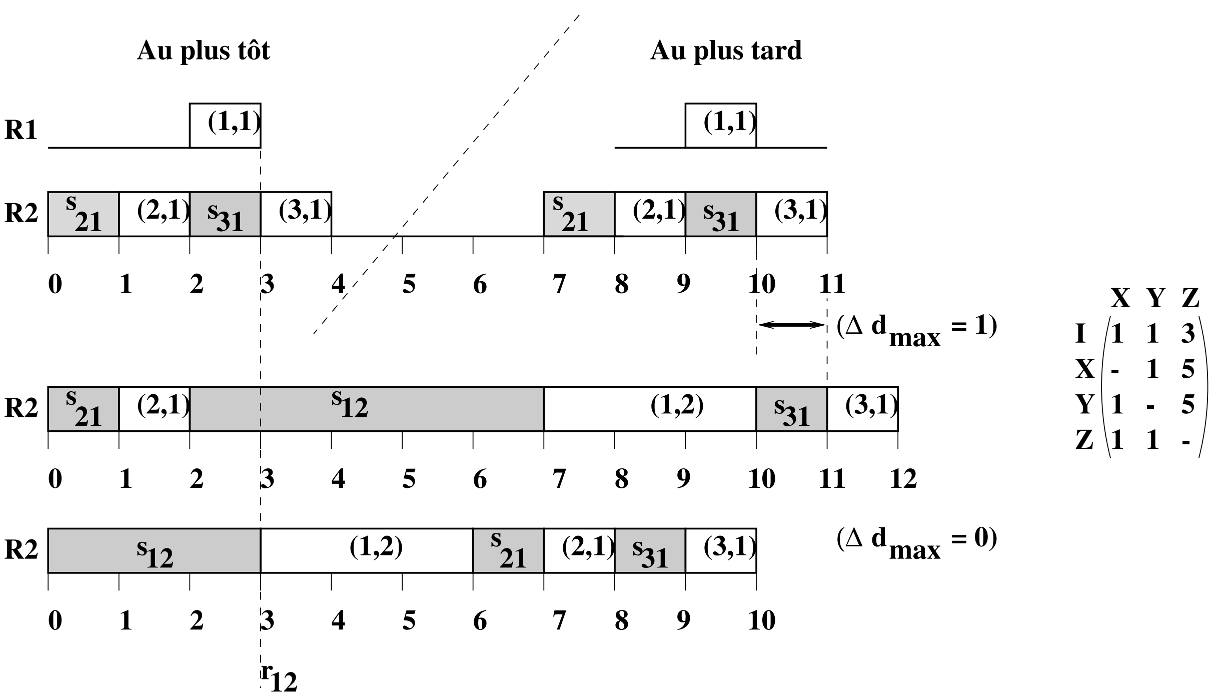


FIG. III.26 – Exemple d'insertion sur une ressource disjonctive avec des opérations de changement de type

chaque itération, l'augmentation soit minimale. Ce processus se poursuit jusqu'à ce qu'on ait $r_{\tilde{\mathcal{C}}_q} \leq r_{ij}$ et $r_{\tilde{\mathcal{C}}_{q+1}} > r_{ij}$, ce qui est la condition de dominance de la clique initiale. On rappelle que le théorème 12 a montré qu'on avait obligatoirement $r_{\tilde{\mathcal{C}}_{q+1}} \geq r_{\tilde{\mathcal{C}}_q}$. La différence avec le cas sans préparation va porter sur la façon de déterminer l'opération pivot et la condition d'arrêt de la procédure avec, dans les deux cas, la perte des conditions suffisantes pour que :

- le déplacement autour de l'opération pivot sélectionnée soit réalisable
- la clique obtenue après la condition d'arrêt soit compatible avec (i, j).

Le choix de l'opération o_{dest} pivot du déplacement doit assurer l'augmentation minimale de $r_{\tilde{\mathcal{C}}_q}$. Si o_{dest} est destination d'un arc de la clique $\mathcal{C}_q(c)$ sur les ressources complémentaires, la contribution de o_{dest} dans $r_{\mathcal{C}_{q+1}}$ est $r_{o_{dest}} + p_{o_{dest}}$. S'il existe une ressource principale k telle que $o_{dest} = f(k, 1)$ où $(k, 1, p(k, 1), f(k, 1))$ est un 4-uplet de la position d'insertion sur les ressources principales $\mathcal{PI}_q(b)$ alors la contribution de o_{dest} dans $r_{\mathcal{C}_{q+1}}$ dépend de la configuration de la position d'insertion obtenue après déplacement autour de o_{dest}. La procédure ChoixOdestMinΔr utilisée par GénèreCliqueMaxInitPrépa et décrite dans la figure III.27 sélectionne l'opération o_{dest} qui garantit l'augmentation minimale de la date de début au plus tôt de la clique par déplacement vers la droite et donne la contribution r_{min} de o_{dest} dans cette date. Cet algorithme est en $\mathcal{O}(K + C)$ où $C = |\mathcal{C}(c)|$ et $K = |\mathcal{PI}(b)|$.

Proc ChoixOdestMinΔr $(\mathcal{PI}(b), \mathcal{C}(c), o_{dest}, r_{min})$

Détermination de o^b_{dest} et r^b_{min} sur les ressources principales

Si $\forall (k,1,p(k,1),f(k,1)),(k',1,p(k',1),f(k',1)) \in \mathcal{PI}_q(b) \times \mathcal{PI}_q(b)$, $f(k,1) = f(k',1) = f$

(une seule opération pivot possible f, de même asssociation que (i,j))

$r^b_{min} \leftarrow r_f + p_f + p_{sct}^{\tau(\mathcal{R}b_{ij})RT_f RT_{ij} g1}$, où $g \in \mathcal{R}b_{ij}$

$o^b_{dest} \leftarrow f$

Sinon

Soit $f(k^*,1)$ telle que $r^b_{min} = r_{f(k^*,1)} + p_{f(k^*,1)} + p_{sd}^{\delta(\mathcal{R}b_{f(k^*,1)})g^*1} + p_{sct}^{\tau(k^*)RT_{f(k^*,1)}RT_{ij}}$

$= \min\limits_{k \in \mathcal{R}b_{ij}} r_{f(k,1)} + p_{f(k,1)} + p_{sd}^{\delta(\mathcal{R}b_{f(k,1)})g1} + p_{sct}^{\tau(k)RT_{f(k,1)}RT_{ij}}$ où $g, g^* \in \mathcal{R}b_{ij}$

$o^b_{dest} \leftarrow f(k^*,1)$

$r^b_{min} \leftarrow r^b_{min} + p_{sm}^{\mu(\mathcal{R}b_{ij})g1}$ où $g \in \mathcal{R}b_{ij}$

FinSi

Détermination de o^c_{dest} et r^c_{min} sur les ressources complémentaires

Soit $o^c_{dest} = dest_\alpha$ telle que $r^c_{min} = r_{dest_\alpha} + p_{dest_\alpha} = \min\limits_{u_\beta \in \mathcal{C}(c)} (r_{dest_\beta} + p_{dest_\beta})$

Si $r^b_{min} < r^c_{min}$ **alors**

$o_{dest} \leftarrow o^b_{dest}$; $r_{min} \leftarrow r^b_{min}$

Sinon

$o_{dest} \leftarrow o^c_{dest}$; $r_{min} \leftarrow r^c_{min}$

FinSi

FinProc

Fig. III.27 – Algorithme de ChoixOdestMinΔr

La façon dont on détermine o_{dest} n'est pas une condition suffisante pour que le déplacement soit réalisable, contrairement au cas sans préparation. En effet, lorsque $o_{dest} = f(k^*,1)$, il peut exister des opérations $f(k,1)$ telles que $r_{f(k,1)} + p_{f(k,1)} < r_{f(k^*,1)}$. Pour chacune de ces opérations, il est alors nécessaire de tester la non existence d'un chemin entre $f(k,1)$ et $f(k^*,1)$. La procédure TesteCheminOdest dont l'algorithme est décrit dans la figure III.28 renvoie si un tel chemin existe, l'opération o_{chemin} de plus petite date de début au plus tôt parmi les opérations $f(k,1)$ précédentes de $f(k^*,1)$. Son algorithme basé sur une exploration dans le graphe $\mathcal{G}$ est dans le pire des cas en $\mathcal{O}(u)$, où u est l'ensemble des arcs de $\mathcal{G}$. On teste d'abord une condition suffisante temporelle de non existence d'un tel chemin avant d'explorer $\mathcal{G}$.

Si une opération o_{chemin} est trouvée, le déplacement autour de o_{dest} n'est pas réalisable. On tente alors d'effectuer le déplacement en appliquant ChoixOdestMinΔr à la position $\mathcal{PI}(b)$ privée des 4-uplets d'opération $f(k,1)$ égale à o_{dest}. On itère cette procédure tant que le déplacement autour de l'opération o_{dest} sélectionnée n'est pas réalisable.

A l'issue de cette procédure, on a obtenu une clique maximale qui ne remplit pas forcément les conditions suffisantes de compatibilité avec (i,j) (cas du contre-exemple ci dessus). Aussi il est nécessaire de tester l'existence d'un chemin entre une des précédentes de (i,j) dans la gamme de i et une opération $f(k,1)$, $k \in \mathcal{R}b_{ij}$. La procédure Teste-CheminPrecij dont l'algorithme est décrit dans la figure III.29 renvoie, si un tel chemin existe, l'opération o_{chemin} de plus petite date de début au plus tôt parmi les opérations $f(k,1)$ situées sur un chemin vers une précédente de (i,j) dans la gamme de i. Son algo-

Proc TESTECHEMINODEST $(\mathcal{PI}(b), o_{dest}, \mathcal{G}, o_{chemin})$
CondSuffis $\leftarrow FAUX$
Si $((r_{o_{dest}} < \min_{(k,1,p(k,1),f(k,1))\in\mathcal{PI}(b)} r_{f(k,1)} + p_{f(k,1)})$ et
$(d_{o_{dest}} - p_{o_{dest}} > \max_{(k,1,p(k,1),f(k,1))\in\mathcal{PI}(b)} d_{f(k,1)})$ **alors**
CondSuffis $\leftarrow VRAI$
FinSi
$o_{chemin} \leftarrow \emptyset$
Si CondSuffis$= FAUX$ **alors**
Déterminer l'opération $f(k_{min},1)$ telle que $(k_{min},1,p(k_{min},1),f(k_{min},1)) \in \mathcal{PI}(b)$ et
$r_{f(k_{min},1)} = \min_{(r,1,p(r,1),f(r,1))\in\mathcal{PI}(b)} r_{f(r,1)}$
Rechercher un chemin dans $\mathcal{G}$ entre $f(k_{min},1)$ et o_{dest}
en explorant le graphe à rebours à partir de o_{dest}
si au cours de l'exploration, une opération
(f,l) est trouvée faire $o_{chemin} \leftarrow f(k,l)$
FinSi
FinProc

FIG. III.28 – Algorithme de TESTECHEMINODEST

rithme est en $\mathcal{O}(n_p u)$, où u est l'ensemble des arcs de $\mathcal{G}$ et n_p est le nombre de précédentes de (i,j) dans la gamme de i. Remarquons qu'avant d'explorer le graphe $\mathcal{G}$, on teste des conditions suffisantes de non-existence d'un chemin tirées de [Dauzère Péres et al, 1996].

Proc TESTECHEMINPRECIJ $(\mathcal{PI}(b), prec^i_{ij}, \mathcal{G}, o_{chemin})$
CondSuffis $\leftarrow FAUX$
Si $\forall(k,1,p(k,1),f(k,1)) \in \mathcal{PI}(b)$, $(r_{f(k,1)} + p_{f(k,1)} > \max_{(i,h)\in prec^i_{ij}} r_{ih}$ ou
$d_{f(k,1)} < \max_{(i,h)\in prec^i_{ij}} (d_{ih} - p_{ih}))$ **alors**
CondSuffis $\leftarrow VRAI$
FinSi
$o_{chemin} \leftarrow \emptyset$
Si CondSuffis$= FAUX$ **alors**
Déterminer l'opération $f(k_{min},1)$ telle que $(k_{min},1,p(k_{min},1),f(k_{min},1)) \in \mathcal{PI}(b)$ et
$r_{f(k_{min},1)} = \min_{(r,1,p(r,1),f(r,1))\in\mathcal{PI}(b)} r_{f(r,1)}$
Rechercher un chemin dans $\mathcal{G}$ entre $f(k_{min},1)$ et chaque opération $(i,h) \in prec^i_{ij}$
en explorant le graphe à rebours à partir de o_{dest}.
Si au cours de l'exploration, une opération
(f,l) est trouvée faire $o_{chemin} \leftarrow f(k,l)$
FinSi
FinProc

FIG. III.29 – Algorithme de TESTECHEMINPRECIJ

Si à l'issue du test, un chemin vers une précédente de (i,j) est déterminé, alors une série de déplacements vers la droite est nécessaire jusqu'à ce que le chemin disparaisse. On donne l'algorithme de la procédure GÉNÈRECLIQUEMAXINITPRÉPA ainsi définie dans la figure III.30.

Proc GÉNÈRECLIQUEMAXINITPRÉPA $(i, j, l_{ij}, r_{ij}, prec^i_{ij}, \mathcal{G}, \tilde{\mathcal{C}}_0)$
  $\tilde{\mathcal{C}}_0 \leftarrow \tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{init}$
  $cpt \leftarrow 0$
  CHOIXODESTMINΔR $(\mathcal{PI}_0(b), \mathcal{C}_0(c), o_{dest}, r_{min})$
  TESTECHEMINODEST $(\mathcal{PI}_0(b), o_{dest}, \mathcal{G}, o_{chemin})$
  $\mathcal{PI}_{temp}(b) \leftarrow \mathcal{PI}_0(b)$
  Tant que $o_{chemin} \neq \emptyset$ **faire**
    $\mathcal{PI}_{temp}(b) \leftarrow \mathcal{PI}_{temp}(b)$
    $\setminus \bigcup_{(k,1,p(k,1),f(k,1)) \in \mathcal{PI}_{temp}(b), f(k,1)=o_{dest}} (k,1,p(k,1),f(k,1))$
    CHOIXODESTMINΔR $(\mathcal{PI}_{temp}(b), \mathcal{C}_{temp}(c), o_{dest}, r_{min})$
    TESTECHEMINODEST $(\mathcal{PI}_0(b), o_{dest}, \mathcal{G}, o_{chemin})$
  FinTantque
  Tant que $r_{min} \leq r_{ij}$ et $\tilde{\mathcal{C}}_{cpt} \neq \tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{fin}$ **faire**:
    $cpt \leftarrow cpt + 1$
    $\tilde{\mathcal{C}}_{cpt} = dd_{o_{dest}}(\tilde{\mathcal{C}}_{cpt-1})$
    CHOIXODESTMINΔR $(\mathcal{PI}_{cpt}(b), \mathcal{C}_{cpt}(c), o_{dest}, r_{min})$
    TESTECHEMINODEST $(\mathcal{PI}_{cpt}(b), o_{dest}, \mathcal{G}, o_{chemin})$
    $\mathcal{PI}_{temp}(b) \leftarrow \mathcal{PI}_{cpt}(b)$
    Tant que $o_{chemin} \neq \emptyset$ **faire**
      $\mathcal{PI}_{temp}(b) \leftarrow \mathcal{PI}_{temp}(b)$
      $\setminus \bigcup_{(k,1,p(k,1),f(k,1)) \in \mathcal{PI}_{temp}(b), f(k,1)=o_{dest}} (k,1,p(k,1),f(k,1))$
      CHOIXODESTMINΔR $(\mathcal{PI}_{temp}(b), \mathcal{C}_{temp}(c), o_{dest}, r_{min})$
      TESTECHEMINODEST $(\mathcal{PI}_{cpt}(b), o_{dest}, \mathcal{G}, o_{chemin})$
    FinTantque
  FinTantque
  test de compatibilité avec (i,j)
  TESTECHEMINPRECIJ $(\mathcal{PI}(b), prec^i_{ij}, \mathcal{G}, o_{chemin})$
  Si $o_{chemin} \neq \emptyset$ **faire**
    effectuer des déplacements vers la droite
    jusqu'à ce que le chemin disparaisse
  FinSi
  $\mathcal{C}_0 \leftarrow \mathcal{C}_{cpt}$
FinProc

FIG. III.30 – Algorithme de GÉNÈRECLIQUEMAXINITPRÉPA

On montre avec des arguments similaires à la démonstration de la proposition 3 du paragraphe III.2.5.c) que la clique $\tilde{\mathcal{C}}_0$ générée par cette procédure domine toute clique maximale $\tilde{\mathcal{C}}'$ telle que $r_{\tilde{\mathcal{C}}'} \leq r_{\tilde{\mathcal{C}}_0}$. L'algorithme de cette procédure est en $\mathcal{O}(M_o^2 K u^2)$ où $K = |\mathcal{PI}(b)|$ et u est le nombre d'arcs de type ressource du graphe $\mathcal{G}$.

III.4.4.c) Exploration des cliques maximales dominantes

A partir de la clique maximale initiale déterminée au paragraphe précédent, on explore un ensemble de cliques maximales dominantes par une série de déplacements vers la droite, jusqu'à une clique maximale finale. Comme au paragraphe précédent, la différence avec le cas sans préparation va porter sur la façon de déterminer l'opération pivot du déplacement et la condition d'arrêt de la procédure avec dans les deux cas la perte des conditions suffisantes pour que :

- le déplacement autour de l'opération pivot sélectionnée soit réalisable
- la clique obtenue après la condition d'arrêt soit compatible avec (i, j).

Comme dans le paragraphe III.2.5.d), l'opération sélectionnée pour le déplacement est celle qui contraint le plus la date de fin au plus tard associée à la clique. La procédure ChoixOdestMind dont l'algorithme est décrit dans la figure III.31 détermine cette opération. Cet algorithme est en $\mathcal{O}(\max(K, C+1))$ où $K = |\mathcal{PI}(b)|$ et $C = |\mathcal{C}(c)|$:

Proc ChoixOdestMind $(\tilde{\mathcal{C}}, \mathcal{PI}(b), \mathcal{C}(c), o_{dest})$
Soit l'ensemble d'arcs $\mathcal{U}_{\min}$ et l'opération o_{dest} tels que
$\forall u_\alpha \in \mathcal{U}_{\min}\,,\ d_\alpha = \min_{u_\beta \in \tilde{\mathcal{C}}} d_\alpha$
Si $\mathcal{U}_{\min} = \{\tilde{u}_{ij}\}$ **alors**
(date contrainte par les ressources principales)
Si $\forall (k, 1, p(k,1), f(k,1)), (k', 1, p(k',1), f(k',1)) \in \mathcal{PI}_q(b) \times \mathcal{PI}_q(b)\,,\ f(k,1) = f(k',1) = f$
(un seul pivot possible)
$o^b_{dest} \leftarrow f$
Sinon
(choix du pivot le plus contraignant dans $\mathcal{G}^*_{ij}$)
Soit $f(k^*, 1)$ telle que $d_{f(k^*,1)} - p_{f(k^*,1)} - p_{sm}^{\mu(\mathcal{R}b_{f(k^*,1)})g1} - p_{sct}^{\tau(k^*)RT_{ij}RT_{f(k^*,1)}}$
$= \min_{k \in \mathcal{R}b_{ij}} d_{f(k,1)} - p_{f(k,1)} - p_{sm}^{\mu(\mathcal{R}b_{f(k,1)})g'1} - p_{sct}^{\tau(k)RT_{ij}RT_{f(k,1)}k1}$
$o^b_{dest} \leftarrow f(k^*, 1)$
FinSi
Sinon
(date contrainte par les ressources complémentaires)
Soit o_{dest} telle que $o_{dest} = dest_\alpha$ avec $u_\alpha \in \mathcal{U}_{\min}$ et
$\forall (k, 1, p(k,1), f(k,1) \in \mathcal{PI}(b)\,,\ f(k,1) \neq o_{dest}$
FinSi
FinProc

Fig. III.31 – Algorithme de ChoixOdestMind

Comme pour la génération de la clique maximale initiale, si $o_{dest} = f(k^*, 1)$ avec $k^* \in \mathcal{R}b_{ij}$, alors la condition suffisante pour que le déplacement soit réalisable n'est pas obligatoirement vérifiée. On applique alors la procédure TesteCheminOdest qui renvoie l'opération o_{chemin} de plus petite date de début au plus tôt parmi toutes les opérations $f(k, 1)$ précédentes de $f(k^*, 1)$, s'il en existe au moins une. Si un tel chemin existe, une série de déplacements vers la droite est effectuée jusqu'à ce que le chemin disparaisse.

Enfin, la condition garantissant la dominance de la clique finale ($d_{\tilde{\mathcal{C}}} \geq d_{ij}$) ne garantit plus la compatibilité de cette clique avec (i, j). Dans le cas où la condition suffisante n'est pas garantie, on peut utiliser la procédure TESTECHEMINSUCCIJ dont l'algorithme, symétrique à l'algorithme de TESTECHEMINPRECIJ, renvoie si un tel chemin existe, l'opération o_{chemin} de plus grande date de fin au plus tard parmi les opérations $p(k, 1)$ situées sur un chemin issu d'une suivante de (i, j) dans la gamme de i. Si un tel chemin existe, la procédure peut s'arrêter. On décrit la procédure CLIQUEPRÉPAOPT ainsi définie dans la figure III.32.

Proc CLIQUEPRÉPAOPT $((i,j), l_{ij}, r_{ij}, d_{ij}, \mathcal{G}, \mathcal{C}_{opt}, \Delta d^{opt}_{max})$
    GÉNÈRECLIQUEMAXINITPRÉPA $(i, j, l_{ij}, r_{ij}, prec^i_{ij}, \mathcal{G}, \tilde{\mathcal{C}}_0)$
    EXPLSOUSCLIQUESDOMIN $(\tilde{\mathcal{C}}_0, i, j, l_{ij}, r_{ij}, d_{ij}, \tilde{\mathcal{C}}_{opt}, \Delta d^{opt}_{max})$
    Si $\Delta d^{opt}_{max} = 0$ **alors**
        $STOP \leftarrow VRAI$
    Sinon
        $STOP \leftarrow FAUX$
    FinSi
    $cpt \leftarrow 0$
    compat $\leftarrow VRAI$
    Tantque $\mathcal{C}_{cpt} \neq \mathcal{C}^{\mathcal{R}_{ij}}_{fin}$ et $d_{\tilde{\mathcal{C}}_{cpt}} < d_{i,j}$ et compat $= VRAI$ et stop = FAUX
        CHOIXODESTMIND $(\tilde{\mathcal{C}}_{cpt}, \mathcal{PI}_{cpt}(b), \mathcal{C}_{cpt}(c), o_{dest})$
        TESTECHEMINODEST $(\mathcal{PI}_0(b), o_{dest}, \mathcal{G}, o_{chemin})$
        Si $o_{chemin} \neq \emptyset$ **alors**
            effectuer des déplacements vers la droite pour supprimer le chemin
        FinSi
        $cpt \leftarrow cpt + 1$
        $\tilde{\mathcal{C}}_{cpt} = dd_{o_{dest}}(\tilde{\mathcal{C}}_{cpt-1})$
        TESTECHEMINSUCCIJ$(\mathcal{PI}(b), prec^i_{ij}, \mathcal{G}, o_{chemin})$
        Si $o_{chemin} = \emptyset$ **alors**
            EXPLSOUSCLIQUESDOMIN $(\tilde{\mathcal{C}}_{cpt}, i, j, l_{ij}, C_{opt}, \Delta d^{opt}_{max})$
            Si $\Delta d^{opt}_{max} = 0$ **alors**
                $STOP \leftarrow VRAI$
            FinSi
        Sinon
            compat $\leftarrow FAUX$
        FinSi
    FinTantque
FinProc

FIG. III.32 – Algorithme de CLIQUEPRÉPAOPT

On peut montrer, avec des arguments similaires à ceux développés dans le paragraphe III.2.5.d), que la procédure CLIQUEPRÉPAOPT trouve la position d'insertion optimale d'une opération dans un ordonnancement cumulatif multi-ressources, avec des opérations de préparation, sans ressource complémentaire de préparation, et à occupation des ressources fixée. L'algorithme est en $\mathcal{O}(q^3u^2)$ où q est le nombre maximal d'arcs dans une clique et u le nombre d'arcs de type ressource du graphe $\mathcal{G}$.

III.5 Prise en compte conjointe de l'affectation et de la préparation sans aucune ressource complémentaire

III.5.1 Définition du problème et principe de résolution

On considère maintenant que l'opération d'exécution à insérer (i,j) a des possibilités d'affectation à la fois sur ses ressources principales et sur ses ressources complémentaires. Pour chaque mode étendu m, on rappelle qu'une ressource doit être sélectionnée au sein de chaque pool principal Pb_{ij}^{mr}, $r = 1, ..., Mb_{ij}^{m}$ et au sein de chaque pool complémentaire Pc_{ij}^{mr}, $r = 1, ..., Mb_{ij}^{m}$. Le nombre de lignes de charge e_{ij}^{k} que peut utiliser l'opération (i,j) sur une ressource complémentaire cimulative k dans le mode étendu m, est compris entre une borne inférieure e_{ij-}^{mk} et une borne supérieure e_{ij+}^{mk}. Par contre, on considère comme dans le paragraphe III.4 que les opérations de préparation des ressources principales ne nécessitent pas de ressources complémentaires. Pour tout mode étendu de montage μ, pour tout mode étendu de démontage δ, pour tout mode étendu de changement de type τ, $\mathcal{P}c_{sm}^{\mu} = \mathcal{P}c_{sd}^{\delta} = \mathcal{P}c_{sct}^{\tau} = \emptyset$.

Une position d'insertion optimale peut être trouvée en appliquant CliquePrépaOpt (voir paragraphe III.4) à chacune des occupations des ressources possibles pour (i,j).

L'objectif de ce paragraphe est l'extension de la procédure de recherche d'une clique suffisante optimale CliqueOccupOpt qui permet d'éviter l'énumération complète des occupations des ressources dans le cas sans préparation et mode étendu fixé (voir paragraphe III.3). Le principe de CliqueOccupOpt est d'explorer un ensemble de cliques d'occupation des ressources maximale sur l'ensemble de toutes les ressources possibles pour un mode étendu donné.

Parmi les positions d'insertions explorées par CliquePrépaOpt sur les ressources principales (arcs $\tilde{u}(i,j)$) on distingue :

- les positions d'insertion pour lesquelles l'association $\mathcal{R}b_{ij}$ est constituée (cas de figure i du paragraphe III.4.2)
- les positions d'insertions pour lesquelles l'association $\mathcal{R}b_{ij}$ est à constituer (cas de figure ii du paragraphe III.4.2)

Pour proposer une procédure optimale alternative à l'énumération, on considère dans ce paragraphe une restriction supplémentaire : l'opération (i,j) à insérer ne possède pas de ressources complémentaires dans le mode étendu considéré, ce qui se traduit par $\mathcal{P}b_{ij}^{m} = \emptyset$. Dans ce cas, on considère uniquement les positions d'insertion valides sur les ressources principales. L'impact de l'insertion de (i,j) dans une position d'insertion valide $\mathcal{PI}(b)$ s'exprime par :

$$\Delta d_{max}(i,j,\mathcal{PI}(b),l_{ij}) = \max(0, \max(r_{ij}, r_{ij}^{*b}) + p_{ij}(l_{ij}) - \min(d_{ij}, d_{ij}^{*b})) \tag{III.32}$$

La procédure proposée décompose dans ce contexte la recherche d'une position d'insertion optimale en deux parties pour tirer parti des possibilités d'affectation liées à un mode étendu :

1. exploration des positions d'insertion correspondant à des associations possibles $\mathcal{R}b_{ij}$ déja constituées (procédure CliqueOccupPrépaAssocConstOpt) ,

2. exploration des positions d'insertion correspondant à des associations possibles $\mathcal{R}b_{ij}$ non constituées (procédure CliqueOccupPrépaAssocNonConstOpt). ,

Dans les deux cas, ces procédures étant proches de celles définies jusqu'ici, on se contente d'en décrire les principes et d'illustrer leur fonctionnement par un exemple.

III.5.2 Enumération des positions d'insertion correspondant à des associations déja constituées

Le nombre maximum d'associations constituées pour (i,j) à un instant t est le nombre de ressources du pool principal de plus petit cardinal. Il s'agit du pool principal $Pb_{ij}^{mr^*}$ tel que : $|Pb_{ij}^{mr^*}| = \min\limits_{r=1,\ldots,Mb_{ij}^m} |Pb_{ij}^{mr}|$.

Pour chercher les associations déja constituées, on explore la séquence des opérations d'exécution déja ordonnancées sur chaque ressource $k \in Pb_{ij}^{mr^*}$, sous la forme d'une suite de 4-uplets $(k, 1, p(k,1), f(k,1))$ en testant si l'association utilisée par chaque opération $f(k,1)$ et $p(k,1)$ n'est pas une association possible pour (i,j) dans le mode étendu m. Pour un 4-uplet donné, $(k, 1, p(k,1), f(k,1))$, On peut envisager les 3 configurations i1, i2 et i3 du paragraphe III.4.2. Si aucune des configuration n'est vérifiée (pas d'association possible constituée) alord on passe au 4-uplet suivant (adjacent) sur la ressource k. Si au moins une configuration est vérifiée, on crée l'arc $\tilde{u}_{ij}$ correspondant et on calcule l'impact sur l'admissibilité.

La figure III.33 illustre le fonctionnement de la procédure d'énumération CliqueOccupPrépaAssocConstOpt ainsi définie. Dans cet exemple une opération (i,j) est à insérer sur deux ressources principales à choisir dans le pool Pb_{ij}^{m1} contenant 2 ressources et Pb_{ij}^{m2} contenant 3 ressources. On ne peut avoir à un instant donné que 2 associations déjà constituées pour (i,j). Les arcs $\tilde{u}_1(i,j)$ à $\tilde{u}_8(i,j)$ représentent les positions d'insertion correspondant à une association contenant la ressource $P1$. Au cours de l'exploration, on ignore le associations qui ne font pas partie des associations possibles pour (i,j) (ici, les zones où P2 est associé à M3). Dans une deuxième étape, on doit effectuer la même exploration pour la deuxième ressource du pool Pb_{ij}^{m1}.

III.5.3 Exploration des positions d'insertion nécessitant la réalisation d'une association

La première étape de la recherche d'une position d'insertion optimale a déterminé la position d'insertion optimale parmi celles insérant (i,j) dans une association déja constituée. On va maintenant rechercher la position d'insertion optimale parmi celles qui nécessitent la réalisation d'une association. On peut considérer que toute position d'insertion possible sur les ressources principales est dans la configuration ii décrite dans le paragraphe III.4.2, c'est-à-dire nécessite l'insertion du cas le plus général du graphe $\mathcal{G}_{ij}^*$.

Pour éviter l'énumération des occupations des ressources possibles, on applique l'idée introduite pour de prise en compte de l'affectation au sein d'un mode étendu sans préparation (paragraphe III.3). Dans ce dernier cas, on explore un ensemble de cliques maximales $\mathcal{C}^+$ correspondant à l'insertion d'une opération (i,s) de mêmes relations de précédence

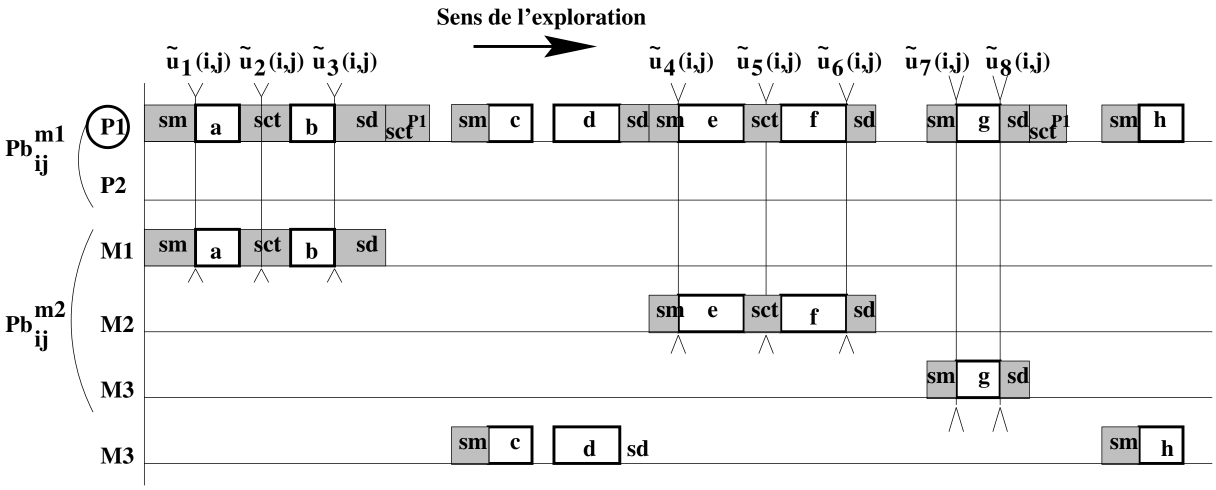


FIG. III.33 – Recherche d'une association déja constituée

que (i, j) et d'occupation des ressources fixée l_{ij+}, borne supérieure de l'occupation des ressources de (i, j). D'après le théorème 10, l'ensemble des cliques maximales explorées pour (i, s) inclut l'ensemble des cliques maximales $\mathcal{C}$ explorées par EXPLCLIQUESMAX-DOMIN pour (i, j) et chaque occupation des ressources possible l_{ij}. Dans le cas considéré ici, la procédure CLIQUEOCCUPPRÉPAASSOCNONCONSTOPT calcule la borne supérieure de l'occupation des ressources dans le mode m l_{ij+} et explore un ensemble de positions d'insertion $\mathcal{PI}^+(b)$ sur l'ensemble $\mathcal{R}b^+_{ij}$. Cette exploration suit le principe de la procédure CLIQUEPRÉPAOPT de la figure III.32 avec les particularités suivantes :

- la clique initiale $\tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{init}$ est remplacée par la position $\mathcal{PI}^+_{init}(b)$, ensemble des 4-uplets $(k, 1, O, f(k, 1))$, $\forall k \in \mathcal{R}b^+_{ij}$ tels que $f(k, 1)$ soit la première opération d'exécution sur la ressource k.

- les déplacements vers la droite s'effectuent autour de l'opération o_{dest} définie par la procédures CHOIXODESTMINΔR qu'on peut appliquer à partir de la clique $\tilde{\mathcal{C}}^{\mathcal{R}_{ij}}_{init}$ maximale courante ainsi définie. On obtient ainsi la position d'insertion initiale $\mathcal{PI}(b)^+_0$.

- A partir de $\mathcal{PI}(b)^+_0$, on effectue des déplacements vers la droite autour de l'opération o_{dest} définie par la procédures CHOIXODESTMIND jusqu'à la position d'insertion finale.

Pour chaque position d'insertion $\mathcal{PI}(b)^+$ explorée, on pourrait déterminer la position d'insertion optimale $\mathcal{PI}_{opt}(b) \subset \mathcal{PI}(b)^+$ en énumérant l'ensemble des sous-positions d'insertion possibles et en calculant l'impact de chacune d'elles sur l'admissibilité.

Pour éviter cette énumération, la procédure CLIQUEOCCUPPRÉPAASSOCNONCONSTOPT applique pour chacune de ces positions là procédure EXPLSOUSCLIQUESDOMINOCCUPPRÉPA pour la position initiale et EXPLSOUSCLIQUESPARTDOMINOCCUPPRÉPA qui renvoient la position d'insertion optimale incluse dans une position d'insertion $\mathcal{PI}(b)^+$.

On se ramène à un problème de recherche d'une sous-clique optimale proche de celui du paragraphe III.3 et résolu par les procédure ExplSousCliquesDominOccup et ExplSousCliquesDominPartOccup, au moyen des manipulations suivantes : Pour tout 4-uplet $(k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)^+$, on ajoute à la date de fin au plus tôt de $p(k,1)$ la somme de la durée de démontage de l'association $\mathcal{R}b_{p(k,1)}$ et de la durée de changement du type de la ressource k de $RT_{p(k,1)}$ vers RT_{ij}. On soustrait à la date de début au plus tard de $f(k,1)$ la somme de la durée de montage de l'association $\mathcal{R}b_{f(k,1)}$ et de la durée de changement du type de la ressource k de RT_{ij} vers $RT_{f(k,1)}$. En effet, ces durées ne dépendent que des associations ou ressources principales préparées, et pas l'affectation finale de (i,j). De plus si un 4-uplet est sélectionné pour l'insertion, ces opérations sont à réaliser systématiquement car on ne considère pas les positions d'insertion utilisant une association existante.

Tout 4-uplet de $\mathcal{PI}^+(b)$ peut alors être vu comme un arc de type ressource associé au dates définies ci-dessus et d'occupation des ressources réduite à la resource k. L'ensemble $\mathcal{PI}^+(b)$ peut être alors considéré comme une clique, dans laquelle on doit insérer successivement l'opération de montage de $\mathcal{R}b_{ij}$, (i,j) et l'opération de démontage de $\mathcal{R}b_{ij}$, opérations dont la durée dépend de l'affectation.

La figure III.34 illustre ces manipulations pour le cas déja présenté dans la figure III.33. On considère une position d'insertion
$\mathcal{PI}_b^+ = \{(P1,1,a,c),(M1,1,a,c),(P2,1,b,d),(M2,1,b,d),(M3,1,e,f)\}$. Les positions d'insertions valides possibles (ne correspondant pas à une association déja réalisée) sont $\{(P1,1,a,c),(M2,1,b,d)\}$, $\{(P1,1,a,c),(M3,1,e,f)\}$, $\{(P2,1,b,d),(M1,1,a,c)\}$, et $\{(P2,1,b,d),(M3,1,e,f)\}$. Les manipulations préliminaires génèrent cinq arcs de type ressource représentés sur la figure, formant une clique dans laquelle on doit insérer les opérations sm^*_{ij}, (i,j) et sd^*_{ij}.

Soit la clique $\mathcal{C}_{equiv}$ ainsi définie. Supposons que l'occupation des ressources de (i,j) soit fixée, ce qui fixe les durées $p_{sm^*_{ij}} = p_{sm}^{\mu(\mathcal{R}b_{ij})k1}$ où k est la ressource guidante de montage sélectionnée, $p_{sd^*_{ij}} = p_{sd}^{\delta(\mathcal{R}b_{ij})k'1}$ où k' est la ressource guidante de démontage sélectionnée, $p_{ij} = p_{ij}^{mk'''1}$ où k''' est la ressource guidante de (i,j) sélectionnée dans le mode étendu m. On peut appliquer la procédure ExplSousCliquesDomin à la clique $\mathcal{C}_{equiv}$ à une opération fictive de durée $p_{sm^*_{ij}} + p_{ij} + p_{sd^*_{ij}}$, de date de début au plus tôt $r_{ij} - p_{sm^*_{ij}}$, de date de fin au plus tard $d_{ij} + p_{sd^*_{ij}}$ et d'occupation des ressources l_{ij}. Il s'agit d'un cas particulier où la fenêtre temporelle de l'opération à insérer dépend de l'occupation des ressources de l'opération. On peut remarquer que l'arbre d'exploration des cliques dans ExplSousCliquesDomin ne dépend ni de la durée de (i,j) ni de sa fenêtre temporelle.

On peut résoudre le problème d'affectation à chaque noeud en calculant la position d'insertion $\mathcal{PI}_{min}(b)$ de plus petit impact sur l'admissibilité, c'est à dire qui minimise $\Delta d_{max}(i,j,\mathcal{PI}(b)) = \max(r_{ij}, r_{\mathcal{C}} + p_{sm^*_{ij}}) + p_{ij} - \min(d_{ij}, d_{\mathcal{C}} - p_{sd^*_{ij}})$. La procédure PosInsertMinΔd décrite dans la figure III.35 trouve cette position d'insertion en parcourant une seule fois chaque ressource de chaque pool principal. Cette procédure, utilisée dans les procédures ExplSousCliquesDominOccupPrépa et ExplSousCliquesPartDominOccupPrépa est à rapprocher de la procédure OccupDuréeMin utilisée pour résoudre le problème d'affectation sans préparation dans les procédures ExplSousCliquesDominOccup et ExplSousCliquesPartDominOccup.

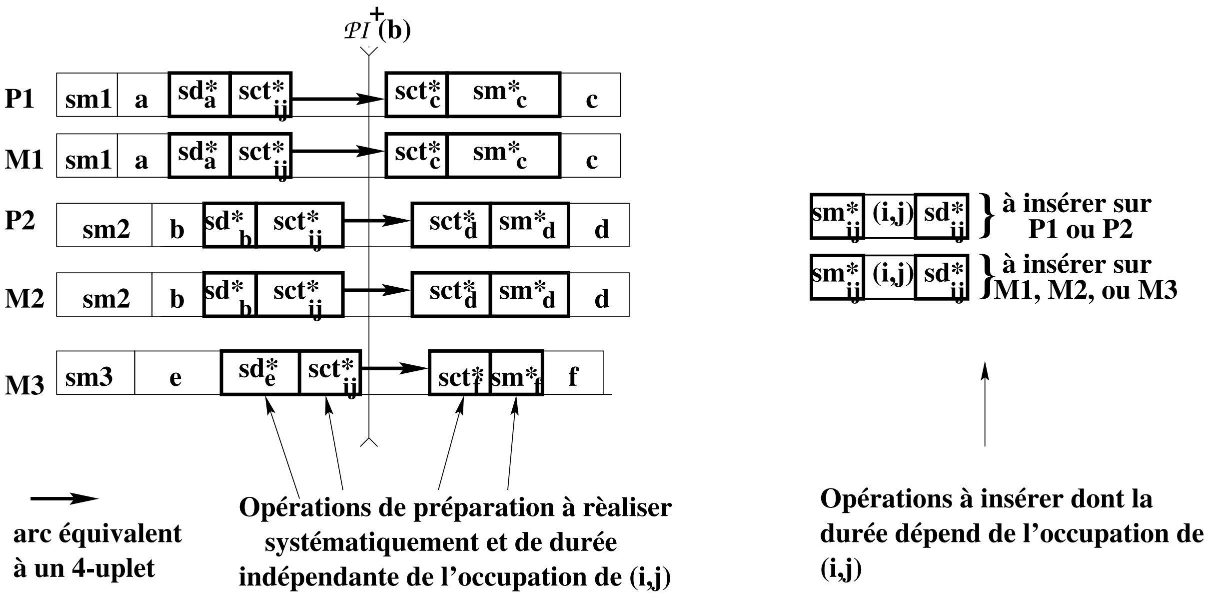


FIG. III.34 – Transformation des 4-uplets en arcs

III.6 Prise en compte des ressources complémentaires de préparation

Dans les paragraphes prédents, des procédures optimales de résolution du problème d'insertion défini à la section III.1 dans des cas particuliers du modèle général présenté en II.2. Dans ce paragraphe, on considère le problème le plus général. On suppose que les opérations de préparation et d'exécution nécessitent des ressources complémentaires. Pour chaque mode étendu de préparation, chaque pool de ressource complémentaires contient plusieurs ressources. Pour toute opération de montage de mode étendu μ, une ressource doit être sélectionnée dans chaque pool $|Pc_{sm}^{\mu r}| = 1, r = 1, ..., Mc_{sm}^{\mu}$ et un nombre de lignes de charge compris entre $e_{sm-}^{\mu k}$ et $e_{sm+}^{\mu k}$ doit être sélectionné sur chaque ressource complémentaire sélectionnée k. Pour toute opération de démontage de mode étendu δ, une ressource doit être sélectionnée dans chaque pool $|Pc_{sd}^{\delta r}| = 1, r = 1, ..., Mc_{sd}^{\delta}$ et un nombre de lignes de charge compris entre $e_{sd-}^{\delta k}$ et $e_{sd+}^{\delta k}$ doit être sélectionné sur chaque ressource complémentaire sélectionnée k. Pour toute opération de changement de type de mode étendu τ, une ressource doit être sélectionnée dans chaque pool $|Pc_{sct}^{\tau r}| = 1, r = 1, ..., Mc_{sct}^{\tau}$ et un nombre de lignes de charge compris entre $e_{sct-}^{\tau k}$ et $e_{sct+}^{\tau k}$ doit être sélectionné sur chaque ressource complémentaire sélectionnée k.

Pour une position d'insertion sur les ressources principales déterminée, les opérations de préparation à supprimer entre chaque paire $(p(k,1), f(k,1))$ sont celles définies qu'au paragraphe III.4.2. Les opérations de préparation à insérer entre chaque paire $(p(k,1), f(k,1)$ peuvent être représentées pour chaque configuration par le graphe $\mathcal{G}_{ij}^{*}$ décrit dans le paragraphe III.4.2, à la différence que pour chacune de ces opérations, une position d'insertion doit être également déterminée sur chaque ressource complémentaire. La date de début au plus tôt r_{ij}^{*b} n'est plus qu'une borne inférieure de la date obtenue

Proc PosInsertMinΔd $(\mathcal{PI}(b), i, j, m, \mathcal{PI}_{min}(b), \Delta d_{max})$

$\Delta d_{max} \leftarrow 0$

$\mathcal{PI}_{min}(b) \leftarrow \emptyset$

Pour $r = 1, ..., Mb_{ij}^{m}$ **faire**

$\Delta d_{pool} \leftarrow \infty$

Pour $k \in Pb_{ij}^{mr}$ **faire**

Si $\exists (k, 1, p(k,1), f(k,1)) \in \mathcal{PI}(b)$ **alors**

$\Delta d(k) \leftarrow p_{ij}^{mk1}$

Si $p_{sm}^{\mu(\mathcal{R}b_{ij})k1} \neq 0$ **alors**

$\Delta d(k) \leftarrow \Delta d(k) + p_{sm}^{\mu(\mathcal{R}b_{ij})k1}$

FinSi

Si $p_{sd}^{\delta(\mathcal{R}b_{ij})k1} \neq 0$ **alors**

$\Delta d(k) \leftarrow \Delta d(k) + p_{sd}^{\delta(\mathcal{R}b_{ij})k1}$

Si $\Delta d(k) < \Delta d_{pool}$ **alors**

$\Delta d_{pool} \leftarrow \Delta d(k)$

$\forall k' \in Pb_{ij}^{mr}$, $\mathcal{PI}_{min}(b) \leftarrow \mathcal{PI}_{min}(b) \setminus \{(k', 1, p(k',1), f(k',1)\}$

$\mathcal{PI}_{min}(b) \leftarrow \mathcal{PI}_{min}(b) \cup \{(k, 1, p(k,1), f(k,1))\}$

FinSi

FinSi

FinPour

$\Delta d_{max} \leftarrow \Delta d_{max} + \Delta d_{pool}$

FinPour

Pour $r \in 1...Mb_{i,j}^{m}$ **faire**

(*test que la position d'insertion est valide*)

Si $\forall k \in Pb_{ij}^{mr}$, $\forall (k', 1, p(k',1), f(k',1)) \in \mathcal{PI}_{min}(b)$, $k \neq k'$ **alors**

$\Delta d_{max} \leftarrow \infty$; $\mathcal{PI}_{min}(b) \leftarrow \emptyset$

FinSi

FinPour

FinPRoc

FIG. III.35 – Algorithme de PosInsertMinΔd

après insertion des opérations de préparation précédant (i, j) dans $\mathcal{G}_{ij}^{*}$. La date de fin au plus tard d_{ij}^{*b} n'est plus qu'une borne supérieure de la date obtenue après insertion des opérations de préparation suivant (i, j) dans $\mathcal{G}_{ij}^{*}$. Le problème est plus complexe que ceux considérés jusqu'à présent car des conflits peuvent apparaître entre opérations de préparation utilisant la même ressource complémentaire. C'est le cas lorsque le démontage et le changement de type de plusieurs ressources principales sont nécessaires avant l'opération (i, j) puisque chaque ressource peut être préparée parallèlement aux autres. C'est le cas aussi après l'opération (i, j) lorsque le changement de type et le montage de plusieurs ressources principales sont nécessaires. Face à la complexité de ce problème, une heuristique est proposée. Elle consiste à sélectionner dans un premier temps un ensemble de positions d'insertion valides pour l'opération d'exécution. Pour chacune de ces position d'insertion, le graphe $\mathcal{G}_{ij}^{*}$ est généré et ses opérations (incluant (i, j) sont insérées une à une sur leurs ressources complémentaires. Le paragraphe III.6.1 décrit l'heuristique utilisée pour insérer ce graphe à une position donnée sur les ressources principales. Le paragraphe III.6.2 décrit comment les positions d'insertion sur les ressources principales sont sélectionnées.

III.6.1 Insertion à une position donnée sur les ressources principales

L'heuristique HEURINSÈREPRÉPA décrite dans la figure III.36 est définie pour insérer un graphe $\mathcal{G}_{ij}^*$ correspondant à une position d'insertion donnée sur les ressources principales. L'idée est de considérer l'insertion de chaque opération du graphe sur ses ressources complémentaires comme un problème d'insertion d'une opération dans un ordonnancement cumulatif multi-resources sans préparation (cas traités en III.2 et III.3). Les opérations du graphe deviennent candidates à l'insertion dès que les opérations qui les précèdent ont été insérées, comme pour une méthode sérielle classique de génération d'un ordonnancement. Une fenêtre temporelle est associée à chaque opération candidate en fonction des opérations déja insérées, de la fenêtre temporelle associée à (i, j) et des dates associées aux 4-uplets de la position d'insertion. Le détail de ce calcul est donné dans la figure III.36. Cette fenêtre assure la compatibilité de la clique trouvée par la procédure CLIQUEOCCUPOPT appliquée à l'opération candidate avec la position d'insertion sur les ressources principales. L'opération candidate sélectionnée pour l'insertion est celle qui est jugée la plus urgente. L'urgence est mesurée par l'impact sur l'admissibilité de l'insertion de l'opération, calculée par CLIQUEOCCUPOPT.

III.6.2 Exploration des positions d'insertion sur les ressources principales

Pour sélectionner un ensemble de positions d'insertion valides pour l'opération d'exécution, on ignore dans un premier temps les ressources complémentaires d'exécution et de préparation. Si on doit traiter un problème d'affectation sur les ressources principales au sein de chaque mode étendu, on applique les procédures CLIQUEOCCUPPRÉPAASSOCCONSTOPT et CLIQUEOCCUPPRÉPAASSOCNONCONSTOPT. S'il n'y a pas de problème d'affectation au sein de chaque mode étendu (une seule ressource dans chaque pool principal) on applique la procédure CLIQUEPRÉPAOPT.

Deux utilisations de HEURINSÈREPRÉPA sont possibles. On peut se contenter d'appliquer HEURINSÈREPRÉPA une seule fois à partir de la position d'insertion optimale sans ressources complémentaires calculée par la procédure d'exploration retenue. Pour obtenir un meilleur résultat, on peut appliquer HEURINSÈREPRÉPA à chaque position d'insertion dominante explorée. Dans ce dernier cas, après avoir calculé l'impact effectif de l'insertion de $\mathcal{G}_{ij}^*$ sur l'admissibilité (nouveau r_H), on doit annuler l'insertion de (i, j) dans la position d'insertion considérée. Pour cela, on retire du graphe les opérations de $\mathcal{G}_{ij}^*$ et on restaure les opérations de préparation nécessaires. On peut ensuite passer à la position d'insertion dominante sans ressources complémentaire suivante.

III.7 Conclusion

Dans ce chapitre, nous avons proposé plusieurs méthodes de résolution de problèmes d'insertion d'une opération dans un ordonnancement avec pour objectif la minimisation

ProcHeurInsèrePrépa $(\mathcal{G}, \mathcal{PI}(b), \mathcal{G}^*_{ij}, r_{ij}, d_{ij}, \Delta d_{max}(i,j,\mathcal{PI}(b)))$

Etape 0 Supprimer les opérations de préparation entre chaque paire $(p(k,1), f(k,1))$ de $\mathcal{PI}(b)$. Mettre à jour les r et les d du graphe $\mathcal{G}'$ ainsi obtenu.

Etape 1 Calculer les dates de début au plus tôt et les dates de fin au plus tard de chaque opération $o \in \mathcal{G}^*_{ij}$ sans tenir compte de ressources complémentaires : en propageant $r_{p(k,1)}+p_{p(k,1)}$ et r_{ij} pour calculer les r_o et en propageant $d_{f(k,1)} - p_{f(k,1)}$ et d_{ij} pour calculer les d_o. $\Delta d_{max}(i,j,\mathcal{PI}(b)) \leftarrow 0$.

Etape 2 Initialiser la liste des opérations candidates à l'insertion : $\mathcal{L} = \{o \in \mathcal{G}^*_{ij} | pred_o^{\mathcal{G}^*_{ij}} = \emptyset\}$.

Etape 3 **Si** $\mathcal{L}$ est vide, alors **Fin**, **sinon** aller à l'étape 4.

Etape 4 **Pour** chaque opération $o \in \mathcal{L}$ **Faire**

Pour Chaque mode étendu m possible pour o **faire**
CliqueOccupOpt $(o, M_o^m, \mathcal{P}_o^m, lc_{o+}, lc_{o-}, r_o, d_o, \mathcal{G}, \mathcal{C}_o, \Delta d^o_{max}, l_o)$
conserver le plus petit Δd^o_{max}, et les l_o et $\mathcal{C}_o$ correspondant
Finpour

Finpour

Etape 5 Sélectionner l'opération $o^{min} \in \mathcal{L}$ telle que o^{min} soit l'opération candidate de plus petite fin au plus tôt $r^*_{o^{min}} + p_{o^{min}}$ dans la clique $\mathcal{C}_{o^{min}}$ trouvée.

Etape 6 Générer l'ensemble $\mathcal{E}$ des opérations de $\mathcal{L}$ en conflit avec o^{min} sur une ressource complémentaire : $\mathcal{E}$ est l'ensemble des opérations $o \in \mathcal{L}$ telles que $r^*_o < r^*_{o^{min}} + p_{o^{min}}$ et $\mathcal{C}_o \cap \mathcal{C}^{o^{min}} \neq \emptyset$.

Etape 7 Sélectionner dans $\mathcal{E}$ l'opération o^u jugée la plus urgente, c'est à dire telle que $\Delta d^{o^u}_{max} = \max_{o \in \mathcal{E}} \Delta d^o_{max}$.

Etape 8 InsèreClique $(o^u, l_{o^u}, \mathcal{C}_{o^u}, \mathcal{G}, \mathcal{R}, l^k_{o^u})$

Etape 9 Propager la date de fin au plus tôt de o^* pour mettre à jour les autres dates. Supprimer o^* de la liste $\mathcal{L}$. Ajouter à $\mathcal{L}$ les opérations de $\mathcal{G}^*_{ij}$ dont toutes les précédentes ont été insérées. Aller à l'étape 3.

Fig. III.36 – Algorithme de HeurInsèrePrépa

de l'impact de l'insertion sur l'admissibilité.

Un certain nombre de cas particuliers de ce problème sont résolus de façon optimale sans toutefois énumérer l'ensemble des positions d'insertion possibles.

Le problème d'insertion d'une opération dans un ordonnancement multi-ressource cumulatif sans préparation et dans le cas où l'occupation des ressources de l'opération est fixée a d'abord été considéré. Ce problème a été résolu en transformant le problème de recherche d'une position d'insertion en un problème de recherche d'une clique dans un graphe dont les sommets sont les arcs de type ressource du graphe $\mathcal{G}$ représentant l'ordonnancement. L'étude de relations de dominances entre cliques concernant l'impact sur l'admissibilité a permis de développer un algorithme de résolution polynomial de ce problème par exploration des cliques dominantes directement dans le graphe $\mathcal{G}$. Des conditions suffisantes garantissent l'impossibilité de générer un cycle par l'insertion dans les positions d'insertion explorées. Cet algorithme polynomial (CliqueOpt) trouve une position d'insertion optimale ce qui constitue une amélioration par rapport à l'algorithme de résolution exponentiel du problème de recherche des cliques développé lors d'études antérieures.

L'extension du problème à la prise en compte de la sélection d'une ressource dans chaque pool et d'un nombre de lignes de charge variable sur une ressource cumulative a été ensuite considérée. Ces possibilités donnent lieu à un ensemble d'occupation des ressources possible pour une opération dans un mode étendu donné. La structure de l'arbre d'exploration des cliques dominantes identique pour toutes les occupations permet de résoudre le problème d'affectation localement à un noeud de l'exploration. Un algorithme exact et polynomial (CLIQUEOCCUPOPT) en est déduit.

L'extension du problème à la prise en compte des opérations de préparation sans ressource complémentaire de préparation et à occupation fixée pour l'opération d'exécution a ensuite été étudiée. Sous l'hypothèse de relations restrictives mais réalistes sur les temps de préparation, des règles de dominances permettant de réduire l'énumération des positions d'insertion sont encore établies. L'application de ces règles pour l'exploration de positions d'insertion ne constitue plus une condition suffisante pour garantir la non création de cycle, ce qui entraine l'utilisation de procédures de recherche de chemin dans le graphe $\mathcal{G}$ Malgré cet inconvénient, un algorithme polynomial CLIQUEPREPAOPT) permet encore de trouver une position d'insertion optimale.

L'étude des cas particuliers s'est terminée sur la prise en compte des possibilités d'affectation dans le contexte décrit ci-dessus mais sans ressource complémentaire d'exécution. Dans ce cas, une énumération préalable des positions d'insertion utilisant une association de ressources principales existante est effectuée (algorithme polynomial CLIQUEOCCUPPRÉPAASSOCCONSTOPT). On considère ensuite que l'insertion génère systématiquement la constitution de l'association requise (montage, démontage). Dans ce cas, l'énumération de l'ensemble des occupations possibles est encore évitée par l'algorithme polynomial CLIQUEOCCUPPRÉPAASSOCNONCONSTOPT. L'utilisation successive des deux procédures trouve une position d'insertion optimale.

La figure III.37 résume les 4 cas particuliers de résolution optimale étudiés. Pour chacun des cas, on distingue la partie d'exploration des cliques maximales domainantes qui résout le problème d'ordonnancement sans résoudre le problème d'affectation, de la partie d'exploration des sous-cliques dominantes qui résout progressivement le problème d'affectation.

Dans le cas général où les opérations d'exécution et de préparation ont des ressources complémentaires, l'insertion du graphe correspondant à une position d'insertion sur les ressources principales est plus difficile, car on doit trouver une position d'insertion sur les ressources compléntaires pour toutes les opérations du graphe et résoudre d'éventuels conflits entre certaines opérations de préparation pouvant s'exécuter en parallèle. Une heuristique HEURINSÈREPRÉPA est proposée prour trouver une solution approchée au problème d'insertion. Le principe est de résoudre succesivement des problèmes d'insertion d'une unique opération dans un contexte sans préparation, en affectant une fenêtre temporelle à chaque opération du graphe.

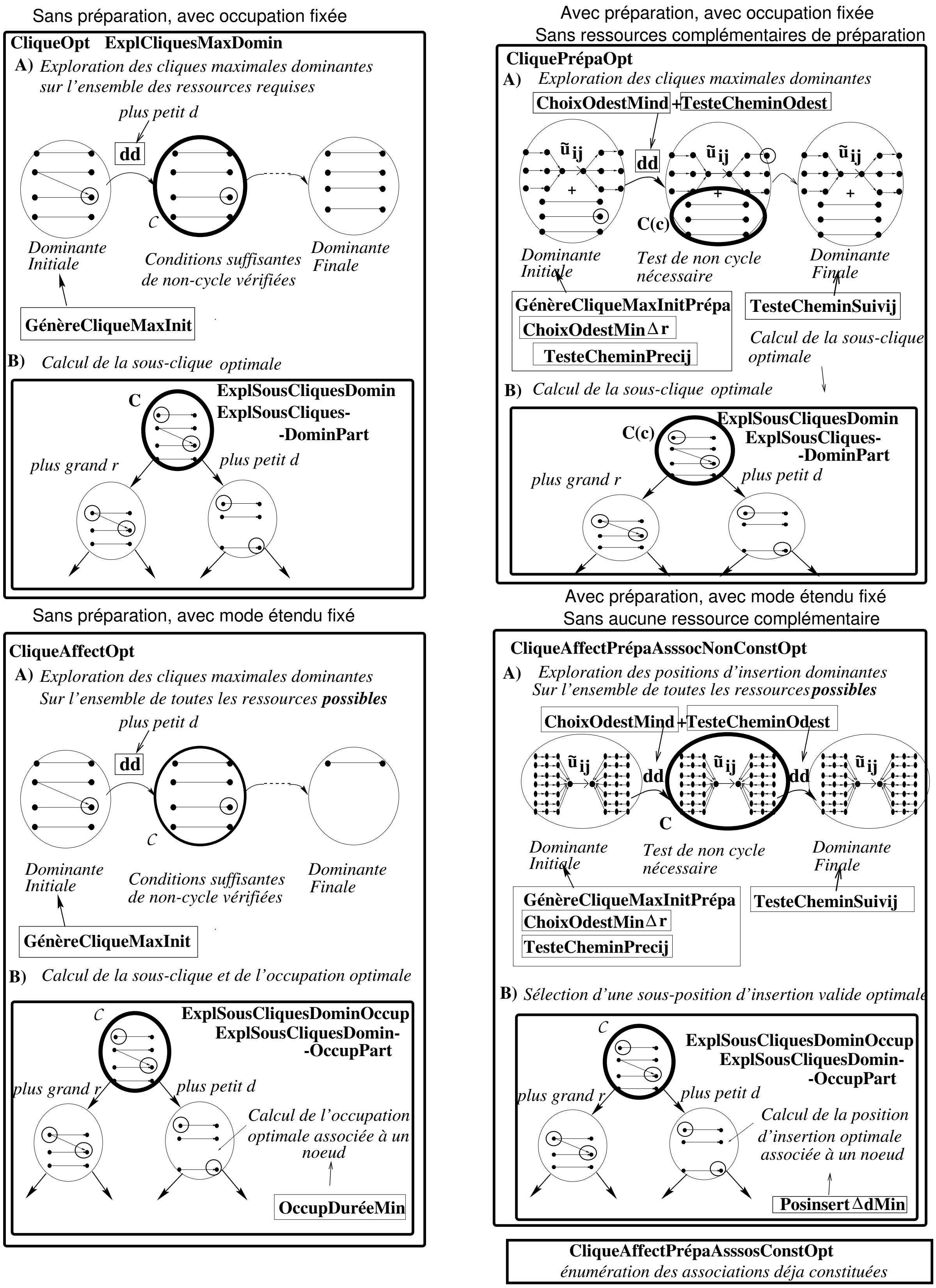


Fig. III.37 – Résumé des procédures optimales

Chapitre IV

Méthodes d'amélioration d'un ordonnancement basées sur l'insertion

Résumé

La validation expérimentale de la procédure d'insertion du chapitre précédent est effectuée par l'intermédiaire de son intégration dans une méthode de voisinage pour l'amélioration d'un ordonnancement. Une méthode de type tabou est présentée dans un contexte sans préparation. Cette méthode est testée sur des problèmes connus d'ordonnancement de projet à mode simple et à modes multiples. La méthode tabou est ensuite étendue au contexte général avec des opérations de préparation. Des résultats expérimentaux sont présentés sur des problèmes générés aléatoirement.

IV.1 Amélioration d'un ordonnancement cumulatif multi-ressources sans opération de préparation

IV.1.1 Présentation générale

On considère un problème d'ordonnancement possédant les caractéristiques décrites dans le paragraphe II.2, avec la restriction suivante : aucune opération d'exécution ne nécessite de préparation. Soit $\mathcal{G}$ le graphe potentiels-tâches représentant un ordonnancement unique, solution du problème considéré. Toute opération de $\mathcal{G}$ est, dans ce contexte, une opération d'exécution. Si $\mathcal{G}$ est tel que $T_{max} > 0$, alors l'ordonnancement associé n'est pas admissible. Dans ces conditions, la recherche d'un ordonnancement admissible du graphe $\mathcal{G}$ peut être réalisée à travers la minimisation du plus grand retard T_{max}.

Les dates de fin au plus tard des opérations du graphe $\mathcal{G}$ ont été calculées pour respecter les dates de livraison de tous les ordres de fabrication en avance et les dates de fin au plus tôt de tous les ordres de fabrication en retard (voir paragraphe II.3.3.b)). Dans l'optique de la minimisation du plus grand retard, nous modifions dans ce chapitre la définition de ces dates. Le poids des arcs de H vers $(i, h_i + 1)$ est maintenant $d_i + T_{max}$, quel que soit l'ordre de fabrication. Dans ces conditions, la violation d'une unité de la date de fin au plus tard d'une opération, augmente le plus grand retard des ordres de fabrication d'une unité. On peut ainsi remplacer l'impact sur l'admissibilité Δd_{max} par l'impact sur le plus grand retard ΔT_{max}.

Une méthode de voisinage est proposée pour résoudre ce problème. Une méthode de voisinage dans un problème d'ordonnancement part d'un ordonnancement initial et applique itérativement une méthode de génération d'un ordonnancement voisin dans le but d'améliorer un critère donné. L'élément principal d'une méthode de voisinage est la méthode utilisée pour générer une solution voisine qui définit une structure de voisinage.

De nombreuses méthodes de voisinage (recuit simulé [Van Laarhoven *et al.*, 1992], méthode tabou [Hertz et Widmer, 1995], algorithmes génétiques [Portmann, 1996]) ont été définies pour les problèmes d'ordonnancement d'atelier à cheminement unique et à cheminements multiples classiques (voir [Vaessens et al. 96] pour un état de l'art de ces méthodes). Pour l'atelier à cheminements multiples, la structure de voisinage la plus utilisée est la permutation d'une paire d'opérations reliées par un arc critique (*pairwise interchange*, voir [Van Laarhoven *et al.*, 1992]). Dans [Dauzère Péres et al, 1996], cette structure est étendue à un problème d'ordonnancement d'atelier proche de celui considéré ici, contenant à la fois des gammes non-linéaires, des problèmes d'affectation et des ressources multiples nécessaires simultanément pour réaliser une même opération. Une solution voisine est obtenue soit en inversant tous les arcs critiques entre deux opérations, soit en changeant l'affectation de l'opération sur une seule de ses ressources.

Pour les problèmes cumulatifs comme l'ordonnancement de projet à moyens limités (RCPSP), toutes les méthodes de voisinage proposées sont basées à notre connaissance sur le classement des opérations dans une liste de priorité (voir par exemple [Pinson et al., 1994], [Boctor, 1996], [Lee et Kim, 1996]). La génération d'une solution voisine consiste en la modification de ce classement. Un unique ordonnancement réalisable est généré à partir d'une liste de priorité donnée par la méthode sérielle [Kelley, 1963]. Très peu de méthodes sont basées à notre connaissance sur l'exploration directe d'un graphe représentant l'ordonnancement. L'application de la méthode de voisinage développée dans [Dauzère Péres et al, 1996], basée sur l'exploration du graphe, donne d'après les auteurs des résultats décevants sur les problèmes de RCPSP.

Dans le paragraphe IV.1.2, une méthode de génération d'un ordonnancement voisin basée sur la procédure d'insertion qui vient d'être définie (et par conséquent sur l'exploration du graphe potentiels-tâches), est décrite. Une méthode élémentaire de voisinage basée sur l'exploration du voisinage ainsi défini en vue de la recherche d'un ordonnancement optimal est présentée dans le paragraphe IV.1.3. Des résultats expérimentaux sur des problèmes cumulatifs sont présentés dans le paragraphe IV.1.4.

IV.1.2 Méthode de génération d'un ordonnancement voisin

A partir d'un ordonnancement représenté par un graphe $\mathcal{G}$, un ordonnancement voisin représenté par un graphe $\mathcal{G}^*$ est obtenu par le transfert d'une opération. On définit le transfert d'une opération (i,j) de $\mathcal{G}$ en trois étapes :

Etape 1 Suppression dans le graphe $\mathcal{G}$ des arcs de type ressource entrant dans (i,j) et sortant de (i,j). Soit $\mathcal{PI}$ la position de (i,j) avant la suppression. Entre chaque opération $p(k,l)$ et chaque opération $f(k,l)$ telles que $(k,l,p(k,l),f(k,l)) \in \mathcal{PI}$, un arc de type ressource $u_{p(k,l)f(k,l)}$ est créé, s'il n'existait pas déjà. Si cet arc existait déjà, son occupation des ressources et son occupation des lignes de charge sont mises à jour (augmentées des lignes de charge communes entre $p(k,l)$, (i,j) et $f(k,l)$). Les arcs de type gamme entrant et sortant de (i,j) sont maintenus. Les arcs de type gamme sortant de (i,j) prennent pour poids la plus petite durée possible de (i,j) dans tous les modes étendus, de façon à donner à (i,j) le moins d'impact possible. Soit $\mathcal{G}'$ le graphe ainsi obtenu. Les dates de début au plus tôt des opérations $f(k,l)$ sont mises à jour et par propagation dans $\mathcal{G}'$, le plus grand retard T'_{max} associé au graphe $\mathcal{G}'$ et issu de la suppression des arcs de type ressource liés à (i,j) est ainsi mis à jour. Les nouvelles dates de fin au plus tard des opérations sont mises à jour par propagation inverse dans $\mathcal{G}'$. Parmi ces dates, on considère la date de début au plus tôt r'_{ij} et la date de fin au plus tard d'_{ij} calculées uniquement à partir des contraintes de gamme (équations de III.2 à III.4 dans le paragraphe III.1).

Etape 2 Le graphe $\mathcal{G}'$ et l'opération (i,j) sont alors dans les conditions d'application de la procédure de recherche d'une position d'insertion optimale décrite dans le paragraphe III.3. Cette étape est ainsi constituée par la sélection d'une des occupations des ressources l^*_{ij} et d'une des cliques suffisantes $\mathcal{C}$ explorées par CliqueAffectOpt pour chaque mode étendu possible dans le graphe $\mathcal{G}'$.

Etape 3 Insertion de (i,j) dans la clique $\mathcal{C}$ avec l'occupation l^*_{ij} par la procédure InsèreClique décrite au paragraphe III.8.

Pour que le graphe $\mathcal{G}^*$ ainsi obtenu corresponde bien à un graphe voisin, on doit s'assurer soit que $l^*_{ij} \neq l'_{ij}$ (changement d'affectation) soit que la position d'insertion $\mathcal{PI}^*$ sélectionnée par InsèreClique à partir de la clique $\mathcal{C}$ sélectionnée dans $\mathcal{G}'$ est telle qu'il existe une opération précédente $p^*(k,l)$ telle que $p^*(k,l) \neq p(k,l)$, pour au moins un 4-uplet $(k,l,p(k,l),f(k,l)) \in \mathcal{PI}$ ou bien que $\mathcal{C}$ est telle qu'il existe une opération suivante $f^*(k,l)$ telle que $f^*(k,l) \neq f(k,l)$, pour au moins un 4-uplet $(k,l,p(k,l),f(k,l)) \in \mathcal{PI}$. (changement dans la séquence sans changement d'affectation). Soit T_{max} le plus grand retard associé à $\mathcal{G}$. Soit T'_{max} le plus grand retard associé à $\mathcal{G}'$. Soit $\Delta T_{max}(i,j,\mathcal{PI}(\mathcal{C}),p_{ij}(l^*_{ij}))$ l'augmentation du plus grand retard causée par l'insertion de (i,j) dans $\mathcal{C}$.

La variation du plus grand retard correspondant au transfert vaut
$T'_{max} + \Delta T_{max}(i,j,\mathcal{PI}(\mathcal{C}),p_{ij}(l^*_{ij})) - T_{max}$. Si la position d'insertion sélectionnée est telle que ($\Delta T_{max}(i,j,\mathcal{PI}(\mathcal{C}),p_{ij}(l^*_{ij})) < T_{max} - T'_{max}$), une diminution du plus grand retard est alors obtenue.

L'impact du transfert peut ainsi être connu précisément sans effectuer l'étape 3 d'insertion effective de l'opération. Lorsqu'on doit tester le transfert de plusieurs opérations avant d'en sélectionner un, cette caractéristique est intéressante. C'est le cas lorsqu'on explore le voisinage défini par l'ensemble des transferts possibles d'une opération (i, j) dans un ordonnancement représenté par un graphe $\mathcal{G}$.

IV.1.3 Méthode de voisinage de type tabou

Pour trouver un ordonnancement de plus grand retard minimal, une méthode de voisinage de type tabou basée sur l'application itérative de la procédure de transfert peut être définie. Le particularité des méthodes d'optimisation globale de type tabou est la possibilité d'accepter localement une augmentation du critère considéré lorsqu'aucune des solutions voisines explorées ne mène à une diminution du critère considéré, afin de s'affranchir d'un éventuel optimum local. Dans notre problème, ceci revient à accepter des transferts tels que $T'_{max} + \Delta T_{max}(i, j, \mathcal{PI}(\mathcal{C}), p_{ij}(l'_{ij})) > T_{max}$, c'est-à-dire des transferts qui augmentent le plus grand retard. Nous décrivons dans le paragraphe IV.1.3.a) la procédure d'exploration du voisinage, puis dans le paragraphe IV.1.3.b) la façon dont on gère une liste tabou, avant de présenter l'algorithme élémentaire de recherche tabou utilisé dans le paragraphe IV.1.3.c).

IV.1.3.a) Méthode d'exploration du voisinage

L'objectif étant de réduire le plus grand retard, on sélectionne l'opération à transférer parmi les opérations situées sur le chemin critique du graphe $\mathcal{G}$, ce qui réduit d'emblée la taille du voisinage. En effet, le plus grand retard ne pourra décroître après le transfert que si l'opération appartient à tous les chemins critiques du graphe $\mathcal{G}$.

Pour explorer la totalité du voisinage ainsi défini dans le but d'effectuer le meilleur transfert, on peut appliquer la procédure ExploreVoisinage décrite dans la figure IV.1.

Proc ExploreVoisinage
Pour chaque opération critique (i, j) du graphe $\mathcal{G}$ de plus grand retard $T_{max} > 0$, itérer les étapes A, B et C suivantes :

- **Etape A** Appliquer l'étape 1 du transfert à (i, j) : obtention d'un graphe $\mathcal{G}'$, du plus grand retard $T'_{max}(i, j) \leq T_{max}$ et d'une fenêtre temporelle r'_{ij} et d'_{ij} par la suppression des arcs de type ressource entrant et sortant de (i, j), la création et la mise à jour d'autres arcs de type ressource.

  Etape B Appliquer l'étape 2 du transfert à (i, j) : Pour chaque mode étendu possible pour (i, j), explorer les cliques suffisantes du graphe $\mathcal{G}'$ et mémoriser la meilleure clique $\mathcal{C}$ et la meilleure occupation l^*_{ij} donnant l'augmentation du plus grand retard $\Delta T_{max}(i, j, \mathcal{PI}(\mathcal{C}(i, j)), p_{ij}(l^*_{ij}))$ minimale (utilisation de la procédure CliqueAffectOpt).

  Etape C Restaurer le graphe $\mathcal{G}$

Sélectionner enfin l'opération critique à transférer de plus petit
$T'_{max}(i, j) + \Delta T_{max}(i, j, \mathcal{PI}(\mathcal{C}(i, j)), p_{ij}(l^*_{ij}) - T_{max}$.

Fig. IV.1 – *Algorithme de* ExploreVoisinage

Des tests préliminaires ont montré que les modifications de $\mathcal{G}$ nécessaires à l'étape A

peuvent pénaliser l'algorithme en terme de temps de calcul pour des problèmes de taille importante. C'est pourquoi plusieurs stratégies ont été mises en oeuvre pour augmenter la rapidité de la sélection d'un ordonnancement voisin.

Une première stratégie consiste en la réduction du temps consacré à un pas élémentaire. Pour cela l'omission de l'étape A peut être envisagée. La procédure d'exploration des cliques s'effectue directement dans le graphe $\mathcal{G}$ qui contient l'opération (i,j). Tout se passe comme si on ajoutait à $\mathcal{G}$ une nouvelle opération identique à (i,j). L'augmentation du plus grand retard obtenue par cette méthode est une borne supérieure de l'impact réel du transfert de (i,j).

Une deuxième stratégie consiste à restreindre arbitrairement l'exploration du voisinage. Pour cela, on peut considérer uniquement un sous ensemble des opérations critiques et effectuer la totalité de la procédure ExploreVoisinage sur ce sous-ensemble. On peut aussi réduire le temps d'exploration des cliques en arrêtant la procédure d'exploration des cliques dominantes avant son terme ou en réduisant le nombre de sous-cliques explorées.

IV.1.3.b) Gestion de la liste tabou

Parmi toutes les cliques explorées, la méthode décrite au paragraphe III.3 permet de sélectionner la clique correspondant à l'augmentation du plus grand retard minimale pour chaque opération critique (Etape B de ExploreVoisinage). Pour éviter de revenir par un transfert à un ordonnancement déjà visité, on modifie cette méthode d'exploration. Pour chaque transfert effectué, on stocke dans une liste tabou L un ensemble d'actions interdites de la façon suivante (structure de liste inspirée de [Dauzère Péres et al, 1996]) : lorsqu'on insère (i,j) entre $p(k,l)$ et $f(k,l)$ on stocke dans L un ensemble de triplets $(k, p(k,l), (i,j))$ pour chaque ressource k sur laquelle $p(k,l)$ et (i,j) se retrouvent consécutives sauf si $p(k,l)$ et (i,j) étaient déjà consécutives sur un sous-ensemble des lignes de charge de k avant l'insertion.

On modifie alors la procédure d'exploration des cliques maximales de la façon suivante : avant d'explorer les sous-cliques d'une clique maximale, on ôte de cette clique les arcs de type ressource u_α tels qu'il existe un triplet $(k,p,s) \in L$ vérifiant soit $orig_\alpha = p$, $(i,j) = s$ et $e_\alpha^k > 0$ soit $(i,j) = p$, $dest_\alpha = s$, et $e_\alpha^k > 0$. On empêche ainsi lors du transfert de créer une configuration où p et s sont à nouveau consécutives sur la ressource k. Un tel transfert est interdit pendant un nombre d'itérations spécifié par la taille de la liste.

IV.1.3.c) Algorithme de la procédure RechercheTabou

Un ordonnancement initial est généré par un algorithme de liste utilisant la règle de priorité SLACK/RPT voir [Billaut, 1993] et le paragraphe V.2. Une fois cet ordonnancement initial obtenu, on applique l'algorithme RechercheTabou décrit dans la figure IV.2.

IV.1.4 Résultats expérimentaux : application à l'ordonnancement de projet à moyens limités

Il n'a pas été possible de trouver dans la littérature des problèmes d'atelier du type considéré ici. Aussi, la procédure a été testée sur deux séries de problèmes tests d'or-

Proc RECHERCHETABOU
Faire $nbiter \leftarrow 0$ et $L \leftarrow \emptyset$

Tantque $T_{max} > 0$ et $nbiter < nbmaxtiter$ et qu'un transfert est possible, **faire**:

- Explorer le voisinage (appliquer EXPLOREVOISINAGE) et sélectionner l'opération (i, j), l'occupation l^*_{ij} et la clique $\mathcal{C}$ non tabou.
- Si un transfert a été trouvé, supprimer (i, j) de $\mathcal{G}$ puis insérer (i, j) avec INSÈRECLIQUE et mettre à jour la liste tabou L.
- Mettre à jour les r_o et les d_o, $\forall o \in \mathcal{O}$ et calculer le nouveau T_{max}.
- Si le T_{max} obtenu est le meilleur trouvé jusqu'à présent, faire $nbiter \leftarrow 0$ et mémoriser $\mathcal{G}$, sinon incrémenter $nbiter$.

FinTantque

FinProc

FIG. IV.2 – *Algorithme de* RECHERCHETABOU

donnancement de projet à moyens limités dans l'objectif de minimiser la durée totale de l'ordonnancement d'un projet unique que l'on note ici C. Pour appliquer notre procédure qui vise à minimiser le plus grand retard à un problème de minimisation de la durée totale, il suffit de donner au projet une date de livraison qui soit une borne inférieure de la durée minimale. Pour cela, on utilise la méthode du chemin critique en omettant les contraintes de type ressource. Dans les deux cas, le nombre maximal d'itérations est fixé à 1000 et la taille de la liste tabou est fixée à 20. Les temps de calcul sont donnés sur une station SUN de type Ultra-2.

IV.1.4.a) Ordonnancement de projet à moyens limités et mode unique

La première série de problèmes est tirée de [Alvarez-Valdes et Tamarit, 1989]. Ces 48 problèmes ont été sélectionnés car ils sont parmi les problèmes d'ordonnancement de projet de plus grande taille trouvés dans la littérature puisqu'ils comportent 103 opérations. Chaque opération possède un mode unique, ce qui signifie que son occupation des ressources et sa durée sont fixées. On se réfère aux résultats optimaux ou aux meilleures solutions connues donnés dans [Beldiceanu *et al.*, 1996]. Ces résultats ont été présentés dans [Artigues et Roubellat, 1997b] et sont résumés dans la figure IV.3.

nb sol. à améliorer	nb sol améliorées	C_{tabou}	ΔC	temps CPU
42	39	+3.89%	-3,91%	514 (s)

FIG. IV.3 – *Résultats sur les problèmes de [Alvarez-Valdes et Tamarit, 1989] à 103 opérations*

La procédure parvient à améliorer la durée totale de 39 problèmes (2ème colonne) sur 42 problèmes pour lesquels la procédure de génération de la solution initiale n'a pas trouvé la solution optimale (1ère colonne). La colonne C_{tabou} donne la déviation moyenne de la durée totale du meilleur ordonnancement trouvé par RECHERCHETABOU au dessus de la durée optimale. Ces résultats sont proches (1% au dessus) de ceux obtenus par la méthode tabou présentée dans [Pinson et al., 1994]. La colonne ΔC donne en outre l'amélioration moyenne de la durée totale obtenue par RECHERCHETABOU par rapport à la durée totale de l'ordonnancement initial.

L'application d'une méthode tabou plus performante comportant des critères d'aspiration, des intensifications de recherche permettrait sans doute d'améliorer ces résultats qui prouvent néanmoins l'efficacité de la procédure d'insertion en ordonnancement cumulatif multi-ressources, basée sur l'exploration efficace du graphe potentiels-tâches.

IV.1.4.b) Ordonnancement de projet à moyens limités et modes multiples

La deuxième série de problèmes est tirée de [Kolish et Sprecher, 1996] et est référencée *n0*. Il s'agit de problèmes d'ordonnancement de projet comportant des modes multiples et uniquement des ressources renouvelables. Un mode correspond à une occupation des ressources et à une durée fixée.

Ce modèle correspond au cas particulier où pour chaque mode étendu m d'une opération (i, j), une seule ressource peut être sélectionnée dans chaque pool P_{ij}^{mr} et le nombre de lignes de charge utilisées par cette opération dans le mode étendu m est constant ($e_{ij-}^{mk} = e_{ij+}^{mk}$, $\forall k \in \mathcal{R}$). Chaque projet comporte de 10 à 20 opérations, ce qui explique la rapidité de la procédure par rapport au cas précédent. Les résultats obtenus ont été présentés dans [Artigues *et al.*, 1996] et sont rappelés dans la figure IV.4. Nous obtenons pour ces problèmes une diminution moyenne de la durée totale du projet meilleure que dans le cas précédent.

nb sol. à améliorer	nb sol améliorées	C_{tabou}	ΔC	temps CPU
164	135	+1,75%	-7,94%	38 (s)

FIG. IV.4 – *Résultats sur les problèmes de [Kolish et Sprecher, 1996] multi-modes*

IV.2 Amélioration d'un ordonnancement cumulatif multi-ressources avec des opérations de préparation

IV.2.1 Présentation générale

Une méthode de voisinage a été définie dans le paragraphe IV.1 pour un ordonnancement cumulatif multi-ressources. Nous décrivons ici une méthode de voisinage dans un contexte plus général où des opérations de préparation sont nécessaires sur les ressources principales et nécessitent des ressources complémentaires. L'objectif de cette méthode est encore de réduire le plus grand retard T_{max} de l'ordonnancement représenté par un graphe $\mathcal{G}$.

L'état de l'art du paragraphe II.1, donne quelques exemples de l'utilisation d'algorithmes de voisinage donnant de bons résultats pour différents problèmes comportant de la préparation ([Hansmann et Höck, 1995] pour le problème à une machine, [França *et al.*, 1995] pour le problème à machines parallèles, [Parthasarathy et Rajendran, 1997] pour le problème d'atelier à cheminement unique, [Widmer, 91] pour le problème d'atelier à cheminements multiples et contraintes de changement d'outil).

Etant donné la structure non-linéaire des gammes et la présence de ressources cumulatives, notre problème peut être formulé comme un problème d'ordonnancement multi-projets à moyens limités et temps de préparation qui n'a à notre connaissance jamais été étudié. La méthode de voisinage proposée dans ce contexte est basée sur la génération d'un ordonnancement voisin par deux types de transfert d'opération, présentés dans le paragraphe IV.2.2. La suppression d'une opération d'exécution, élément constitutif de la procédure de transfert, est décrite dans le paragraphe IV.2.3. Enfin la méthode de voisinage de type tabou proposée est présentée dans le paragraphe IV.2.4. Des résultats expérimentaux sont présentés dans le paragraphe IV.2.5.

IV.2.2 Méthodes de génération d'un ordonnancement voisin

Un ordonnancement voisin est généré comme au paragraphe IV.1, en transférant une opération. Dans le contexte considéré ici, on est amené à définir deux types de transfert :

type 1 transfert d'une opération d'exécution. Ce transfert est basé sur la suppression de l'opération des ressources sur lesquelles elle est affectée (procédure décrite plus précisément dans le paragraphe IV.2.3) puis sur l'insertion de cette opération à une position définie par l'heuristique d'exploration proposée au paragraphe III.6 dans le contexte de l'insertion avec préparation et ressources complémentaires de préparation.

type 2 transfert d'une opération de préparation ou d'une opération d'exécution limité aux ressources complémentaires. On cherche une nouvelle position pour l'opération de préparation ou d'exécution uniquement sur ses ressources complémentaires définie par la procédure d'exploration décrite en III.3 dans le contexte d'insertion dans un ordonnancement sans préparation. On peut en effet considérer une opération de préparation comme une opération quelconque et chercher à ré-optimiser l'utilisation des ressources complémentaires de préparation, en respectant évidemment la séquence des opérations de préparation sur les ressources principales.

IV.2.3 Suppression d'une opération d'exécution

Comme pour la procédure de transfert en l'absence de temps de préparation, la suppression de l'opération est limitée aux ressources : on conserve dans le graphe $\mathcal{G}$ les arcs entrant et sortant de type gamme, après avoir donné à (i, j) la plus petite durée p_{ij} possible pour donner à (i, j) un impact minimal dans le graphe résultant de la suppression. Soit (i, j) une opération d'exécution à supprimer du graphe $\mathcal{G}$. La suppression de (i, j) entraîne dans $\mathcal{G}$ des modifications plus importantes qu'en l'absence d'opérations de préparation. Ces modifications ont été décrites dans [Artigues et Roubellat, 1997c] et sont résumées ici. La position de (i, j) dans $\mathcal{G}$ est représentée par une position d'insertion

$\mathcal{PI}$. Les opérations de préparation ordonnancées entre chaque opération $p(k,1)$ et (i,j) d'une part et les opérations de préparation ordonnancées entre (i,j) et chaque opération $f(k,1)$ d'autre part, telles que $(k,1,p(k,1),f(k,1)) \in \mathcal{PI}$ forment avec (i,j) un graphe $\mathcal{G}_{ij}$ qu'on doit ici ôter de $\mathcal{G}$. Les opérations de $\mathcal{G}_{ij}$ peuvent être ôtées une à une et être remplacées par les opérations de préparation nécessaires entre chaque paire $(p(k,l),f(k,l))$. La suppression génère ainsi l'insertion dans $\mathcal{G}$ d'un ensemble d'opérations de préparation de remplacement. Après avoir inséré les opérations de remplacement, on peut propager dans $\mathcal{G}$ les nouvelles dates de début au plus tôt des opérations $f(k,l)$ pour connaître l'impact de la suppression sur le plus grand retard.

IV.2.4 Méthode de voisinage de type tabou

Pour définir la procédure tabou dans le contexte considéré, on présente dans un premier temps la méthode retenue pour explorer le voisinage défini par les deux types de transfert (paragraphe IV.2.4.a)). On présente ensuite la gestion de deux listes tabou adaptées aux deux types de transfert retenus (paragraphe IV.2.4.b)). On décrit ensuite l'algorithme élémentaire de recherche tabou RechercheTabouPrepa qui en découle (paragraphe IV.2.4.c)).

IV.2.4.a) Méthode d'exploration du voisinage

L'exploration du voisinage défini en IV.2.2 peut être effectuée selon plusieurs stratégies. On présente ici une exploration du voisinage alternant les transferts de type 1 et les transferts de type 2.

Comme dans le paragraphe IV.2.4.a), on se limite aux opérations de préparation et d'exécution situées sur le chemin critique de $\mathcal{G}$. L'exploration complète du voisinage correspondant au transfert de type 1 suppose de supprimer chaque opération critique et le sous-graphe associé, d'insérer les opérations de préparation de remplacement puis d'explorer l'ensemble des positions d'insertion sur les ressources principales pouvant recevoir cette opération critique en testant pour chaque position d'insertion l'insertion du graphe $\mathcal{G}^*_{ij}$ correspondant (voir III.6). Pour une utilisation pratique de la procédure, ce voisinage doit être par exemple restreint à un sous-ensemble des opérations critiques, voire à une opération critique choisie au hasard ou à l'aide d'une estimation (borne supérieure) de la diminution du plus grand retard causée par la suppression.

On considère d'abord les transferts du type 2, qui correspondent à l'optimisation des ressources complémentaires. On cherche à effectuer des transferts de ce type jusqu'à ce qu'un nombre $nbmaxiter2$ d'itérations sans améliorer le plus grand retard soit atteint. On effectue ensuite un seul transfert du type 1 qui correspond à la modification plus profonde du graphe: la ré-insertion d'une opération d'exécution par la sélection d'une nouvelle position d'insertion sur les ressources principales. On recommence ensuite une série de $nbmaxiter1$ transferts de type 2. On itère la procédure ainsi définie jusqu'à ce qu'un nombre $nbmaxiter1$ d'enchaînement d'un transfert de type 1 et d'un ensemble de transferts de type 2 ait été atteint.

IV.2.4.b) Gestion des listes tabou

Puisqu'on accepte d'effectuer des transferts qui augmentent provisoirement le plus grand retard, on cherche à éviter de revenir à un ordonnancement déjà visité par l'utilisation de listes tabou. On gère deux listes tabou, chacune correspondant à un des deux types de transferts envisagés. Une première liste est utilisée dans l'étape de transfert des opérations de préparation et d'exécution sur les ressources complémentaires et n'est valable que pendant cette étape. Une deuxième liste est utilisée dans l'étape de transfert d'une opération d'exécution à une autre position sur ses ressources principales et est maintenue tout au long de l'exécution de la procédure. Ces listes ont la même structure que celle définie au paragraphe IV.1.3.b).

IV.2.4.c) Algorithme de la procédure RECHERCHETABOUPREPA

Un ordonnancement initial est d'abord généré à l'aide d'un algorithme de liste basé sur une règle de priorité qui a été présenté dans [Artigues et Roubellat, 1996] et qui est décrit au paragraphe V.2. Une fois cet ordonnancement initial obtenu, on lance la procédure RECHERCHETABOUPREPA dont l'algorithme est décrit dans la figure IV.5.

IV.2.5 Résultats expérimentaux sur des problèmes générés aléatoirement

Il n'a pas été possible de trouver dans la littérature des problèmes de test comportant les caractéristiques de préparation définies dans cette étude. Aussi un ensemble de 30 problèmes a été généré aléatoirement. Ces problèmes comportent les caractéristiques suivantes :

L'atelier est composé de 12 ressources principales et de 4 ressources complémentaires. Le nombre de lignes de charge E_k d'une ressource complémentaire k est généré aléatoirement entre 1 et 10. Chaque opération d'exécution ne possède qu'un mode étendu, et à l'intérieur de ce mode, une seule ressource est nécessaire dans chaque pool. L'occupation l_{ij} et la durée sont ainsi fixées. Chaque opération d'exécution nécessite simultanément 3 ressources principales et aléatoirement une ou deux ressources complémentaires. La première ressource principale est choisie aléatoirement parmi les ressources de 1 à 4. La seconde ressource principale est choisie aléatoirement parmi les ressources de 5 à 8. La troisième ressource principale est choisie aléatoirement parmi les ressources de 9 à 12. Le nombre de lignes de charge utilisées par chaque opération d'exécution sur une ressource complémentaire k est généré aléatoirement entre 1 et E_k. Les modes étendus sont tels qu'il n'existe qu'un mode de montage, un mode de démontage et un mode de changement de type valables pour toute association de 3 ressources principales. Il n'existe qu'un mode de changement de type valable pour toutes les ressources principales isolées. Les opérations de préparation utilisent aléatoirement une ou deux ressources complémentaires. Le nombre de lignes de charge utilisées par chaque opération de préparation sur une ressource complémentaire k est généré aléatoirement entre 1 et E_k. Les temps de montage, de démontage et de changement de type sont générés aléatoirement entre 5 et 10. On peut remarquer que ces bornes assurent que l'inégalité triangulaire est toujours respectée. Un ensemble de 10 ordres de fabrication de 15 opérations chacun doit être ordonnancé. Toutes les dates de début au plus tôt des ordres de fabrication sont égales à 0. Les dates

Proc RECHERCHETABOUPRÉPA

$nbiter1 \leftarrow 0$; $L_1 \leftarrow \emptyset$.

Tantque $nbiter1 < nbmaxiter1$ et $T_{max} > 0$ et qu'un transfert de type 1 est possible **faire**

- $nbiter2 \leftarrow 0$; $L_2 \leftarrow \emptyset$
- **Tantque** $nbiter2 < nbmaxiter2$ et $T_{max} > 0$ et qu'un transfert de type 2 est possible **faire**
  - Explorer le voisinage de type 2 et sélectionner une opération $o \in \mathcal{O} \cup \mathcal{S}$ à transférer sur les ressources complémentaires telle que le transfert ne soit pas tabou (liste L_2)
  - Transférer l'opération avec l'occupation sélectionnée dans la clique sélectionnée.
  - Mettre à jour r_o, d_o, $\forall o \in \mathcal{O} \cup \mathcal{S}$ et calculer le nouveau T_{max}.
  - **Si** on améliore le meilleur T_{max} trouvé jusqu'ici, **alors** mémoriser le graphe $\mathcal{G}$ courant et faire $nbiter2 \leftarrow 0$, **sinon** incrémenter $nbiter2$.
    Si aucun transfert n'a été trouvé, **alors** il n'y a plus de transfert de type 2 possible, **Sinon** mettre à jour la liste tabou L_2.

  FinTantQue
- Explorer le voisinage de type 1 et sélectionner une opération $(i,j) \in \mathcal{O}$ à transférer à une nouvelle position d'insertion $\mathcal{PI}(b)$ non tabou (liste L_1) sur les ressources principales.
- Transférer l'opération à la position d'insertion sélectionnée $\mathcal{PI}(b)$ en appliquant HEURINSÈREPRÉPA qui insère une à une les opérations du graphe $\mathcal{G}^*_{ij}$ correspondant sur les ressources principales et complémentaires.
- **Si** on améliore le meilleur T_{max} trouvé jusqu'ici, **alors** mémoriser le graphe $\mathcal{G}$ courant et faire $nbiter1 \leftarrow 0$, **sinon** incrémenter $nbiter1$.
  Si aucun transfert n'a été trouvé, **alors** il n'y a plus de transfert de type 1 possible, **sinon** mettre à jour la liste tabou L_1.

FinTantQue

FinProc

FIG. IV.5 – *Algorithme de* RECHERCHETABOUPREPA

de livraison sont telles que $d_i = \sum_{i=1}^{h_i}(p_{ij} + S_i)$ où S_i est généré aléatoirement entre 100 et 200.

Les résultats de la procédure RECHERCHETABOUPREPA ont été présentés dans [Artigues et Roubellat, 1997d] et sont résumés la figure IV.6. Pour chaque problème, le plus grand retard initial T^{init}_{max} et le plus grand retard obtenu par RECHERCHETABOUPREPA T^{tabou}_{max} sont donnés. La dernière colonne donne le temps de calcul sur une station SUN de type Ultra-2. La taille de la liste tabou est 20 est le nombre maximum d'itérations $nbmaxiter1$ est 100. Le nombre d'itérations $nbmaxiter2$ est ici 0, c'est-à-dire qu'on n'a pour cet exemple autorisé aucun transfert de type 2.

Ces résultats montrent tout d'abord que les délais sont très serrés. Ceci s'explique par le fait que le nombre de ressources est relativement petit par rapport au nombre d'opérations les utilisant. Sur l'ensemble des problèmes la réduction du plus grand retard est de 13,51%. En augmentant le nombre maximum d'itérations jusqu'à 500, le plus grand retard est réduit simplement de 0,2% en plus. On peut en conclure que des améliorations

numéro du problème	T_{max}^{init}	T_{max}^{tabou}	temps cpu (s)
1	4522	4111	1725
2	3850	3623	1725
3	3590	2830	1901
4	3878	3572	2000
5	2584	2244	1510
6	5924	5392	514
7	6827	6369	2200
8	3773	3166	2110
9	4970	4572	845
10	4139	3288	915
11	6419	5799	1785
12	4452	3903	597
13	4765	3879	2433
14	6873	6780	997
15	3002	2532	1181
16	4580	3646	1963
17	5536	5300	2350
18	4060	3088	823
19	2472	2011	982
20	3942	3406	1184
21	3317	2768	2087
22	4203	3369	1042
23	5030	3781	1302
24	3958	3480	1573
25	5246	5131	950
26	5379	5236	93
27	4313	3663	355
28	3490	2909	723
29	3998	2645	1231
30	6298	6068	1692

FIG. IV.6 – *Résultats de* RECHERCHETABOUPREPA *sur 30 problèmes générés aléatoirement*

significatives sont obtenues en quelques itérations. En effet le transfert de type 1 insère et supprime simultanément plusieurs opérations du graphe (tout le sous-graphe $\mathcal{G}_{ij}^*$). L'exploration de ce voisinage génère des temps de calculs longs. Aussi, pour une utilisation pratique de cette procédure, la stratégie de réduction des positions d'insertion explorées ainsi que du nombre d'opérations critiques testées avant un transfert doit être utilisée.

Troisième partie

Construction, suivi et exploitation de la séquence de groupes

Chapitre V

Prise en compte de la préparation dans la méthode ORABAID

Résumé

L'extension de la méthode ORABAID à la prise en compte des opérations de préparation est présentée. En l'absence de temps de préparation, les procédures définies dans les études précédentes sont améliorées, par l'utilisation des algorithmes polynomiaux d'insertion présentés au chapitre III. L'extension et l'amélioration concernent la procédure de génération d'une séquence de groupes d'opérations d'exécution permutables, la procédure de recherche de l'admissibilité de la séquence de groupes et le système interactif d'aide à la décision pour l'exploitation en temps réel de cette séquence. L'extension au logiciel d'Ordonnancement $ORDO^R$ *est présentée.*

Ce chapitre est consacré à la génération, l'amélioration et l'exploitation en temps réel d'une séquence de groupes d'opérations d'exécution permutables lorsque des ressources doivent être préparées selon le contexte décrit en II.2.2. Les procédures décrites ici s'appliquent aussi au cas où les ressources de l'atelier ne nécessitent pas de préparation. Dans ce cas, on considère que l'ensemble des ressources principales $\mathcal{P}_b$ est vide et que les ressources sont toutes complémentaires $\mathcal{R} = \mathcal{R}_c$. Pour faciliter l'exploitation des séquences de groupes, on rappelle en V.1 des conditions nécessaires et suffisantes d'admissibilité d'un groupe d'opérations d'exécution permutables ainsi que le graphe des groupes, représentation de la séquence plus compacte que le graphe des opérations décrit en II.3.3.a). Les procédures d'insertion d'une opération dans un ordonnancement du chapitre III sont étendues à l'insertion dans une séquence de groupes. Le module SINIT, chargé de la génération d'une séquence de groupes réalisable, est décrit en V.2. Le module ANALYSE, chargé de la recherche de l'admissibilité de la séquence de groupes, est décrit en V.3. Le module ARBITRE chargé de l'exploitation en temps réel de la séquence de groupes est décrit en V.4. Enfin l'intégration des activités de préparation dans le logiciel ORDO est présentée dans le paragraphe V.5.

V.1 Définitions préliminaires

V.1.1 Conditions nécessaires et suffisantes d'admissibilité d'un groupe

Un groupe d'opérations d'exécution permutables G est dit admissible si la sélection de chaque permutation possible de ses opérations donne une séquence de groupes admissible. Un groupe est donc admissible si et seulement si la sélection de la permutation la plus défavorable donne une séquence de groupes admissible. Pour obtenir l'ensemble des permutations les plus défavorables, on est amené à considérer deux cas de figure selon que la même opération ou bien deux opérations différentes fixent le $\max_{(i,j)\in G} r_{ij}$ et le $\min_{(i,j)\in G} d_{ij}$. [Thomas, 1980] :

- S'il existe (x, y) telle que $r_{xy} = \max_{(i,j)\in G} r_{ij}$ et (v, w) telle que $d_{vw} = \min_{(i,j)\in G} d_{ij}$ et $(x, y) \neq (v, w)$, alors les permutations les plus défavorables sont celles où (x, y) est en première position et (v, w) en dernière position. La condition nécessaire et suffisante d'admissibilité du groupe G s'écrit :

$$d_{vw} - r_{xy} - \sum_{(i,j)\in G} p_{ij} \geq 0 \tag{V.1}$$

- S'il existe (x, y) telle que $r_{xy} = \max_{(i,j)\in G} r_{ij}$ et $d_{xy} = \min_{(i,j)\in G} d_{ij}$ alors les permutations les plus défavorables sont celles qui placent (x, y) en première position et l'opération (v, w) telle que $d_{vw} = \min_{(i,j)\in G\setminus\{(x,y)\}} d_{ij}$ en dernière position d'une part et celles qui placent (x, y) en dernière position et l'opération (r, s) telle que $r_{rs} = \max_{(i,j)\in G\setminus\{(x,y)\}} r_{ij}$ en première position d'autre part. Les conditions nécessaires et suffisantes d'admissibilité du groupe G s'écrivent alors :

$$d_{rs} - r_{xy} - \sum_{(i,j)\in G} p_{ij} \geq 0 \tag{V.2}$$

$$d_{ij} - r_{vw} - \sum_{(i,j)\in G} p_{ij} \geq 0 \tag{V.3}$$

V.1.2 Représentation d'une séquence de groupes par un graphe des groupes

V.1.2.a) Définition du graphe des groupes

Pour des problèmes de taille réaliste, lorsque le nombre d'opérations dans un même groupe est important, la gestion des graphes définis en II.3.3.a) et II.3.3.b) peut devenir très lourde en raison de la présence d'un arc de type ressource entre chaque paire d'opérations appartenant à deux groupes successifs. Pour un groupe G_2 de n_2 opérations consécutif à un groupe G_1 de n_1 opérations, $n_1 n_2$ arcs de type ressource sont ainsi nécessaires dans les graphes $\mathcal{G}$ et $\mathcal{G}^d$. L'idée de [Thomas, 1980] est de remplacer l'ensemble de ces arcs par un arc unique entre les deux groupes G_1 et G_2. On définit ainsi un graphe des groupes $\mathcal{G}_G$ tel que :

- les sommets de $\mathcal{G}_G$ sont les groupes (on rappelle que toute opération de préparation constitue un groupe à elle seule),
- deux sommets sont reliés par un arc de type gamme s'il existe une contrainte de succession liée à une gamme entre au moins une opération du groupe origine et au moins une opération du groupe destination,
- deux sommets sont reliés par un arc de type ressource s'il existe une contrainte de succession sur une des ressources entre au moins une opération du groupe origine et au moins une opération du groupe destination,
- un arc relie le sommet O à tout groupe contenant la première opération d'un ordre de fabrication,
- un arc relie tout groupe contenant la dernière opération d'un ordre de fabrication au sommet H.

Les arcs de $\mathcal{G}_G$ ne sont pas valués mais il est possible de calculer les dates de début au plus tôt des opérations par propagation d'un potentiel à partir de O dans $\mathcal{G}_G$ en utilisant les expressions analytiques des contraintes de gamme (II.1), de groupe (II.14), et de ressource (II.12) pour calculer la date de début au plus tôt de chaque opération du groupe destination par rapport aux dates de début au plus tôt des opérations du groupe origine. Par propagation à rebours d'un potentiel à partir de H en utilisant les expressions analytiques des contraintes de gamme (II.1), de groupe (II.15), et de ressource (II.12) $\mathcal{G}_G$ permet de calculer les dates de fin au plus tard de chaque opération du groupe origine par rapport aux dates de fin au plus tard des opérations du groupe destination.

Dans $\mathcal{G}_G$, un arc liant deux groupes est dit critique dès qu'il existe sur $\mathcal{G}$ un arc critique entre une opération du groupe origine et une opération du groupe destination. Les deux groupes sont dits critiques.

Le graphe $\mathcal{G}_G$ est décomposé en rangs et chaque groupe G_a^r est le $a$ième groupe du rang r. On définit le nombre de chemins critiques à un rang donné du graphe de la façon suivante. Soit n_1 le nombre de groupes critiques au rang r. Soit n_2 le nombre d'arcs reliant un groupe de rang inférieur à r à un groupe de rang supérieur à r. Le nombre de chemins critiques sur le graphe au rang r est égal à $n_1 + n_2$.

V.1.2.b) Attributs associés aux arcs de type ressource de $\mathcal{G}_G$

Soit u_α un arc de type ressource du graphe $\mathcal{G}_G$. On note $Gorig_\alpha$ le groupe origine de l'arc u_α et $Gdest_\alpha$ le groupe destination de l'arc u_α. Les opérations d'un groupe étant affectées aux mêmes ressources, et sur ces ressources aux mêmes lignes de charge, on peut associer à chaque arc de type ressource u_α du graphe $\mathcal{G}_G$ une occupation des ressources l_α et une occupation des lignes de charge l_α^k sur chaque ressource $k \in \mathcal{R}$. De même, on associe à l'arc u_α un intervalle I_α de date de début au plus tôt r_α^G obtenue en calant le groupe $Gorig_\alpha$ au plus tôt et de date de fin au plus tard d_α^G obtenue en calant le groupe $Gdest_\alpha$ au plus tard :

$$r_\alpha^G = \max_{(i,j)\in Gorig_\alpha} r_{ij} + \sum_{(i,j)\in Gorig_\alpha} p_{ij} \quad \text{(V.4)}$$

$$d_\alpha^G = \min_{(i,j)\in Gdest_\alpha} d_{ij} - \sum_{(i,j)\in Gdest_\alpha} p_{ij} \quad \text{(V.5)}$$

V.1.2.c) Exemple de graphe des groupes

On donne dans la figure V.1, le graphe des groupes correspondant à l'exemple du paragraphe II.3.3.c). On a ainsi

rang 1 $G_1^1 = \{(1,1)\}$, $G_2^1 = \{(4,1),(5,1)\}$ et $G_3^1 = \{(2,1),(3,1)\}$,

rang 2 $G_1^2 = \{sd_{11}\}$, $G_2^2 = \{(4,2)\}$, $G_3^2 = \{(5,2)\}$, $G_4^2 = \{sd_{31}\}$ et $G_5^2 = \{(3,2)\}$,

rang 3 $G_1^3 = \{sct_{12}^1\}$, $G_2^3 = \{sct_{12}^2\}$, $G_3^3 = \{(4,3)\}$ et $G_4^3 = \{(5,3)\}$,

rang 4 $G_1^4 = \{sm_{12}\}$,

rang 5 $G_1^5 = \{(1,2),(2,2)\}$,

$G_1^6 = \{(1,3)\}$ et $G_2^6 = \{(2,3)\}$.

On observe bien une réduction substancielle du nombre d'arcs de type ressource et la disparition des arcs de type groupe.

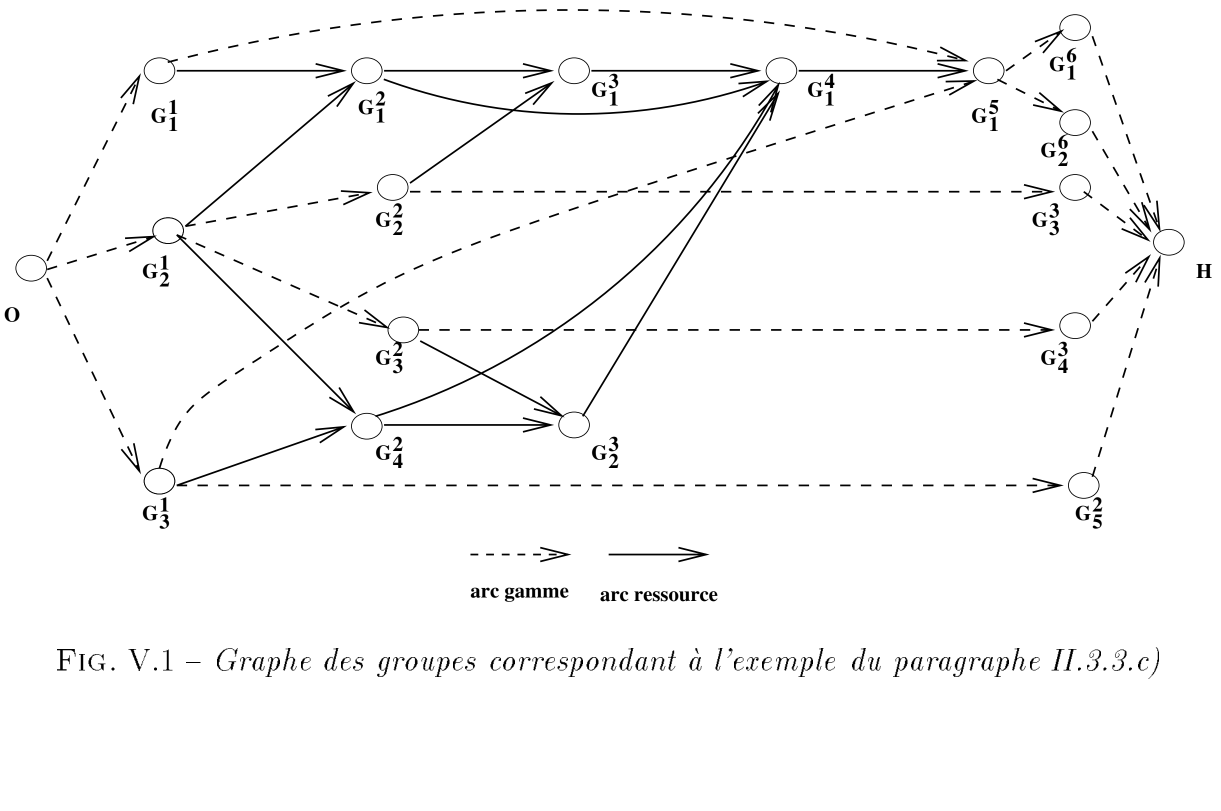


FIG. V.1 – *Graphe des groupes correspondant à l'exemple du paragraphe II.3.3.c)*

V.1.3 Insertion d'une opération dans une séquence de groupes

On considère qu'une opération d'exécution (i, j) doit être insérée dans une séquence de groupes avec les contraintes suivantes :

- pas de changement de la séquence ni de l'affectation des opérations d'exécution déjà ordonnancées,
- violation minimale des dates de fin au plus tard de (i, j) et des opérations déjà ordonnancées.
- création d'un groupe ne contenant que l'opération (i, j) (l'opération n'est pas inclue dans un groupe existant).

Dans ces conditions, on peut se ramener facilement au contexte d'insertion dans un ordonnancement unique (chapitre III) en effectuant les considérations suivantes. Le graphe $\mathcal{G}_G$ représentant la séquence de groupes partielle peut être assimilé à un graphe représentant une séquence d'opérations. En effet (voir paragraphe V.1.2.b)), chaque groupe possède une date de début au plus tôt et une date de fin au plus tard calculées en fonction des dates des opérations qui le composent. Chaque arc de type ressource u_α de $\mathcal{G}_G$ possède un ensemble d'attributs identiques à ceux d'un graphe représentant un ordonnancement (une occupation l_α et un intervalle I_α). La fenêtre temporelle associée à (i, j), si autres opérations de l'ordre de fabrication i sont déjà incluses dans la séquence de groupes, doit tenir compte des contraintes de type groupe pour le calcul de la date de début au plus tôt (équation II.14 du paragraphe II.3.2.c)) et de la date de début au plus tard (équation II.15 du paragraphe II.3.2.c)) de (i, j).

V.2 Module SINIT

Dans ce paragraphe, deux méthodes de génération de la séquence de groupe sont décrites. La première méthode décrite en V.2.2 est basé sur une extension des procédures classiques de génération d'ordonnancements actifs de [Conway *et al.*, 1967] (dites heuristiques de liste). Les inconvénients de cette méthode en présence de temps de préparation sont décrits en V.2.3. Une méthode alternative basée sur l'insertion d'opérations d'exécution est décrite en V.2.4.

V.2.1 Objectifs de la procédure de génération de la séquence initiale

La procédure de génération de la séquence initiale construit la séquence de groupes d'opérations d'exécution permutables et génère dynamiquement les opérations de préparation nécessaires sur les ressources principales. L'objectif principal de la procédure est l'admissibilité, c'est-à-dire la tenue des délais. L'objectif secondaire est de favoriser le rapprochement d'opérations de même classe de préparation (voir définition 1 du paragraphe II.3.1), c'est-à-dire utilisant les mêmes ressources principales et de même type de préparation, afin d'éviter la génération d'opérations de démontage, de changement de type et de montage. Ce rapprochement est appelé un *regroupement technologique*. On peut remarquer, d'après la définition 2 du paragraphe II.3.1, que l'objectif secondaire est compatible avec l'objectif de constitution de groupes d'opérations permutables de plus grand cardinal. Un compromis doit être réalisé entre ces deux objectifs qui peuvent être contradictoires. L'admissibilité, c'est-à-dire le respect des dates de livraison correspond à un critère régulier, alors que la minimisation du nombre d'opérations de préparation générées correspond à un critère non régulier.

V.2.2 Une heuristique de liste pour la génération de la séquence initiale

V.2.2.a) Principes de la méthode

Les heuristiques de liste visent à construire progressivement l'ordonnancement en utilisant des règles de priorité pour résoudre les conflits d'utilisation des ressources.
[Le Gall, 1989] et [Billaut, 1993] ont proposé une extension de la procédure classique de génération d'ordonnancements actifs proposée dans [Conway *et al.*, 1967] pour générer une séquence de groupes initiale. L'intérêt est que l'ensemble des ordonnancements actifs est dominant pour l'admissibilité (critère régulier). Nous reprenons ici les principes de cette méthode, tout en autorisant la génération d'ordonnancements seulement semi-actifs, sous certaines conditions, pour atteindre l'objectif secondaire de minimisation du nombre d'opérations de préparation générées qui correspond à un critère non régulier.

La séquence de groupes est construite par étapes. On considère à chaque étape trois sous-ensembles de l'ensemble des opérations d'exécution : l'ensemble des opérations d'exécution ordonnancées (initialement vide), l'ensemble des opérations d'exécution dont les opérations précédentes sont ordonnancées (initialement égal à l'ensemble des opérations qui n'ont pas de précédentes dans la gamme) appelé ici ensemble des opérations candidates et noté $\mathcal{O}_c$, et enfin l'ensemble des opérations restantes dont au moins une précédente dans la gamme n'est pas ordonnancée. A chaque étape, une opération de $\mathcal{O}_c$ est sélectionnée et incluse dans un groupe à la fin de la séquence partielle, sur un ensemble de ressources principales et complémentaires, après avoir ordonnancé s'il y a lieu les opérations de préparation nécessaires à son exécution sur les ressources principales. L'étape de sélection de cette opération se décompose en trois étapes : l'étape d'affectation de l'ensemble des opérations candidates décrite en V.2.2.b), l'étape de sélection d'un conflit à résoudre décrite en V.2.2.c) et enfin l'étape de résolution du conflit décrite en V.2.2.d). L'étape d'inclusion de l'opération d'exécution dans un groupe est décrite en V.2.2.e).

V.2.2.b) Etape d'affectation des opérations candidates

Cette étape consiste à sélectionner pour chaque opération candidate $(i, j) \in \mathcal{O}c$ une occupation des ressources l_{ij}, c'est-à-dire un mode étendu m, une ressource principale dans chaque pool principal de m, une ressource complémentaire dans chaque pool complémentaire de m et et une occupation des lignes de charge $l_{ij}^k, \forall k \in \mathcal{R}$, c'est-à-dire un ensemble de lignes de charge non vide sur chaque ressource complémentaire cumulative k sélectionnée.

Pour chaque occupation possible l_{ij} et pour chaque occupation des lignes de charge $l_{ij}^k, \forall k \in \mathcal{R}$, l'opération (i, j) étant ajoutée à la fin de la séquence de groupes peut avoir une date de fin au plus tôt différente notée $c_{ij}(l_{ij}^k)$. Cette date dépend des dates de fin au plus tôt des groupes d'opérations d'exécution précédents $Gp(k, l)$ sur chaque ligne de charge l et chaque ressource k telles que $e_{ij}^{kl} = 1$. Elle dépend aussi des opérations de préparation générées entre chaque groupe d'opérations d'exécution précédant $Gp(k)$ sur chaque ressource principale telle que $e_{ij}^{k} = 1$ et (i, j).

Soient l_{min} et l_{min}^k les occupations possibles pour (i, j) qui lui donnent la plus petite date de fin au plus tôt. Soient l_{id} et l_{id}^k les occupations possibles pour (i, j) qui lui donnent la plus petite date de fin au plus tôt parmi l'ensemble des occupations qui permettent d'éviter la génération de toute opération de préparation pour (i, j). On peut remarquer qu'il n'existe pas forcément d'occupation possible l_{id} et l_{id}^k satisfaisant ces conditions.

La sélection de l_{min} et de l_{min}^k correspond à l'objectif principal de génération d'ordonnancements actifs et de recherche de l'admissibilité. En revanche, la sélection de l_{id} et de l_{id}^k correspond à l'objectif secondaire de minimisation du nombre d'opérations de préparation. Elle peut provoquer l'obtention d'ordonnancements non actifs.

En supposant que les occupations l_{id} et l_{id}^k existent pour une opération candidate (i, j) donnée, et que $c_{ij}(l_{id}^k) > c_{ij}(l_{min}^k)$, on donne une règle pour sélectionner l'une ou l'autre des occupations. Pour cela on estime une date de fin au plus tard pour (i, j) en lui allouant une partie de la marge totale de l'ordre de fabrication i. La marge sl_i de l'ordre de fabrication i est calculée à partir de l'affectation qui minimise la date de fin au plus tôt de (i, j) de la façon suivante :

$$sl_i = d_i - c_{ij}(l_{min}^k) - LP((i, j), h_i) \tag{V.6}$$

où $LP((i, j), h_i)$ est le chemin le plus long dans le graphe $\mathcal{G}_i$ entre (i, j) et l'opération h_i en donnant aux opérations de i suivantes de (i, j) leur durée moyenne (puisqu'elles ne sont pas affectées). La date de fin au plus tard d_{ij} est estimée en distribuant à (i, j) une partie de sa marge proportionellement à sa durée :

$$d_{ij} = c_{ij}(l_{min}^k) + p_{ij}\frac{sl_i}{LP((i, j), h_i)} \tag{V.7}$$

Les occupations l_{id} et l_{id}^k ne sont sélectionnées que si on a :

$$c_{ij}(l_{id}^k) \leq d_{ij} \tag{V.8}$$

Obtention de l_{min}, l_{min}^k, l_{id} et l_{id}^k Pour obtenir l_{min}, l_{min}^k, l_{id} et l_{id}^k en évitant d'énumérer l'ensemble des occupations possibles, on se ramène à un problème d'insertion de (i, j) à la fin de la séquence de groupes. On considère l'ensemble des positions d'insertion $\mathcal{PI}_{fin}^{m+}(b) = \{(k, l, Gp(k, l), H)\}$ où $Gp(k, l)$ est le dernier groupe d'opérations d'exécution ordonnancées sur chaque ligne de charge l de chaque ressource k avec $k \in \mathcal{R}_{ij}^{m+}$ (ensemble de toutes les ressources possibles dans le mode m), pour tous les modes étendus m.

- Le problème de recherche de l_{min} et l_{min}^k revient ici à trouver la position d'insertion valide optimale incluse dans chaque position d'insertion $\mathcal{PI}_{fin}^{m+}(b)$. Toutefois, pour trouver une occupation des ressources et une occupation des lignes de charge qui minimisent la date de fin au plus tôt, on ajuste la date de fin au plus tard de l'opération de façon à ce qu'elle soit suffisamment petite pour assurer l'exploration complète de l'arbre (par exemple $d_{ij} = 0$).

Le graphe $\mathcal{G}_{ij}^*$ à insérer obtenu par la procédure est limité aux opérations de préparation précédant (i,j).

– Pour obtenir l_{id} et l_{id}^k, on applique la procédure d'énumération des positions d'insertion correspondant à une association déjà positionnées (procédure CliqueOccupPrépaAssocConstOpt décrite dans le paragraphe III.5.2) incluses dans $\mathcal{PI}_{fin}^{m+}(b)$. Parmi ces positions d'insertion, on sélectionne, si elle existe, celle qui évite la génération d'une opération de changement de type. Parmi toutes les positions satisfaisant ce critère, on sélectionne enfin cette qui donne à (i,j) la plus petite date de fin au plus tôt. Si une telle occupation n'est pas trouvée, on pose $l_{id} = \emptyset$ et $l_{id}^k = \emptyset$.

L'utilisation des procédures d'insertion décrites dans le chapitre III permet ainsi d'éviter l'énumération de l'ensemble des occupations des lignes de charge possibles (méthode proposée dans [Billaut, 1993]) pour trouver l'affectation qui minimise la date de fin au plus tôt, ce qui entraîne des gains significatifs en complexité lorsque l'aspect multi-ressources est prédominant. Toutefois, on rappelle qu'en présence de ressources complémentaires de préparation, cette faible complexité est obtenue en déterminant la position d'insertion de façon heuristique (voir paragraphe III.6).

V.2.2.c) Etape de sélection d'un conflit à résoudre

L'ensemble des affectations ainsi réalisées sur les ressources principales et complémentaires crée un ensemble de conflits d'utilisation des ressources. La sélection de l'opération à ordonnancer est effectuée dans l'ensemble des opérations éligibles $\mathcal{O}_e$ contenant l'opération candidate (r,t) de plus petite date de fin au plus tôt et l'ensemble des opérations candidates en conflit avec (r,t). Cette sélection est conforme à la génération d'ordonnancements actifs.

Soit (r,t) l'opération de $\mathcal{O}c$ telle que :

$$c_{rt} = \min_{(i,j)\in\mathcal{O}_c} c_{ij} \tag{V.9}$$

Pour chaque opération éligile o, on considère le graphe $\mathcal{G}_o^*$ contenant o et l'ensemble des opérations de préparation à insérer avant o sur l'ensemble des ressources principales définies par l'occupation l_o retenue lors de l'étape d'affectation. On va déterminer l'ensemble des opérations en conflit avec (r,t) en prenant en compte les conflits possibles des opérations de préparation sur les ressources complémentaires. Soit $\mathcal{G}_{rt}^*$ le graphe des opérations incluant (r,t) et ses opérations de préparation précédentes. L'ensemble $\mathcal{O}_e$ est l'ensemble des opérations (i,j) telles que $\exists k \in \mathcal{R}$, $\exists o \in \mathcal{G}_{ij}^*$, $\exists o' \in \mathcal{G}_{rt}^*$

$$l_o^k \cap l_{o'}^k \neq \emptyset_l \tag{V.10}$$

$$r_o < c_{o'} \tag{V.11}$$

Il s'agit de l'ensemble des opérations d'exécution (i,j) pour lesquelles une opération du graphe à insérer $\mathcal{G}_{ij}^*$ est en conflit avec une opération du graphe à insérer $\mathcal{G}_{rt}^*$ sur au moins une ligne de charge de l'atelier.

V.2.2.d) Etape de résolution du conflit

On estime l'urgence relative des opérations d'exécution de l'ensemble $\mathcal{O}_e$ en calculant pour chacune d'elles un indice priorité cr_{ij}. La règle de priorité choisie est $SLACK/RPT$ (*slack per remaining processing time*) si $sl_i > 0$ et $SLACK \times RPT$ (*slack times remaining processing time*) si $sl_i < 0$ (voir [Billaut, 1993]). Cette règle est basée à la fois sur la marge des opérations ($SLACK$) et sur le temps estimé restant à faire dans l'ordre de fabrication (RPT). Pour chaque opération (i,j) de l'ensemble $\mathcal{O}_e$, on calcule :

$$cr_{ij} = \frac{s_{ij}}{LP((i,j),h_i)} \quad \text{si} \quad s_{ij} \geq 0 \tag{V.12}$$

$$cr_{ij} = s_{ij} \times LP((i,j),h_i) \quad \text{si} \quad s_{ij} < 0 \tag{V.13}$$

Soit (a,b) l'opération de $\mathcal{O}_e$ de plus petit cr. La sélection de (a,b) pour inclusion dans la séquence partielle de groupes a pour but de tenir les délais (objectif principal).

S'il existe dans $\mathcal{O}_e$ une opération (u,v) telle que :

- $cr_{uv} > cr_{ab}$,
- (u,v) et (a,b) ont au moins une ressource principale k commune
- (u,v) utilise un ensemble de ressources principales déja préparées ($\mathcal{G}^*_{uv} = \{(u,v)\}$),

alors la sélection de (u,v) favorise la réalisation de regroupements technologiques (objectif secondaire). Dans ce cas, on sélectionne l'opération (u,v) à la place de (a,b) uniquement si le décalage de la date de début au plus tôt de (a,b) dû à la sélection de (u,v) est compatible avec la date de fin au plus tard estimée d_{ab}. Pour estimer ce décalage, on génère le graphe $\mathcal{G}^{*'}_{ab}$ nécessaire si (u,v) est sélectionnée avant (a,b), de façon à calculer la nouvelle date de fin au plus tôt de (a,b).

Dans le cas où l'opération la plus urgente a un coefficient négatif, elle est sélectionnée systématiquement car elle n'a plus de marge et l'ordre de fabrication correspondant est déja en retard. Si plusieurs opérations ont un coefficient négatif, plusieurs ordres de fabrication sont en retard. SINIT propose la sélection de l'opération de plus petit coefficient. Le décideur peut intervenir pour sélectionner une autre opération [Billaut, 1993].

V.2.2.e) Etape d'inclusion de l'opération d'exécution sélectionnée dans un groupe

L'opération d'exécution sélectionnée (i,j) doit être incluse dans la séquence de groupes. On distingue alors deux cas :

- Si l'opération (i,j) ne génère pas d'opérations de préparation ($\mathcal{G}^*_{ij} = \{(i,j)\}$), alors on tente d'insérer (i,j) dans le dernier groupe G ordonnancé sur l'ensemble des lignes de charge défini par l_{ij} et $l^k_{ij}, \forall k \in \mathcal{R}$, si ce groupe existe.

  Pour cela, trois conditions doivent être vérifiées. La première est que le groupe G ne doit pas être fermé. Un groupe devient *fermé* lorsque :

  - une opération suivante d'une des opérations de G au sens de la gamme est incluse dans la séquence,

– un groupe suivant G sur au moins une de ses lignes de charge est créé.

La deuxième condition nécessite les tests de permutabilité de (i,j) et des opérations du groupe G donnés en V.1.1. La troisième condition à vérifier pour pouvoir insérer (i,j) dans G (voir définition 2 du paragraphe II.3.1) est l'identité des occupations des lignes de charge entre (i,j) et les opérations de G : pour tout $k \in \mathcal{R}$, $\forall o \in G$, $l_{ij}^k = l_{ij}$. Si les conditions de permutabilité sont vérifiées, si le groupe n'est pas fermé mais si la condition d'identité des occupations n'est pas vérifiée, le réajustement de l'occupation de (i,j) est envisagé. Ce réajustement est effectué sous quatre conditions :

1. (i,j) utilise exactement les mêmes ressources principales que G,
2. le réajustement est possible compte tenu des possibilités d'affectation de (i,j),
3. il ne remet pas en cause l'admissibilité de la séquence (pas de violation de la date de début au plus tard de G),
4. il ne remet pas en cause les conditions de permutabilité.

Si ces conditions ne sont pas vérifiées, alors un nouveau groupe ne contenant que (i,j) est créé et ajouté au graphe des groupes.

– Si l'opération (i,j) génère des opérations de préparation alors elles sont insérées une par une selon le principe de la procédure HeurInsèrePrépa définie en III.6. Pour cela, un groupe par opération de préparation est créé et ajouté au graphe des groupes. Enfin, un groupe ne contenant que l'opération (i,j) est ajouté à la séquence.

A l'issue de cette étape, l'ensemble des opérations candidates est augmenté de l'ensemble des opérations suivantes de (i,j) dans la gamme de l'ordre de fabrication i dont toutes les opérations précédentes sont incluses dans la séquence partielle. La procédure est itérée jusqu'à ce que $\mathcal{O}_c = \emptyset$

V.2.3 Inconvénient des heuristiques de liste classiques en présence de temps de préparation

La méthode proposée dans le paragraphe précédent utilise, lorsque les délais sont serrés, la méthode classique de génération des ordonnancement actifs (algorithme de liste) initialement proposée pour l'atelier à cheminements multiples par [Giffler et Thompson, 1960].

L'utilisation d'algorithmes de liste est justifiée par le fait qu'il existe une liste des opérations Li^{opt} telles que la sélection des opérations par un algorithme de liste (encore appelée méthode sérielle) dans l'ordre défini par Li^{opt} mène à la solution optimale pour tout critère régulier. Dans [Ovacik et Uzsoy, 1993], il est prouvé que cette affirmation n'est plus vraie en présence de temps de préparation dépendant de la séquence pour un algorithme de liste générant des ordonnancements sans délai. Dans ce paragraphe, on montre à l'aide d'un contre-exemple, que dans un problème d'atelier à cheminements multiples et temps de préparation dépendant de la séquence, l'affirmation n'est plus vraie pour l'algorithme de liste de [Giffler et Thompson, 1960].

Dans l'exemple très simple de la figure V.2, on considère un atelier à cheminements multiples à deux machines et deux ordres de fabrication. L'ordre de fabrication 1 est tel que $(1,1)$ doit être exécutée sur $M1$ et $(1,2)$ doit être exécutée sur $M2$. L'ordre de fabrication 2 est tel que $(2,1)$ doit être exécutée sur $M2$ et $(2,2)$ doit être exécutée sur $M1$. Seule la machine $M1$ nécessite de la préparation. Les temps de changement de type sont indiqués dans la matrice donnée dans la figure. Les durées des opérations sont indiquées dans la figure. On suppose que les ordres de fabrication 1 et 2 ont une date de livraison commune égale à 14 et une date de début au plus tôt commune égale à 0. L'algorithme classique de génération d'ordonnancements actifs place obligatoirement l'opération $(1,1)$ en tête de la ressource. En effet, l'ensemble initial des opérations candidates est ici $\mathcal{O}_c = \{(1,1),(2,1)\}$. Or $c_{11} = 3$ et $c_{21} = 5$. D'où comme $c_{11} < c_{21}$, on choisit de résoudre le conflit concernant $(1,1)$. $(1,1)$ est ainsi placée en premier quelle que soit la règle de priorité car il n'y a pas d'opérations en conflit avec elle. Un exemple d'un tel ordonnancement est donné sous la forme d'un diagramme de Gantt en haut de la figure V.2. Quel que soit l'ordre choisi ensuite sur $M2$ pour $(2,1)$ et $(1,2)$, le plus grand retard de l'ordonnancement A ainsi défini est $T_{max} = 1$ car $(2,2)$ se termine à la date 15. A n'est pas admissible.

En bas de la figure V.2, l'ordonnancement (B) représenté place $(2,2)$ avant $(2,1)$, ce que ne peut faire l'algorithme de liste classique car cela suppose de placer $(2,1)$ en premier, ce qui est incompatible avec la génération d'ordonnancement actif en l'absence de préparation. Or, l'ordonnancement B est toujours actif car on ne peut insérer $(1,1)$ avant $(2,2)$ sans décaler vers la droite cette opération, à cause de la préparation. Cet ordonnancement a un plus grand retard de 0 et est donc admissible, c'est-à-dire meilleur que n'importe quel ordonnancement généré par la procédure classique.

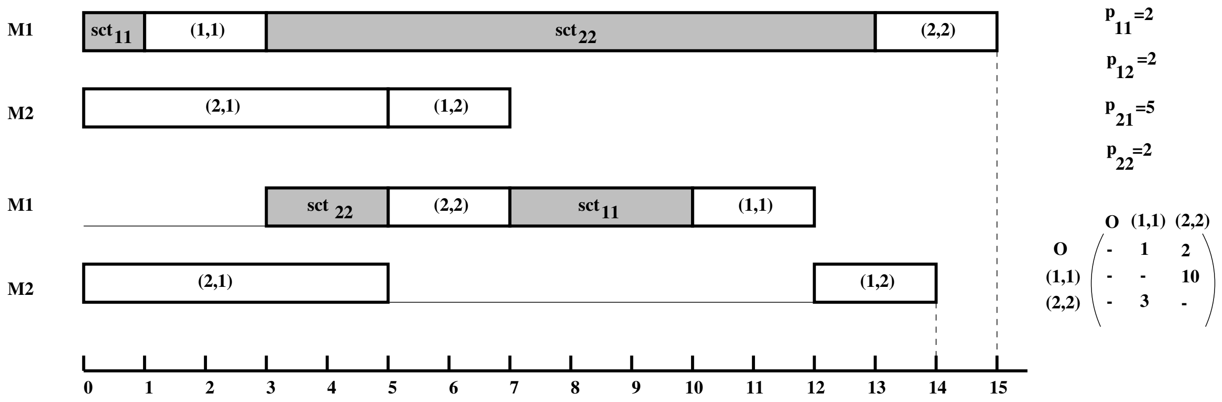


FIG. V.2 – *Contre-exemple dans le cas d'un atelier à cheminements multiples à deux machines*

On peut en conclure qu'en présence d'opérations de préparation, certains ordonnancements actifs ne peuvent pas être générés par une heuristique de liste classique. Par conséquent, l'existence a priori d'un ordonnancement optimal généré par une heuristique de liste n'est plus garantie.

Pour tenter de corriger ce défaut, on utilise l'observation suivante. Soit l'ordonnancemenent partiel composé de $(1,1)$ sur $M1$, et de $(2,1)$ suivi de $(1,2)$ sur $M2$. Cet ordonnancement partiel est obtenu par l'algorithme classique des heuristiques de liste. Si au lieu d'ajouter $(2, 2)$, à la fin de la séquence sur $M1$, ce qui mène à l'ordonnancement A non admissible, on insère $(2, 2)$ avant $(1, 1)$, on obtient l'ordonnancement B admissible. Ceci suggère la mise en oeuvre d'une méthode basée sur l'insertion. Dans le paragraphe V.2.4, la méthode de génération d'une séquence de groupes initiale proposée en V.2.2 est modifiée dans ce sens.

V.2.4 Une méthode de génération alternative basée sur l'insertion

V.2.4.a) Présentation de la procédure

On modifie uniquement l'étape d'inclusion de l'opération sélectionnée (i, j) dans un groupe décrite dans V.2.2.e) de telle sorte qu'à cette étape (i, j) soit non pas systématiquement ajoutée à la séquence partielle de groupes mais éventuellement insérée à l'intérieur de cette séquence.

Pour pouvoir appliquer la procédure d'insertion du chapitre III étendue à une séquence de groupes dans le paragraphe V.1.3 à (i, j) et au graphe $\mathcal{G}_G$ il reste à définir une date de fin au plus tard pour (i, j) et les opérations déja incluses dans la séquence. Les dates de début au plus tôt sont parfaitement déterminées car l'ensemble des opérations précédentes de (i, j) dans g_i sont déja incluses.

Pour déterminer les dates de fin au plus tard de ces opérations, on considère uniquement les arcs de type gamme pour les opérations non ordonnancées. On donne à chaque opération non ordonnancée sa durée moyenne dans l'ensemble de toutes ses durées possibles. On obtient ainsi par propagation une date de fin au plus tôt C_u estimée pour chaque ordre de fabrication u, ainsi qu'une marge $sl_u = D_u - C_u$ En propageant à rebours depuis chaque sommet h_i le potentiel $\max(C_i, d_i)$, on obtient les dates de fin au plus tard souhaitées. Pendant la propagation les poids des arcs de type gamme issus des opérations (u, v) non incluses dans la séquence sont augmentés, si $sl_u > 0$, de la valeur

$$\frac{sl_{u,v} p_{uv}}{LP((u, x), h_u)}$$

qui correspond à la distribution de la marge de u à (u, v) proportionellement à sa durée. (u, x) est telle que :

- (u, x) fait partie des opérations de u incluses dans la séquence mais qui possèdent une suivante non incluse dans la séquence
- $LP((u, x), h_u)$ est le plus long chemin dans la gamme g_u parmi tous les chemins issus d'une opération de u qui vérifie la propriété précédente.

Ceci a pour but de ne pas surévaluer les dates de fin au plus tard des opérations déja incluses dans la séquence de groupes partielle, puisque la propagation dans la partie non incluse ne tient pas compte des contraintes de ressource.

On peut alors appliquer à (i, j) et au graphe $\mathcal{G}_G$, la procédure d'insertion du chapitre III. On trouve la position d'insertion pour (i, j) qui minimise l'impact de (i, j) sur l'admissibilité estimée de la séquence partielle.

V.2.4.b) Exemple

On donne dans ce paragraphe un exemple d'application de la procédure d'insertion en reprenant l'exemple du paragraphe V.2.3.

On considère la séquence de groupes partielle dont le graphe est représenté en haut de la figure V.3. Toutes les opérations d'exécution ont été incluses dans la séquence, à l'exception de l'opération $(2,2)$. Aussi, le graphe $\mathcal{G}_G$ représenté ne contient que les arcs de type gamme liés à $(2,2)$. Parmi les groupes déja générés, on trouve G_1^1 contenant l'opération de préparation s_{11}, G_2^1 contenant l'opération d'exécution $(2,1)$, G_1^2 contenant l'opération $(1,1)$ et G_1^3 contenant l'opération $(1,2)$.

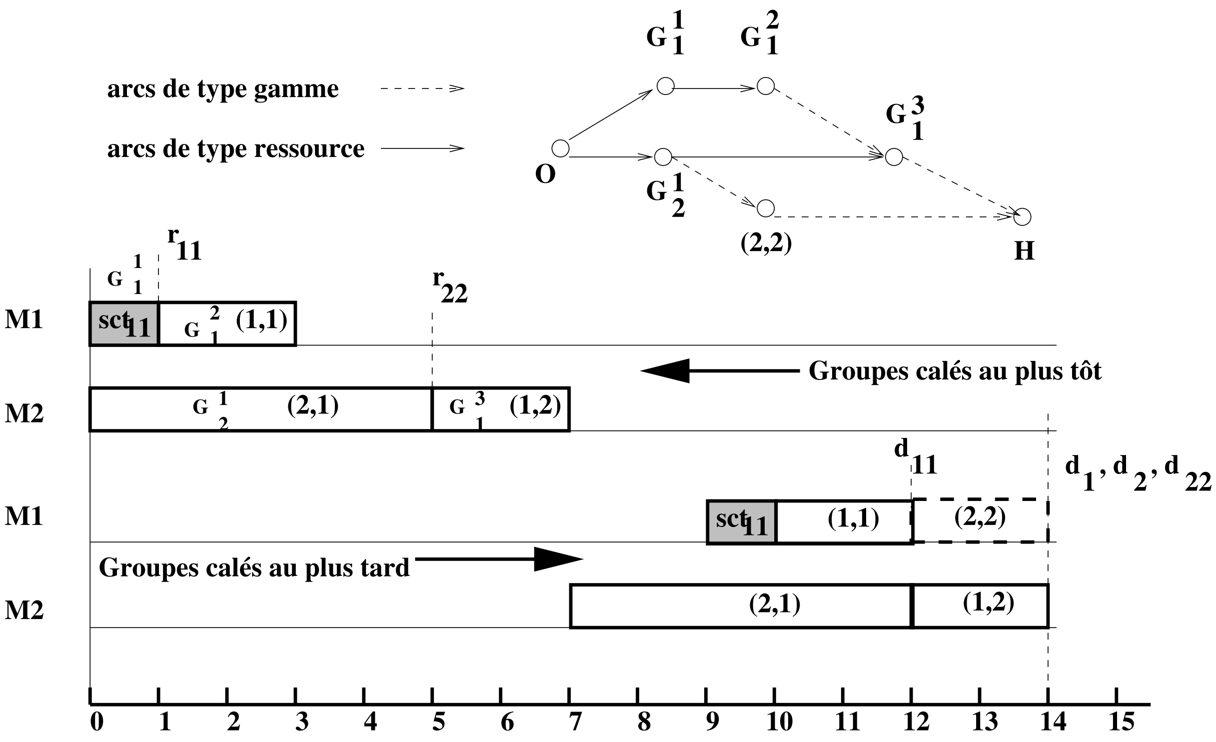


FIG. V.3 – *Séquence de groupes partielle*

Avant d'appliquer la procédure d'insertion à l'opération $(2,2)$ sur la ressource $M1$, on doit calculer les dates de début au plus tôt et de fin au plus tard associées aux groupes d'opérations d'exécution séquencés sur $M1$ et à l'opération $(2,2)$. Par propagation dans le graphe on obtient les dates de début au plus tôt associées aux opérations: $r_{11} = 1$, $d_{11} = 12$, $r_{22} = 5$, $d_{22} = 14$. En effet, $(2,2)$ étant la dernière opération de l'ordre de fabrication 2, la marge sl_2 est entièrement attribuée à $(2,2)$. Chaque groupe ne contenant qu'une opération, les dates associées aux groupes sont les mêmes que les dates associées aux opérations. Par souci de clarté, on utilise dans cet exemple les notations des opérations plutôt que celle des groupes.

Les dates étant calculées, on peut appliquer la procédure CLIQUEOPTPRÉPA décrite au paragraphe III.4. En effet, on se trouve dans un contexte où l'occupation de $(2,2)$ est fixée et où il n'y a aucune ressource complémentaire. On rappelle que toute position d'insertion valide sur les ressources principales (ici ressource principale unique $M1$) est représentée par un arc de type ressource fictif $\tilde{u}_{22}$.

On utilise d'abord la procédure GÉNÈRECLIQUEMAXINITPRÉPA (voir paragraphe III.4.4.b)) pour générer la clique initiale dominante. On a $\mathcal{R}_{22} = \{M1\}$. On considère la clique initiale $\tilde{\mathcal{C}}^{M1}_{init} = \{\tilde{u}^0_{22}\}$ où $\tilde{u}^0_{22}$ est l'arc fictif correspondant à la position d'insertion définie par le 4-uplet $\{(M1, 1, O, (1, 1))\}$. Les dates de début au plus tôt et de fin au plus tard associées à cet arc sont :

$$\begin{aligned} r_{\tilde{u}^0_{22}} &= p_{sct}^{1RT_O RT_{22}\{M1\}1} \\ &= 2 \\ d_{\tilde{u}^0_{22}} &= d_{11} - p_{11} - p_{sct}^{1RT_{22} RT_{11}\{M1\}1} \\ &= 12 - 2 - 3 \\ &= 7 \end{aligned}$$

Dans le cas général, à partir de la clique $\tilde{\mathcal{C}}^{M1}_{init}$, on génère la clique suivante par déplacement vers la droite autour de l'opération o_{dest} qui entraîne la plus faible augmentation de la date de début au plus tôt $r_{\tilde{u}^0_{22}}$. Cette opération est déterminée par la procédure CHOIXODESTMINΔR qui calcule aussi la contribution r_{min} de o_{dest} dans la date de début au plus tôt de la clique suivante. Ici, $(1, 1)$ est la seule opération pivôt possible et $r_{min} = r_{\tilde{u}^1_{22}}$ où $\tilde{u}^1_{22}$ est l'arc fictif correspondant à la position d'insertion définie par le 4-uplet $(M1, 1, (1, 1), H)$. On a ainsi $r_{min} = t_{11} + p_{11} + p_{sct}^{1RT_{11} RT_{22}\{M1\}1} = 1 + 2 + 10 = 13$.

Comme $r_{min} > r_{ij}$, $\tilde{\mathcal{C}}^{M1}_{init}$ est la clique intitiale dominante au regard des dates. cette clique correspond à l'insertion de $(2, 2)$ avant $(1, 1)$. La compatibilité de $\tilde{\mathcal{C}}^{M1}_{init}$ avec $(2, 2)$, c'est à dire la non existence dans le graphe d'un chemin entre $(1, 1)$ et $(2, 1)$ doit être vérifiée. Cette vérification est effectuée par la procédure TESTECHEMINPRECIJ. La condition $r_{11} + p_{11} > r_{21}$ suffit pour établir la non existence du chemin sans parcourir le graphe. $\tilde{\mathcal{C}}^{M1}_{init}$ est bien la clique maximale dominante initiale. Le calcul de l'impact sur l'admissibilité donne

$$\begin{aligned} \Delta d_{max}(2, 2, \mathcal{PI}(\tilde{\mathcal{C}}^{M1}_{init})) &= \max\{0, \max(r_{\tilde{u}^0_{22}}, r_{22}) + p_{22} - \min(d_{\tilde{u}^0_{22}}, d_{ij})\} \\ &= \max(0, 5 + 2 - 7) \\ &= 0 \end{aligned}$$

Cet impact étant nul, la clique est retenue et $(2, 2)$ est insérée avant $(1, 1)$ sur la ressource $M1$.

V.3 Module ANALYSE

L'utilisation de l'une ou l'autre des procédures de génération de la séquence de groupes initiale ne peut garantir l'admissibilité de la séquence de groupes générée. La recherche de l'admissibilité constitue alors une deuxième étape et est effectuée par le module ANALYSE. Cette recherche est basée sur l'amélioration itérative de la séquence issue de SINIT, en s'appuyant sur l'analyse du chemin critique du graphe potentiels-tâches représentant la séquence. Deux principes d'amélioration sont appliqués de façon itérative :

- l'exclusion d'ordonnancements non admissibles par *scission* d'un groupe d'opérations d'exécution, décrite en V.3.1,

– le *transfert* d'une opération d'exécution ou de préparation, décrit en V.3.2

A la fin de l'analyse, une phase de regroupements décrite en V.3.3 cherche à augmenter le nombre d'opérations de chaque groupes d'une part et à rapprocher des opérations de même classe de préparation d'autre part, sans remettre en cause l'admissibilité (ou les plus petites dates de fin au plus tôt obtenues). L'algorithme général du module ANALYSE est décrit en V.3.4

V.3.1 Recherche de scissions

Le principe de scission d'un groupe a été défini dans [Thomas, 1980], puis repris dans les travaux de [Le Gall, 1989] et [Billaut, 1993]. L'idée est de réduire la longueur du chemin critique en scindant en deux certains groupes critiques. A partir d'un groupe G_α^r, on obtient deux groupes G_α^r et G_β^{r+1}. Cette manipulation ne peut concerner que des groupes de plus d'une opération qui sont des groupes d'opérations d'exécution. On reprend ainsi les règles établies dans [Thomas, 1980] qui sont les suivantes :

– Il est inutile de scinder un groupe dont toutes les opérations sont sur un chemin critique de $\mathcal{G}$.

– Dans le cas où seulement une partie des opérations du groupe sont critiques, alors une scission telle que toutes les opérations critiques appartiennent à G_β^{r+1} et les opérations non critiques au groupe G_α^r peut amener la longueur du chemin critique à diminuer.

Seule la scission d'un groupe G_α^r situé sur tous les chemins critiques de $\mathcal{G}_G$ peut diminuer la longueur du chemin critique. S'il y a plusieurs chemins critiques au rang r, la scission de $\mathcal{G}_G$ peut toutefois être envisagée pour diminuer le nombre de chemins critiques. On peut ainsi rechercher les scissions des groupes G_α^r par ordre croissant du nombre de chemins critiques au rang r.

V.3.2 Recherche de transferts

Nous avons défini au chapitre IV le principe du transfert d'une opération dans un ordonnancement unique, dans le cas sans préparation au paragraphe IV.1.2 et dans le contexte général avec préparation au paragraphe IV.2.2. Dans ce paragraphe, nous adaptons les procédures d'amélioration par voisinage RechercheTabou et RechercheTabouPrepa aux transferts d'opérations (d'exécution et de préparation) dans une séquence de groupes d'opérations permutables.

L'idée est d'assimiler le graphe des groupes $\mathcal{G}_G$ à un graphe d'opérations représentant un ordonnancement unique, ce qui revient à considérer chaque groupe comme une opération. On limite les transferts aux opérations situées dans des groupes critiques. Si le transfert concerne un groupe critique G_α^r contenant plusieurs opérations d'exécution, il s'agit de sélectionner l'opération à transférer. Dans [Le Gall, 1989], une étude détaillée a conduit aux remarques suivantes :

– si G_α^r contient une opération critique dont une suivante dans la gamme est aussi critique, alors seul le transfert de cette opération est à envisager.

– sinon, on peut envisager le transfert de n'importe quelle opération du groupe. Le choix se porte sur l'opération qui possède la plus grande fenêtre temporelle induite par les contraintes de gamme. En effet, cette fenêtre définit le domaine de recherche de la procédure d'exploration. Aussi, plus la fenêtre est grande, plus on a de chance de trouver une position d'insertion admissible.

Une fois l'opération d'exécution ou de préparation o sélectionnée, on applique la procédure d'exploration sur le graphe $\mathcal{G}_G$ en suivant l'algorithme des procédures RECHERCHETABOU et RECHERCHETABOUPREPA. A la position d'insertion sélectionnée, on insère un groupe par opération (de préparation et d'exécution) du graphe G_o^* généré si o est une opération d'exécution ou un groupe ne contenant que o si o est une opération de préparation. On ne cherche pas ici à insérer l'opération dans un groupe déjà existant, l'objectif étant la recherche de l'admissibilité et non l'augmentation du nombre d'opérations dans les groupes (voir paragraphe V.3.3).

Par rapport à la procédure de transfert décrite dans [Le Gall, 1989] et [Billaut, 1993], la procédure présentée ici intègre la gestion des opérations de préparation et possède les avantages supplémentaires suivants :

– Complexité polynomiale de l'exploration du voisinage,

– Possibilité de changement de position dans la séquence sans changement d'affectation,

– Possibilité de dégradation provisoire de l'admissibilité (méthode tabou).

V.3.3 Regroupements

On définit deux procédures de regroupement. La première procédure consiste à réduire le nombre d'opérations de préparation présentes dans la séquence de groupes. Ce regroupement, appelé regroupement technologique, est décrit dans le paragraphe V.3.3.a). La deuxième procédure, appelée fusion, consiste à augmenter le nombre d'ordonnancements caractérisés par la séquence de groupes. Elle est décrite dans le paragraphe V.3.3.b).

V.3.3.a) Regroupement technologique

Dans le respect de l'admissibilité qui reste l'objectif principal, on peut chercher à transferer une opération d'exécution d'une position où elle génère un ensemble d'opérations de préparation vers une autre position où les ressources sont déja preparées pour la recevoir. Si une position d'insertion de ce type est trouvée, alors le nombre d'opérations de préparation présentes après le transfert est inférieur au nombre d'opérations de préparation présentes avant le transfert, ce qui correspond à l'objectif secondaire. Le procédure qui réalise les regroupements technologique applique à chaque opération isolée (i, j) dans un groupe de $\mathcal{G}_G$ et entourée d'opération de préparation la procédure suivante. Pour rechercher de telles positions d'insertion, on applique la procédure d'énumération des positions d'insertion correspondant à une association déja constituée (procédure CLIQUEOCCUPPRÉPAASSOCCONSTOPT décrite dans le paragraphe III.5.2) dans le graphe $\mathcal{G}_G$. Parmi ces positions d'insertion $\mathcal{PI}$, on ne regarde que celles dont la configuration (voir paragraphe III.4.2) évite aussi la génération d'opérations de changement de type et qui sont telles que $\Delta d_{max}(i, j, \mathcal{PI}) = 0$ (respect de l'admissibilité)

V.3.3.b) Fusion

Ce principe de regroupement a été défini dans [Thomas, 1980], puis repris dans les travaux de [Le Gall, 1989] et [Billaut, 1993]. Il s'agit de regrouper deux groupes d'opérations d'exécution adjacents $G_\alpha^{r_1}$ et $G_\beta^{r_2}$ en un seul groupe sans remettre en cause l'admissibilité.

Pour cela, le regroupement doit être :

correct au sens de la définition d'un groupe, c'est-à-dire que les deux groupes $G_\alpha^{r_1}$ et $G_\beta^{r_2}$ doivent occuper exactement les mêmes lignes de charge.

réalisable , c'est-à-dire tel qu'aucune des permutations après fusion ne doit générer de cycle dans le graphe. Pour vérifier cette condition, on teste les conditions suffisantes de compatibilité pour l'insertion de toutes les opérations du groupe $G_\beta^{r_2}$ avant le groupe $G_\alpha^{r_1}$ (voir procédure TESTECHEMINPRECIJ du paragraphe III.4.4.b)).

admissible , c'est-à-dire que les conditions nécessaires et suffisantes de permutabilité données en V.1.1 soient vérifiées. On teste ces conditions après avoir calculé les nouvelles dates de début au plus tôt des opérations de $G_\beta^{r_2}$ et les nouvelles dates de fin au plus tard de $G_\alpha^{r_1}$.

On peut faire remarquer que la première condition pénalise la constitution de groupes dans un contexte fortement cumulatif ou multi-ressources. Aussi, dans le cas où seule la première condition n'est pas vérifiée et si les deux groupes occupent exactement les mêmes ressources principales (cas de figure qui est favorisé par l'étape précédente de regroupements technologiques), on cherche à ajuster l'occupation d'un des deux groupes sur les ressources complémentaires pour assurer l'égalité. On effectue pour cela les mêmes tests que pour l'inclusion d'une opération dans un groupe lors de la génération de la séquence initiale (V.2.2.e)). On doit en plus s'assurer que ce transfert local aux lignes de charge des ressources complémentaires est d'impact nul sur l'admissibilité.

V.3.4 Algorithme général du module ANALYSE

L'algorithme général du module ANALYSE présenté dans la figure V.4 donne l'enchaînement des procédures de scission, de recherche tabou et de regroupements. Un premier tri des ordres de fabrication en retard est effectué par le décideur en entrée de la procédure d'ANALYSE. Ce tri conduit à accepter les retards de certains ordre de fabrication, jugés peu importants [Billaut, 1993]. A l'issue d'un enchaînement d'une recherche de scission et d'une recherche de transferts, le décideur peut poursuivre l'analyse s'il souhaite obtenir une séquence plus proche de l'admissibilité. Dans ce cas, les retards d'une partie des ordres de fabrication peuvent être acceptés (ce qui revient à décaler les dates de livraison) avant de relancer le processus. Le décideur peut aussi accepter tous les retards (s'il y en a) et on passe alors aux étapes de regroupements technologiques, puis de fusions.

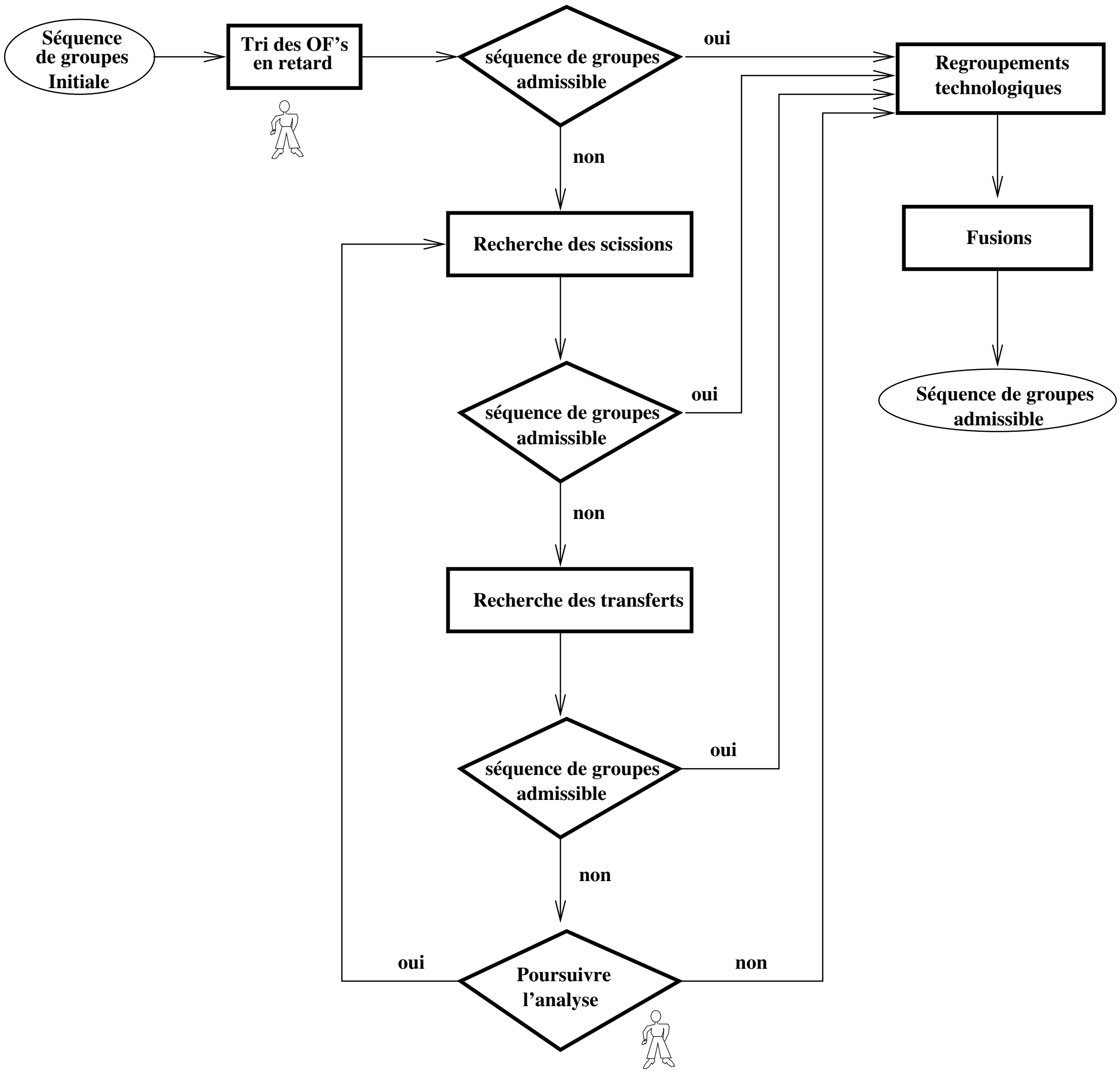


FIG. V.4 – *Algorithme du module ANALYSE*

V.4 Un système interactif d'aide à la décision : le module ARBITRE

Le module ARBITRE est utilisé comme base d'un système d'aide à la décision pour l'ordonnancement en temps réel, pour exploiter la séquence de groupes admissible issue du module ANALYSE. ARBITRE effectue la mise à jour du graphe potentiels-tâches $\mathcal{G}_G$ représentant la séquence de groupes en fonction des événements survenant dans l'atelier et des décisions prises par les responsables de ressources. En réponse à un événement, ARBITRE établit un diagnostic d'admissibilité et détermine un ensemble de décisions possibles et/ou souhaitables ce qui constitue un ensemble de *propositions*. En réponse à une décision envisagée, ARBITRE détermine l'impact sur la tenue des délais, et fournit un ensemble d'*informations* pertinentes sur ses conséquences. Ces propositions et ces informations sont des éléments destinés à aider le décideur dans l'élaboration de sa décision (voir paragraphe I.2.2). Pour organiser cette aide, l'ensemble des décisions et événements significatifs pour la conduite en temps réel de l'atelier doit être tout d'abord identifié (paragraphes V.4.1 et V.4.2). Un événement est caractérisé par la modification associée de l'état de l'atelier, défini par l'état des entités qui le composent : les ressources principales, les lignes de charge des ressources complémentaires et les opérations (d'exécution et de préparation). L'enchaînement de ces états en relation avec les événements et les décisions a été décrit dans [Billaut, 1993] et [Billaut *et al.*, 1997] dans un contexte multi-ressources sans préparation. Pour prendre en compte cette préparation, de nouveaux états, événements et décisions liés aux opérations de préparation sont nécessaires. Chaque entité (ressource principale, ligne de charge d'une ressource complémentaire, opération d'exécution et de préparation) peut être suivie individuellement par le décideur grâce à un ensemble d'états associés. Ces états sont décrits dans le paragraphe V.4.3.

L'enchaînement des états des différentes entités est représenté par un réseau de Petri de haut niveau dans lequel les jetons (colorés) représentent les entités, les places représentent les états, et les transitions représentent les événements et les décisions. Ce réseau est décrit dans le paragraphe V.4.4. L'aide à la décision proposée au paragraphe V.4.5 porte sur les décisions associées aux événements attendus, les décisions associées aux événements inattendus (par exemple débuts de pannes de ressources), les blocages d'opérations mais aussi les modifications du plan de production (insertion d'ordres de fabrication imprévus). Ce paragraphe est partiellement basé sur [Artigues et Roubellat, 1997e].

V.4.1 Types de décision

- Décision de type **E** pour début d'Exécution. Cette décision consiste à coupler une opération d'exécution et les ressources principales et complémentaires nécessaires afin d'engager l'opération. Suite à cette décision, l'opération est ôtée de son groupe et placée dans un nouveau groupe en tête de ces ressources. On fait par la suite l'hypothèse que le chevauchement n'est pas autorisé. Dans le cas où les ressources principales ne sont pas préparées pour exécuter cette opération (associées et dans le type voulu), le système spécifie pour le décideur l'activité de préparation nécessaire préalablement à l'engagement, qui ne pourra être confirmé qu'au terme de cette préparation.
- Décision de type **M** pour début de Montage. Cette décision consiste à engager le montage d'un ensemble de ressources principales isolées au moyen d'un ensemble spécifié de ressources complémentaires.
- Décision de type **D** pour début de Démontage. Cette décision consiste à engager le démontage d'un ensemble de ressources principales au moyen d'un ensemble spécifié de ressources complémentaires.
- Décision de type **CTI** pour début de Changement de Type sur une ressource principale Isolée. Cette décision consiste à engager le changement de type d'une ressource principale isolée au moyen d'un ensemble spécifié de ressources complémentaires.
- Décision de type **CTA**. pour début de Changement de Type sur des ressources principales Associées. Cette décision consiste à engager le changement de type d'un ensemble de ressources principales associées au moyen d'un ensemble spécifié de ressources complémentaires.

- Décision de type **I** pour Interruption. On distingue plusieurs cas d'interruption. La décision de type **IE** pour Interruption d'une opération d'Exécution consiste à interrompre l'exécution de cette opération avant qu'elle ne soit complètement réalisée. Elle découple cette opération de toutes les ressources qu'elle utilise et libère ainsi les ressources utilisées par l'opération.
  La décision de type **IM** pour Interruption de Montage, consiste à interrompre le montage avant qu'il ne soit complètement réalisé, ce qui libère les ressources complémentaires, mais pas les ressources principales qui ne sont pas utilisables pour l'exécution (ni isolées ni associées). Cette décision est obligatoirement suivie soit d'une reprise de montage (voir RM) pour terminer le montage, soit d'un démontage de l'ensemble des ressources principales partiellement associées.
  La décision de type **ICT** pour Interruption de Changement de Type soit sur une ressource principale isolée (**ICTI**) soit sur des ressources principales associées (**ICTA**). Cette décision consiste à interrompre le changement de type sur la ressource principale isolée (ou les ressources principales associées), ce qui libère les ressources complémentaires, mais pas la (les) ressource(s) principale(s) qui est (sont) dans un état ne correspondant au type d'aucune opération d'exécution. On fait l'hypothèse que cette décision est obligatoirement suivie d'une reprise de changement de type (voir RCT).
  La décision de type **ID** pour Interruption de Démontage consiste à interrompre le démontage avant son terme, ce qui libère les ressources complémentaires, mais pas les ressources principales qui sont dans un état intermédiaire. Cette décision est obligatoirement suivie d'une reprise de démontage (voir RD) ou d'un remontage de l'ensemble des ressources principales concernées.
- Décision de type **R** pour reprise. On distingue plusieurs cas de reprise.
  La décision de type **RE** pour Reprise d'Exécution consiste à reprendre une opération d'exécution précédemment interrompue pour la terminer et ainsi à la coupler de nouveau à un ensemble de ressources principales et de lignes de charge de ressources complémentaires.
  La décision de type **RM** pour Reprise de Montage consiste à reprendre la réalisation du montage après son interruption, avec éventuellement un nouvel ensemble spécifié de ressources complémentaires.
  La décision de type **RD** pour Reprise de Démontage consiste à reprendre la réalisation du démontage après son interruption, avec éventuellement un nouvel ensemble spécifié de ressources complémentaires.
  La décisions de type **RCT** pour Reprise de Changement de Type. Cette décision consiste à reprendre le changement de type sur une ressources principale isolée (**RCTI**) ou sur des ressources principales associées (**RCTA**), avec éventuellement un nouvel ensemble de ressources complémentaires et en changeant éventuellement le type prévu lors du début de changement de type. Ceci permet d'annuler ou de modifier une changement de type avant la fin de sa réalisation.
- Décision de type **C** pour Changement d'affectation. Cette décision a pour but de changer l'affectation d'une opération d'exécution sur les ressources principales ou complémentaires ou bien d'une opération de préparation sur les ressources complé-

mentaires uniquement. Ce changement d'affectation ne peut pas être effectué sur une opération d'exécution couplée à des ressources, ni sur une opération de préparation en train d'être réalisée. Dans ce cas, une décision préalable d'interruption est nécessaire.

Les décisions E, M, D, CTA, CTI, RE, RM, RD, RCTA et RCTI sont appelées des *décisions d'engagement.*

V.4.2 Evénements pertinents pour la prise de décision

On a coutume de distinguer les événements attendus dont on sait qu'ils doivent se produire suite à un certain état de l'atelier, des événements inattendus correspondant à un aléa.

V.4.2.a) Evénements inattendus

On considère les événements inattendus (aléas) suivants:

- **PC** pour panne d'une ligne de charge complémentaire. Si la panne intervient pendant l'exécution, l'opération est arrêtée. Toute décision d'engagement mettant en jeu cette ligne de charge est provisoirement impossible. On associe à cet événement une durée de panne estimée, quand cette information est disponible.
- **PP** pour panne d'une ressource principale. Si la panne intervient pendant l'exécution, l'opération est arrêtée. Toute décision de début d'exécution (E) mettant en jeu une ressource principale ayant subi cet événement est provisoirement impossible. On distingue deux cas selon que la panne implique ou non le démontage obligatoire d'une association contenant la ressource en panne :
  PP1 Cette panne de premier type peut être réparée si la ressource concernée est isolée, fait partie d'une association ou est en cours de montage. Par exemple, une panne de machine comme un manque de lubrifiant peut éventuellement être réparée sans démonter les outils montés ou en cours de montage sur cette machine. Si cette panne intervient pendant un montage, on peut décider de mener ce montage à son terme. **PP2** Cette panne de second type ne peut être réparée que si la ressource est dans l'état isolée. Elle entraine obligatoirement un démontage si la ressource fait partie d'une association ou est en cours de montage. Par exemple, le bris d'un outil ne peut être réparé qu'en démontant l'outil de la machine. Dans tous les cas, la panne n'empêche ni le démontage, ni le changement de type.
- **B** pour blocage d'une opération, c'est-à-dire l'arrêt de l'opération pour une autre raison que la panne d'une ressource. On associe à cet événement une durée de blocage estimée, quand cette information est disponible. On peut bloquer une opération d'exécution (**BE**) ou une des 4 opérations de préparation (**BM**), (**BD**), (**BCTI**), (**BCTA**).

V.4.2.b) Evénements attendus

Les différents événements attendus liés aux décisions ou aux événements inattendus sont les suivants:

- **FE** pour fin d'exécution d'une opération d'exécution. L'occurence de l'événement est reportée si une des ressources principales ou complémentaires tombe en panne, ou si l'opération a subi un événement BE. Dans ce cas FE ne pourra arriver qu'après la fin de l'aléa.
- **FB** pour fin de blocage d'une opération. On distingue la fin de blocage d'une opération d'exécution (**FBE**) ou d'une opération de préparation (**FBM**, **FBD**, **FBCTI**, **FBCTA**).
- **FP** pour fin de panne. On distingue la fin de panne d'une ressource principale (**FPP1**, **FPP2**) ou d'une ligne de charge complémentaire (**FPC**). A la différence de FPP1, FPP2 ne peut arriver que si la ressource principale en panne est isolée.
- **FM** pour fin de montage d'une association. L'occurence de cet événement est provisoirement impossible si le montage subit un événement BM, si une ligne de charge d'une ressource complémentaire utilisée subit un événement PC ou si une ressource principale à monter subit un événement PP2.
- **FD** pour fin de démontage d'une association. L'occurence de cet événement est provisoirement impossible si le démontage subit un événement BD ou si une ligne de charge d'une ressource complémentaire utilisée subit un événement PC.
- **FCTI** pour fin de changement de type sur une ressource principale isolée. L'occurence de cet événement est provisoirement impossible si le changement de type subit un événement BCTI ou si une ligne de charge d'une ressource complémentaire utilisée subit un événement PC.
- **FCTA** pour fin de changement de type sur une association de ressources principales. L'occurence de cet événement est provisoirement impossible si le changement de type subit un événement BCTA ou si une ligne de charge d'une ressource complémentaire utilisée subit un événement PC.

V.4.3 Description de l'état de l'atelier

Pour proposer une aide à la décision, il est nécessaire de connaître précisément l'état de l'atelier, en relation bien sûr avec les événements et décisions décrits ci-dessus. Cet état est caractérisé par l'état des ressources principales, des lignes de charge des ressources complémentaires, et des opérations des ordres de fabrication.

De plus, le système doit en permanence donner au(x) décideur(s) et aux opérateurs des informations pertinentes pour le suivi de l'atelier. L'aide à la décision est concue de façon à ce que plusieurs décideurs ou opérateurs utilisent simultanément le système, chacun ayant besoin d'une vison différente de l'atelier, selon sa fonction : vision d'une ressource ou d'un ensemble de ressources, vision d'un ou plusieurs ordres de fabrication, vision des opérations de préparation. Pour chacune de ces vues, il s'agit de donner des informations sur la ou les entité(s) concernée(s) mais aussi des informations agrégées sur son (leur) environnement.

V.4.3.a) Etats des ressources principales

A chaque instant, une ressource principale peut se trouver dans un état représenté par un 5-uplet $\{Ep1, Ep2, Ep3, Ep4, Ep5\}$. Lorsqu'il suit la ressource principale k, le décideur est informé en permanence des valeurs de ces cinq champs.

Le champ $Ep1$ décrit l'état de la ressource lié au type d'opérations d'exécution qu'elle peut exécuter :

- **de type X**. La ressource est préparée pour exécuter des opérations d'exécution du type $X \in \mathcal{T}$.
- **Changement de type en cours**(CTEC). La ressource subit une opération de changement de type
- **En attente de changement de type**(ACT). Le changement de type de la ressource a été interrompu et la reprise du changement de type est nécessaire.

Le champ $Ep2$ décrit l'état de la ressource lié à l'association :

- **Isolée**(ISOL). La ressource n'est associée à aucune autre ressource principale.
- **Associee**(ASSOC). La ressource est associée à un ensemble d'autres ressources principales.
- **Montage En Cours** (MEC). La ressource subit une opération de montage,
- **Démontage En Cours** (DEC).La ressource subit une opération de démontage,
- **En attente de démontage ou de montage**(ADM). Le montage ou le démontage a été interrompu et une reprise de montage ou de démontage est nécessaire.

Le champ $Ep3$ décrit l'état de la ressource lié à l'exécution :

- **Libre** (LI). La ressource n'est couplée à aucune opération d'exécution
- **Active** (AC). Une opération d'exécution est couplée à la ressource.

Le champ $Ep4$ décrit l'état de la ressource lié aux pannes :

- **En panne de type 1** (PN1). Une panne de premier type s'est produite sur la ressource.
- **En panne de type 2** (PN2). Une panne de deuxième type s'est produite sur la ressource
- **Non en panne** ($\overline{\mathrm{PN}}$). La ressource n'est pas en panne.

Le champ $Ep5$ décrit l'impossibilité d'utiliser la ressource, bien que celle-ci ne soit pas en panne, ce qu'on traduit par une anomalie externe à la ressource et qui donne ainsi au décideur une information agrégée sur l'environnement de chaque ressource :

- **anomalie externe** (AE)
- **non anomalie externe** ($\overline{\mathrm{AE}}$)

Les différentes valeurs des champs ne sont pas toutes compatibles entre elles. Une ressource principale ne peut être couplée à une opération d'exécution que si elle est dans les états associée ou isolée. La valeur AC du champ $Ep2$ est ainsi uniquement compatible avec la valeur ISOL ou ASSOC du champ $Ep1$. Le réseau de Petri décrit au paragraphe V.4.4 permettra d'identifier toutes les possibilités.

Une anomalie externe n'est possible qu'en état d'association ou de couplage. L'impossibilité d'utiliser la ressource, bien que celle-ci ne soit pas en panne, peut arriver dans les cas suivants :

- La ressource est couplée à une opération d'exécution et cette opération est bloquée.
- La ressource est couplée à une opération d'exécution et une autre ressource, principale ou complémentaire couplée à l'opération est en panne.
- La ressource n'est pas couplée, mais elle est associée avec une autre ressource principale en panne.
- La ressource est en cours de préparation et une ressource complémentaire est en panne.
- La ressource est en cours de préparation et cette opération de préparation est bloquée.

V.4.3.b) Etats d'une ligne de charge d'une ressource complémentaire

Les états décrits dans ce paragraphe concernent aussi le cas où aucune ressource utilisée par une opération d'exécution ne nécessite de préparation [Billaut *et al.*, 1997]. Une ressource disjonctive est alors équivalente à une ligne de charge.

La ligne de charge q d'une ressource complémentaire c peut se trouver dans un état représenté par un 3-uplet $\{Ec1, Ec2, Ec3\}$. Lorsqu'il suit la ligne de charge q, le décideur est informé en permanence des valeurs de ces trois champs.

Le champ $Ec1$ décrit l'état de la ligne de charge lié à l'exécution :

- **Libre** (LI). La ligne de charge n'est couplée à aucune opération d'exécution et n'exécute aucune opération de préparation.
- **Active** (AC). Une opération d'exécution ou de préparation est en cours sur la ligne de charge

Le champ $Ec2$ décrit l'état de la ligne de charge lié aux pannes :

- **En panne** (PN). La ligne de charge est en panne.
- **Non en panne** ($\overline{\text{PN}}$). La ligne de charge n'est pas en panne.

Le champ $Ec3$ décrit l'impossibilité d'utiliser la ligne de charge, bien que celle-ci ne soit pas en panne, ce qu'on traduit par une anomalie externe à la ressource et qui donne au décideur une information agrégée sur l'état de l'environnement de la ligne de charge :

- **Anomalie externe** (AE)

- **Non anomalie externe** ($\overline{\text{AE}}$)

La valeur AE du champ $Ec3$ est possible uniquement si le champ $Ec1$ vaut AC. L'anomalie externe peut arriver dans tous les cas d'utilisation d'une ressource complémentaire pour de l'exécution ou de la préparation.

Dans le cas de ressources cumulatives, le système peut proposer une information agrégée de ces champs : il peut fournir ainsi le *nombre* de lignes de charge libres, le *nombre* de lignes de charge actives, le *nombre* de lignes de charge en panne, le *nombre* de lignes de charge en aléa externe.

V.4.3.c) Etats d'une opération d'exécution

L'opération d'exécution (i, j) peut se trouver dans un état représenté par un 3-uplet $\{Eo1, Eo2, Eo3\}$. Lorsqu'il suit un ordre de fabrication i, le décideur est informé en permanence des valeurs de ces trois champs pour chaque opération de i.

Le champ $Eo1$ concerne l'état de l'opération par rapport à l'état de ses précédentes dans la gamme et à son exécution :

- **blanc**(BL). Il existe une opération précédente de (i, j) dans la gamme de l'ordre de fabrication i qui n'est pas commencée.
- **en approche** (AP). Toutes les opérations précédentes de (i, j) dans la gamme de l'ordre de fabrication i sont commencées.
- **disponible** (DI). Toutes les opérations précédentes de (i, j) dans l'ordre de fabrication i sont terminées. (i, j) se trouve devant son lieu d'exécution.
- **exécution en cours** (EEC). L'opération est couplée et en cours d'exécution sur l'ensemble des ressources principales et complémentaires utilisées.
- **terminée** (TR). L'exécution de l'opération est terminée.

Le champ $Eo2$ concerne l'état de l'opération par rapport aux ressources auxquelles elle est couplée :

- **arrêtée** (EAR). L'opération est arrêtée suite à une panne d'une ou plusieurs lignes de charge ou ressources utilisées.
- **non arrêtée** ($\overline{\text{EAR}}$). L'opération n'est pas arrêtée.

La valeur EAR du champ $Ep2$ n'est possible que si le champ $Eo1$ vaut EEC.

Le champ $Eo3$ concerne l'état de l'opération lié à l'aléa de blocage :

- **bloquée** (EBQ). L'opération ne peut plus être exécutée suite à un incident autre qu'une panne de ressource, par exemple un problème lié à l'ordre de fabrication.
- **non bloquée** ($\overline{\text{EBQ}}$). L'opération n'est pas bloquée.

V.4.3.d) Etats d'une opération de préparation

Le décideur peut consulter à tout moment l'état associé à une opération de préparation (prévue ou pas dans la séquence) d'une ressource isolée ou d'une ressource associée qui est représenté par trois champs $\{Es1, Es2, Es3\}$.

Le champ $Es1$ concerne l'état de l'opération par rapport à sa réalisation :

- **blanc** (MBL, DBL, CTIBL, CTABL). L'état nécessaire à l'engagement de l'opération de préparation n'est pas atteint et n'est pas en passe d'être atteint. C'est le cas par exemple d'une opération de montage de ressources principales dont les états correspondent à des types d'opérations d'exécution différents.
- **en approche** (MAP, DAP, CTIAP, CTAAP). L'état nécessaire à l'engagement de l'opération de préparation est en passe d'être atteint. Pour une opération de démontage d'un ensemble de ressources principales, c'est le cas par exemple lorsque une opération de montage est en cours sur les mêmes ressources principales.
- **disponible** (MDI, DDI, CTIDI, CTADI). Les ressources principales sont dans l'état nécessaire à l'engagement de l'opération de préparation. Par exemple, les ressources principales sont toutes isolées avant un montage.
- **en cours** (MEC, DEC, CTIEC, CTAEC). L'opération est en train d'être réalisée.
- **Terminée** (MTR, DTR, CTITR, CTATR). Les ressources principales sont dans l'état consécutif à la fin de l'opération de préparation.

Le champs $Es1$ pour une opération de préparation possède les mêmes valeurs que le champs $Eo1$ pour une opération d'exécution. Toutefois, dans le cas de l'opération d'exécution ces valeurs spécifient une relation avec la gamme qui sont structurelles et statiques. Dans le cas de l'opération de préparation ces valeurs spécifient une relation qui est dynamique. Par exemple, l'état "Terminée" pour une opération de préparation n'est pas définitif - contrairement au même état d'une opération d'exécution - puisqu'il dépend de l'état des ressources à préparer.

Le champ $Es2$ concerne l'état de l'opération par rapport aux ressources mises en jeu :

- **arrêtée** (MAR, DAR, CTIAR, CTAAR). L'opération est arrêtée suite à une panne d'une ou plusieurs lignes de charge ou ressources mises en jeu.
- **non arrêtée** ($\overline{\text{MAR}}$, $\overline{\text{DAR}}$, $\overline{\text{CTIAR}}$, $\overline{\text{CTAAR}}$). L'opération n'est pas arrêtée.

Les valeurs MAR, DAR, CTIAR et CTAAR de ce champ sont possibles uniquement si l'opération est dans l'état en cours.

Le champ $Es3$ concerne l'état de l'opération associé à l'aléa de blocage :

- **bloquée** (MBQ, DBQ, CTIBQ, CTABQ). L'opération de préparation ne peut plus être effectuée suite à un incident autre qu'une panne de ressource. Par exemple, lors de l'enchaînement de couleurs, l'impossibilité d'effectuer la préparation du $ROUGE$ vers le $VERT$ à cause du manque d'un solvant particulier entraîne le blocage de toutes les opérations de changement prévues du type $ROUGE$ vers le type $VERT$.
- **non bloquée** ($\overline{\text{MBQ}}$, $\overline{\text{DBQ}}$, $\overline{\text{CTIBQ}}$, $\overline{\text{CTABQ}}$). L'opération de préparation n'est pas bloquée.

V.4.4 Représentation des enchaînements d'états possibles par un réseau de Petri

La représentation de l'état de chacune des entités par un réseau de Petri coloré (voir [Valette, 1995]) décrit dans les figures V.5, V.7, V.8 et V.6, permet au système d'aide à la décision :

- à partir d'un état donné, d'identifier l'ensemble des événements possibles liés à une ou plusieurs entités,
- à partir d'un état donné, d'identifier l'ensemble des décisions possibles concernant une ou plusieurs entités,
- de représenter le passage d'un état à un autre en fonction des événements et des décisions,
- de représenter l'état global de l'atelier sans énumérer l'ensemble des situations possibles,
- de donner à chaque instant une valeur aux différents champs de l'état de chaque entité.

On considère dans ce paragraphe uniquement le cas où chaque opération d'exécution doit être exécutée par une association de ressources principales et non par une ressource principale isolée.

V.4.4.a) Interprétation des éléments du réseau

Dans le réseau de Petri coloré considéré, un **jeton simple** représente une entité de l'atelier. On distingue :

- les jetons de type $< R_k >$ représentant chaque ressource principale $k \in \mathcal{R}b$,
- les jetons de type $< ldc_q >$ représentant chaque ligne de charge $q = 1, ..., E_c$ de chaque ressource complémentaire $c \in \mathcal{R}c$,
- les jetons de type $< (i,j) >$ représentant chaque opération j de chaque ordre de fabrication i.

Les **jetons composés** représentent les cas où différentes entités de l'atelier doivent être considérées conjointement. On distingue :

- les jetons de type $< R_1, ..., R_k, ... >$ représentant une association d'un ensemble de ressources principales,
- les jetons de type $< R_1, ..., R_k, ..., ldc_1, ..., ldc_q, ... >$ représentant l'utilisation d'un ensemble de lignes de charge de ressources complémentaires pour effectuer un montage ou un démontage sur l'association $< R_1, ..., R_k, ... >$,
- les jetons de type $< R_k, X >$ représentant une ressource principale préparée pour le type X,

- les jetons de type $< R_k, X, Y, ldc_1, ..., ldc_q, ... >$ représentant l'utilisation d'un ensemble de lignes de charge de ressources complémentaires pour effectuer un changement du type X vers le type Y sur la ressource isolée R_k,
- les jetons de type $< R_1, ..., R_k, X, Y, ..., ldc_1, ..., ldc_q, ... >$ représentant l'utilisation d'un ensemble de lignes de charge de ressources complémentaires pour effectuer un changement du type X vers le type Y sur l'ensemble de ressources associées R_1, $..., R_k, ...$,
- les jetons de type $< (i, j), R_1, ..., R_k, ..., ldc_1, ..., ldc_q, ... >$ représentant le couplage de l'opération d'exécution (i, j) avec un ensemble de ressources principales et de lignes de charge de ressources complémentaires.

A chaque décision (voir V.4.1), à chaque événement attendu ou inattendu (voir V.4.2) correspond une unique **transition** du réseau.

La présence d'un jeton, correspondant à un ensemble d'entités, dans une **place** du réseau, fixe la valeur d'un ou plusieurs champs de l'état de chaque entité. Dans les figures V.5, V.7, V.8 et V.6, si une place contient des jetons simples, son nom est soit le nom du champ unique concerné, soit une composition du nom des différents champs affectés. Si une place contient des jetons composés, on donne à la place un nom pour chaque ensemble de champs affectés de chaque entité concernée par le jeton composé.

V.4.4.b) Décomposition en sous-réseaux

Pour éviter l'explosion combinatoire des états dûe aux différentes combinaisons possibles des divers champs de chaque entité, le réseau a été décomposé en huit sous réseaux. Dans chaque sous-réseau, on trouve au maximum un seul exemplaire des jetons représentant la même entité.

Sur chaque arc du réseau est indiqué le type des jetons transitant par cet arc. Le type des jetons autorisé dans chaque place est ainsi déterminé. A une transition T d'un sous-réseau donné, peut être associé un ensemble de conditions (regroupées par une accolade). Chaque condition (du type $P < J >$) est un test de présence du jeton $< J >$ dans la place P d'un autre sous réseau. Cette présence est nécessaire pour tirer la transition T dans le sous-réseau considéré. Chaque condition pourrait être en fait représentée par deux arcs portant tous deux le jeton $< J >$, l'un dirigé de la transition T vers la place P, l'autre dirigé de la place P vers la transition T.

Pour faciliter la lecture, les 7 premiers sous-réseaux sont tous représentés dans la figure V.5 alors que le dernier sous-réseau est représenté dans trois figures différentes (V.7, V.8 et V.6) :

Changements d'état dûs aux événements inattendus et aux événements attendus consécutifs Pour chaque entité de l'atelier, un sous-réseau définit les changements d'états de l'entité en fonction des événements inattendus (aléas) et des événements marquant la fin de l'aléa. Ces réseaux sont regroupés dans la figure V.5. On décrit les sous réseaux de gauche à droite et de haut en bas. Le premier réseau concerne les enchaînements des deux types de panne sur chaque ressource principale R_k. A partir de l'état non

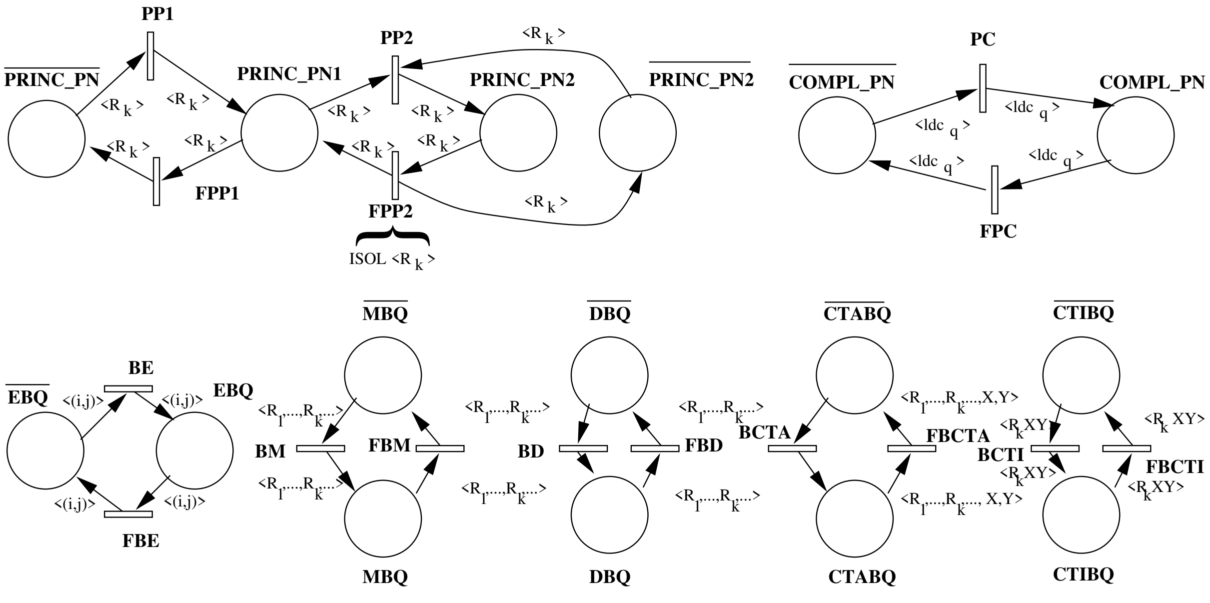


FIG. V.5 – *Changements d'états des entités suite aux événements inattendus et aux événements attendus consécutifs*

en panne d'une ressource principale $Ep4 = \overline{\mathrm{PN}}$ on peut atteindre l'état panne de type 1 $Ep4 =$ PN1, puis l'état $Ep4 =$ PN2 par les événement successifs PP1 et PP2. L'occurence des événements fin de panne provoque le retour à un état de panne de type inférieur. La particularité de la panne de type 2 est qu'elle ne peut être réparée que si la ressource principale est dans l'état isolée. Cela se traduit par la condition ISOL$< R_k >$ sur la transition FPP2 qui impose la présence du jeton $< R_k >$ dans la place PRINC_LI_ISOL de la figure V.7. Son occurence provoque donc obligatoirement un démontage, ce qui n'est pas le cas de l'autre type de panne. Dans le cas où une ressource tombe directement en panne de type 2, sans passer par la panne de type 1, les événements $PP1$ et $PP2$ sont enchaînés instantanément.

Le deuxième réseau concerne la panne sur les lignes de charge des ressources complémentaires. En cas de panne (événement PC), la ligne de charge ldc_q passe dans l'état en panne ($Ec2 =$PN) jusqu'à ce que l'événement attendu FPC la refasse passer à l'état non en panne ($Ec3 = \overline{\mathrm{PN}}$). Les réseaux suivants concernent les événements et les états liés au blocage des opérations d'exécution (champs $Eo2$) et de préparation (champs $Es2$). En cas de blocage (événement BQ), l'opération de préparation ou d'exécution reste dans l'état bloqué jusqu'à ce que l'événement attendu FBQ la refasse passer à l'état non bloquée. Hormis pour la fin de panne de type 2 d'une ressource principale (événement FPP2) qui ne peut arriver que si la ressource principale est dans l'état isolé, chaque événement inattendu (puis chaque événement attendu consécutif) peut arriver à n'importe quel moment, ce qui se traduit sur le réseau par l'absence d'arcs de la transition concernée avec d'autres places du réseau global.

Changements d'état des entités mises en jeu pour l'exécution Dans la figure V.6, sont représentées les conditions précises de validité des décisions de début d'exécution E, d'interruption d'exécution IE et d'occurence de l'événement attendu de fin d'exécution FE. L'engagement de l'opération d'exécution fait passer l'opération de l'état disponible (DI) à l'état en cours (EEC), l'ensemble des lignes de charge des ressources complémentaires requises de l'état libre ($Ec1$ =LI) à l'état actif ($Ec1$ =AC) et l'ensemble des ressources principales requises de l'état libre ($Ep3$ =LI) à l'état actif ($Ep3$ =AC). De plus, si (i, q) est une opération suivante de (i, j) dans la gamme g_i, (i, q) passe de l'état blanc ($Eo1$ =BL) à l'état en approche ($Eo1$ =AP). Le cas représenté ici est le cas d'une gamme linéaire car dans le cas contraire, l'opération ne passe en approche que si toutes ses précédentes sont commencées. Pour pouvoir engager l'opération, les ressources principales mises en jeu doivent être dans l'état associé ($Ep2$ =ASSOC). Les conditions données sur la transition E indiquent de plus qu'aucune ressource principale ou ligne de charge d'une ressource complémentaire ne doit être en panne ($Ec2 = \overline{\mathrm{PN}}$ et $Ep4 = \overline{\mathrm{PN}}$), que les ressources principales doivent être dans l'état correspondant au type de (i, j) ($Ep1 = RT_{ij}$), et que l'opération (i, j) ne doit pas être bloquée ($\overline{\mathrm{BQ}}$). La fin d'exécution (FE) libère les ressources principales et les lignes de charge des ressources complémentaires (retour aux états libres) et fait passer l'opération de l'état en cours $Eo1$ =EEC à l'état terminé $Eo1$ =TR. L'opération (i, q) passe de l'état en approche ($Eo1$ =AP) à l'état disponible ($Eo1$ =DI). Si une des lignes de charge des ressources complémentaires ou une des ressources principales est en panne, ou si (i, j) est bloquée, l'occurrence de l'événement de fin d'exécution est reportée au moins jusqu'à la fin de l'aléa (mêmes conditions que pour la décision de début d'exécution). Dans ce cas, (i, j) et les ressources requises sont couplées mais l'exécution est arrêtée ($Eo2$ =EAR). On peut libérer tout de même les ressources principales et les lignes de charge des ressources complémentaires par la décision d'interruption IE qui provoque en outre pour (i, j) le retour à l'état disponible, et pour l'opération précédente de (i, j) le retour à l'état blanc. Suite à l'interruption, et si (i, j) n'est pas bloquée, on peut la réengager sur un ensemble de ressources différent (décision RE).

Changements d'état des entités mises en jeu pour le montage et le démontage
Dans la figure V.7 sont représentées les conditions précises de validité des décisions de montage (M), de démontage (D), d'interruption (de montage IM et de démontage ID), de reprise (de montage RM et de démontage RD) et d'occurence des événements attendus de fin de montage (FM) et de fin de démontage (FD). Les places intitulées PRINC_ASSOC_LI et COMPL_LI sont les mêmes que dans la figure précédente.

La décision de montage (M) fait passer un ensemble de ressources principales à monter de l'état isolé ($Ep2$ =ISOL) à l'état montage en cours ($Ep2$ =MEC) et un ensemble de lignes de charge de ressources complémentaires requises pour le montage de l'état libre ($Ec1$ =LI) à l'état actif ($Ec1$ =AC). Pour engager un montage, chaque ressource principale doit être dans l'état libre ($Ep3$ =LI) aucune des lignes de charge des ressources complémentaires requises ne doit être en panne, l'opération de montage ne doit pas être bloquée et aucune des ressources principales concernées ne doit être en panne de type 2. L'événement fin de montage (FM) fait passer l'ensemble des ressources principales de l'état de montage en cours ($Ep2$ =MEC) à l'état associé ($Ep2$ =ASSOC) et libère les lignes de charge des ressources complémentaires. Un aléa peut se produire au cours du montage.

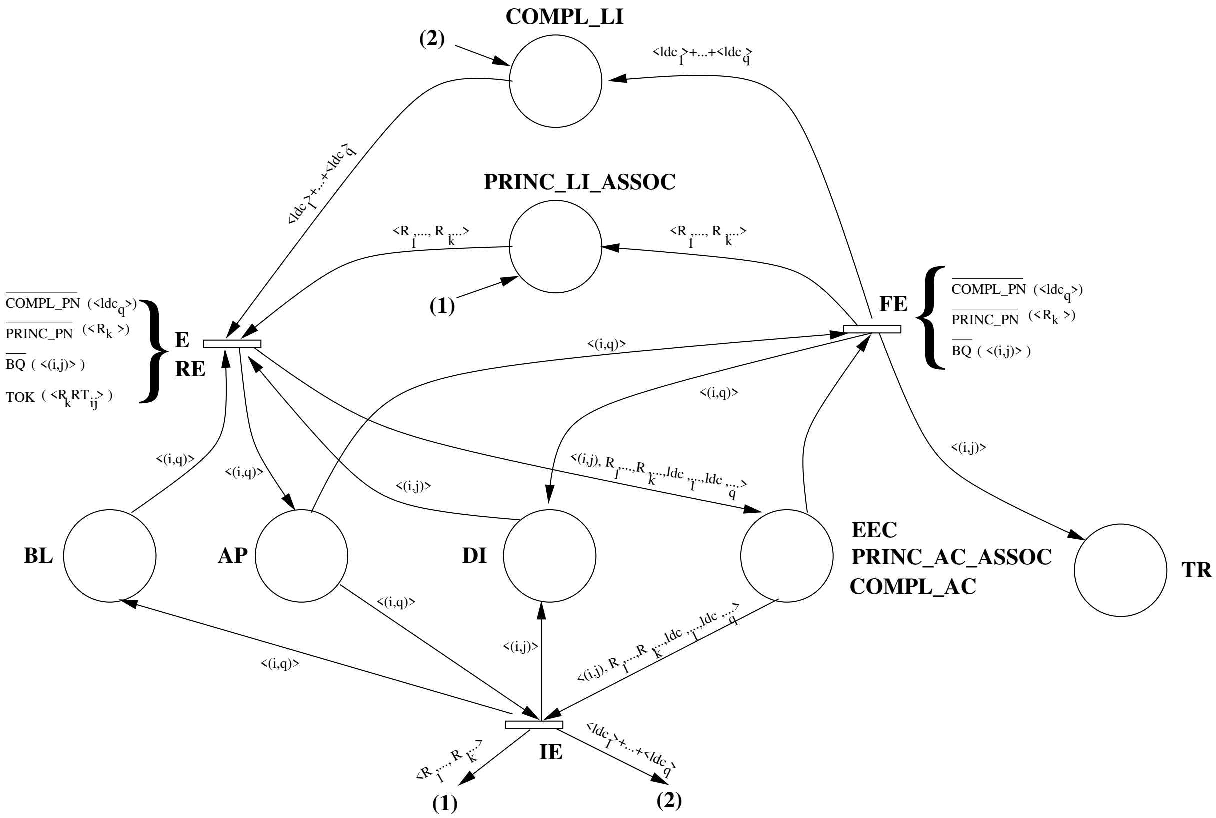


FIG. V.6 – *Changements d'état des entités mises en jeu pour l'exécution*

La transition FM étant associée aux mêmes conditions que la transition M, le montage ne peut être achevé si une des lignes de charge requises sur les ressources complémentaires est en panne, si l'opération de montage est bloquée et si une des ressources principales est en panne de type 2. Par contre le montage peut être achevé si une ressource principale est en panne de type 1.

Le démontage est un processus symétrique au montage. La décision de démontage (D) fait passer un ensemble de ressources principales à démonter de l'état associé ($Ep2$ =ASSOC) à l'état démontage en cours ($Ep2$ =MEC) et un ensemble de lignes de charge de ressources complémentaires requises pour le démontage de l'état libre ($Ec1$ =LI) à l'état actif ($Ec1$ =CAC). Pour engager un démontage, chaque ressource principale doit être dans l'état libre ($Ep3$ =LI). Aucune des lignes de charge requises sur les ressources complémentaires ne doit être en panne et l'opération de démontage ne doit pas être bloquée. A la différence du montage, le démontage peut être engagé même s'il existe dans l'association à démonter des ressources principales en panne de type 2. L'événement fin de démontage (FD) fait passer l'ensemble des ressources principales de l'état de démontage en cours ($Ep2$ =DEC) à l'état isolé ($Ep2$ =ISOL) et libère les lignes de charge des ressources complémentaires utilisées.

La décision d'interruption de montage (IM) fait passer les ressources principales de l'état de montage en cours à l'état d'attente de montage ou de démontage et libère les lignes de charge requises par le montage sur les ressources complémentaires. Cette décision peut être prise à tout moment, et notamment suite à un aléa. Dans l'état d'attente de montage ou de démontage ($Ep2$ =AMD) les ressources principales sont inutilisables pour l'éxécution. On peut alors envisager l'engagement d'une opération de démontage (décision RD), ce qui fait passer les ressources principales à l'état de démontage en cours, sous les mêmes conditions que la décision D. On distingue deux cas d'annulation de montage :

- Si l'aléa qui a provoqué l'interruption est une panne de type 2 alors l'annulation du montage est la seule décision possible car la panne ne pourra être réparée que dans l'état isolé.
- Sinon, l'annulation est volontaire et elle traduit la volonté de libérer les ressources principales suite à un aléa (blocage du montage, panne des ressources complémentaires ou panne de type 1 des ressources principales).

Si l'aléa qui a provoqué l'interruption n'est pas une panne de type 2, on peut envisager la reprise de montage (décision RM) qui fait passer les ressources principales à l'état de montage en cours, sous les mêmes conditions que la décision M. On envisage deux cas de reprise :

- L'interruption avait pour but de libérer les ressources complémentaires suite à un blocage du montage et ce blocage est terminé.
- L'interruption a eu lieu suite à une panne d'une ligne de charge complémentaire, la reprise a alors pour but d'utiliser d'autres lignes de charge non en panne.

La décision d'interruption de démontage (ID) fait passer les ressources principales de l'état de démontage en cours à l'état $Ep2$ =AMD et libère les lignes de charge requises

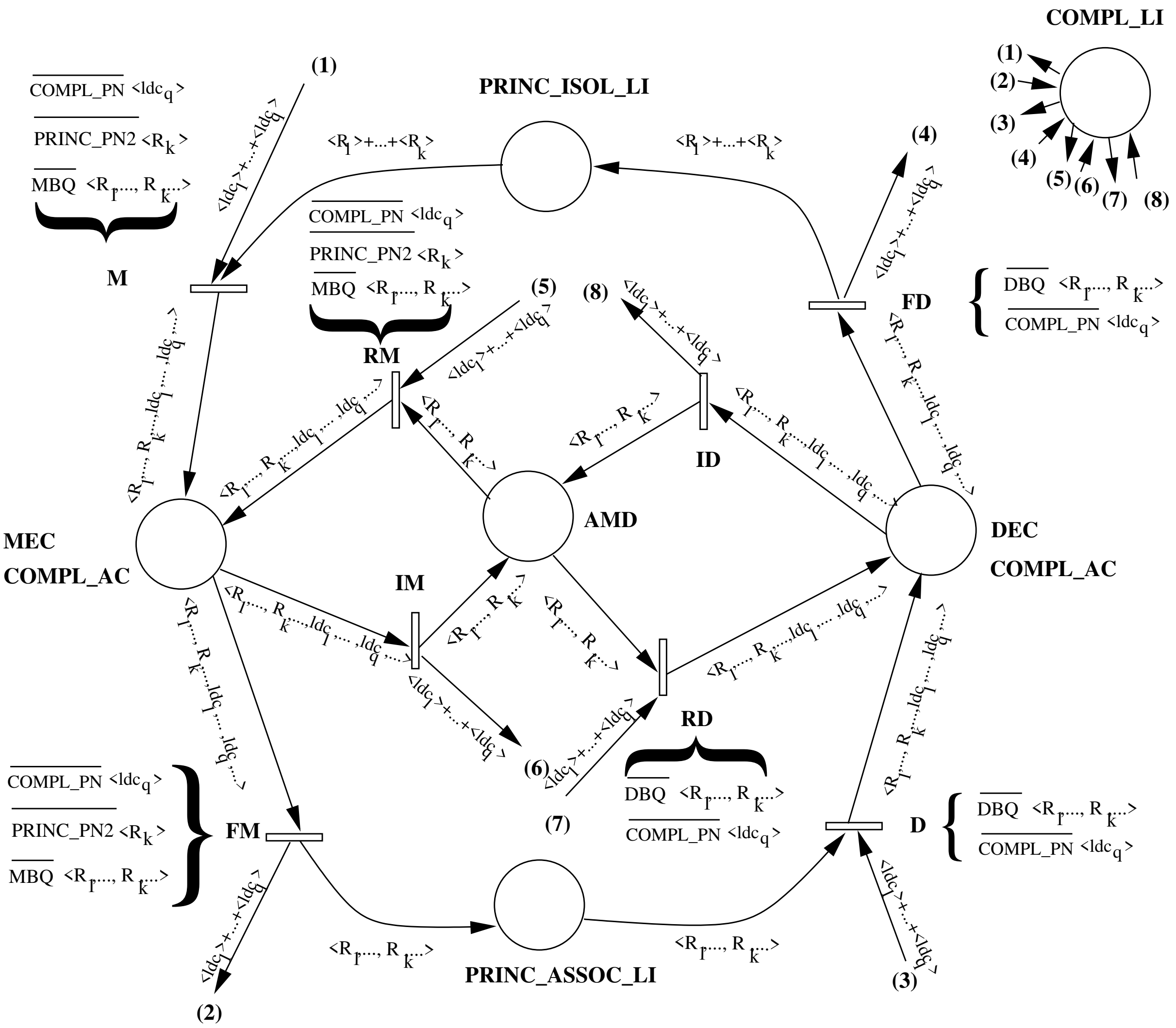


FIG. V.7 – *Changement d'état des entités mises en jeu pour le montage et le démontage*

par le démontage sur les ressources complémentaires. Elle permet :

- Suite à un blocage, de libérer les lignes de charge des ressources complémentaires de démontage, ou même d'annuler ce démontage.
- Suite à une panne de ligne de charge sur les ressources complémentaires, de reprendre le démontage sur des lignes de charge non en panne.

Changement d'état des entités mises en jeu pour le changement de type Dans la figure V.8, sont représentées les conditions précises de validité des décisions de changement de type (CT), d'interruption de changement de type (ICT), de reprise (RCT) et d'occurence des événements attendus de fin de changement de type (FCT). Les places intitulées PRINC_ISOL_LI et COMPL_LI sont les mêmes que dans la figure précédente. Le changement de type des ressources principales peut être effectué à partir de l'état isolé ou associé. Sur la figure V.8, seul le changement de type d'une ressource principale isolée est représenté ($Ep2 =$ PRINC_ISOL), les deux processus étant identiques. A chaque instant , on trouve dans la place intitulée TOK, un ensemble de jetons composés du type $< R_k, X >$ où R_k est une ressource principale et X est le type d'opérations d'exécution pour lequel la ressource k est préparée (champ $Ep1$). Les autres jetons $< R_k, Y >$ où $Y \neq X$ sont dans la place $\overline{TOK}$, ce qui signifie que R_k ne peut pas exécuter d'opération du type Y.

La décision de changement du type X vers le type Y (CTI) fait passer la ressource principale R_k de l'état $Ep1 = X$ à l'état de changement de type en cours $Ep1 =$ CTEC et le jeton $< R_k, X >$ de la place TOK à la place $\overline{TOK}$, ce qui signifie que la ressource n'est plus préparée pour le type X. Le jeton $< R_k, Y >$ est ôté de la place $\overline{\text{TOK}}$, ce qui signifie que la ressource est en cours de préparation pour le type Y.

L'événement de fin de changement de type (FCTI) fait passer la ressource principale R_k de l'état changement de type en cours $Ep1 =$CTEC à l'état $Ep1 = Y$. Le jeton $< R_k, Y >$ est ajouté à la place TOK. La ressource peut exécuter des opérations du type Y.

Comme pour les autres décisions d'engagement, le changement de type utilise des ressources complémentaires. Les événements de début et de fin de changement de type ne peuvent avoir lieu si les ressources complémentaires sont en panne ou si l'opération de changement de type est bloquée. Dans le cas où un de ces deux aléas arrive pendant le changement de type, l'interruption (ICTI) permet de libérer les lignes de charge des ressources complémentaires. Le jeton $< R_k, Y >$ repasse alors dans la place $\overline{TOK}$, ce qui signifie que le changement de type est interrompu (état $Ep1 =$ACT). La seule décision possible est une reprise de changement de type (RCTI) sur une ensemble de lignes de charge complémentaires différent (cas de la panne) ou vers un type destination différent (cas du blocage).

V.4.4.c) Valeurs implicites des champs

Certains champs des états des entités définis en V.4.3 ne sont pas directement lisibles dans le réseau. Les valeurs de ces champs sont obtenues par le système en fonction des valeurs des autres champs et de l'état des autres entités. Nous donnons les règles de génération des valeurs des champs concernés :

champ $Ep5$. Ce champ concerne l'anomalie externe sur une ressource principale. L'anomalie externe est détectée par la présence du jeton ressource $< R_k >$ dans un jeton composé pour lequel au moins une entité représentée par un autre jeton est dans l'état en panne si l'entité est une ressource, ou bloqué si l'entité est une opération.

champ $Ec3$. Ce champ concerne l'anomalie externe sur une ressource complémentaire. L'anomalie externe est detectée pour le jeton $< ldc_q >$ de la même manière que pour une ressource principale

champ $Eo2$. Ce champ concerne une opération (i, j) couplée à une ressource en panne. La panne est détectée par la présence de $< (i, j) >$ dans un jeton composé contenant aussi un jeton $< R_k >$ tel que R_k soit en panne, ou un jeton $< ldc_k >$ tel que ldc_k soit en panne.

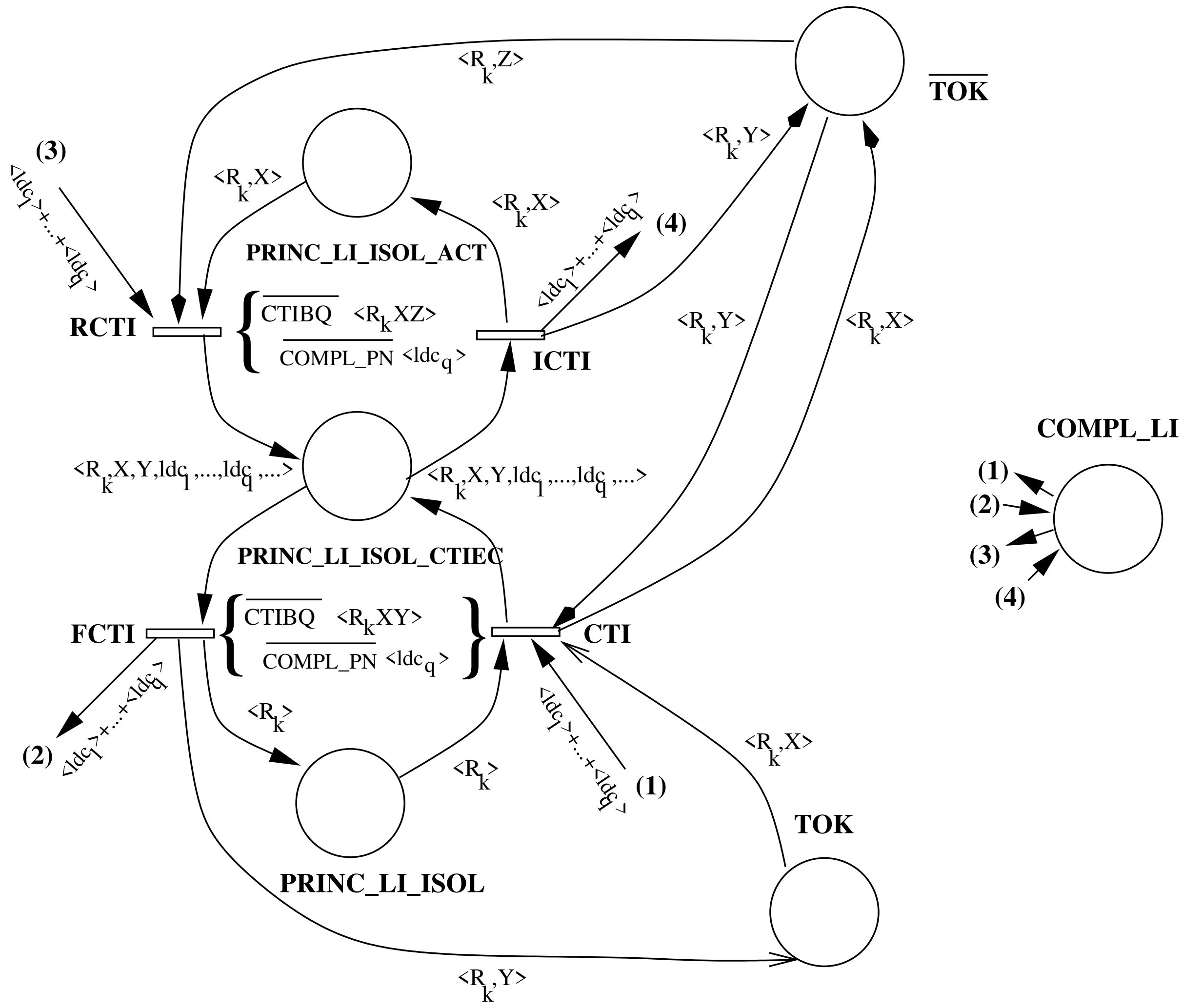


FIG. V.8 – *Changement d'état des entités mises en jeu pour le changement de type*

champ $Es1$**,** $Es2$ **et** $Es3$. L'état d'une opération de préparation est généré par le système en fonction de l'état des ressources principales préparées.

V.4.4.d) Marquage initial

Le marquage initial du réseau n'a pas été représenté car il dépend de l'état initial réel de l'atelier. Un certain nombre de ressources peuvent être associées, d'autres isolées, en panne etc...

V.4.5 Procédures d'aide à la décision

La séquence de groupes est utilisée par le système d'aide à la décision pour proposer, à l'occasion de chaque décision, un ensemble d'actions pertinentes, ou pour tester les propositions du décideur et calculer l'impact d'une décision envisagée sur les délais des ordres de fabrication. La validité de chaque décision est testée en utilisant le réseau de Petri décrit dans le paragraphe précédent, en fonction de l'état des différentes entités composant l'atelier. Les dates de début au plus tôt et de fin au plus tard des opérations sont recalculées suite à un événement ou à une décision par propagation dans les graphes potentiels-tâches afin de calculer en particulier les marges des opérations d'exécution et de préparation. A un instant donné, un *groupe de tête* G_t est un groupe qui est en première position sur toutes les ressources et lignes de charge qu'il utilise. Chaque groupe de tête peut être composé :

- soit d'une unique opération d'exécution couplée aux ressources (état $Eo1$ =EEC).
- soit d'un ensemble d'opérations d'exécution non couplées (états $Eo1$ ={BL, AP, DI})
- soit d'une unique opération de préparation disponible (états $Es1$ ={MDI, DI, CTIDI, CTADI})
- soit d'une unique opération de préparation en cours (états $Es1$ ={MEC, DEC, CTIEC, CTAEC}.

V.4.5.a) Marges des opérations et surveillance de l'admissibilité de la séquence de groupes

L'utilisation d'une séquence de groupes pour l'aide à la décision suppose de surveiller en permanence l'admissibilité de cette séquence, ce qui est réalisé au moyen des marges définies ci-dessous.

Marge libre séquentielle des opérations d'exécution non couplées. On distingue deux marges associées à une opération d'exécution (i,j) à l'intérieur d'un groupe G à l'instant t :

- la *marge propre* $sl_p(i,j,t)$ met uniquement en jeu la date de début au plus tôt réactualisée $r_{ij}(t)$ (voir équation V.17), et la date de fin au plus tard d_{ij} de (i,j) calculée pour le respect de l'admissibilité à la fin de la procédure de génération de la séquence:

$$sl_p(i,j,t) = d_{ij} - p_{ij} - r_{ij}(t) \tag{V.14}$$

- la *marge groupe* $sl_g(i,j,t)$ prend en compte les autres opérations de G (si elles existent) en considérant les permutations les plus défavorables pour lesquelles (i,j) est exécutée en premier :

$$sl_g(i,j,t) = \min_{(u,v)\in G,(u,v)\neq(i,j)} d_{u,v} - \sum_{(u,v)\in G} p_{uv} - r_{ij}(t) \tag{V.15}$$

Cette marge groupe n'existe que si G contient au moins deux opérations.

La *marge libre séquentielle* $sl(i,j,t)$ est définie par [Thomas, 1980] comme le minimum des deux marges définies ci-dessus :

$$sl(i,j,t) = \min\{sl_p(i,j,t), sl_g(i,j,t)\} \tag{V.16}$$

Si l'opération (i,j) appartient à un groupe de tête, et si elle est disponible (toutes ses précédentes dans la gamme sont terminées) la date de début au plus tôt $r_{ij}(t)$ peut être définie :

$$r_{ij}(t) = \max(t, r_{ij}(t-1)) \tag{V.17}$$

à la fin du module ANALYSE.

Si l'opération (i,j) n'appartient pas à un groupe de tête, $r_{ij}(t)$ est obtenue par propagation dans $\mathcal{G}_G$ à partir des groupes de tête.

Marge des opérations de préparation non en cours. Pour une opération de préparation s non en cours, seule la marge propre est définie et on a :

$$sl(s,t) = sl_p(s) = d_s - p_s - r_s(t) \tag{V.18}$$

Si le groupe de s est de tête, la date de début au plus tôt $r_s(t)$ est égale à :

$$r_s(t) = \max(t, r_s(t-1)) \tag{V.19}$$

où $r_s(t_s)$ est la date de début au plus tôt prévue à la date t_s de création de l'opération de préparation. Sinon, $r_s(t)$ est obtenue par propagation dans $\mathcal{G}_G$ à partir des groupes de tête.

Marge des opérations de préparation ou d'exécution en cours, non arrétées et non bloquées Pour une opération d'exécution ou de préparation o en cours, la marge est calculée à partir de la date t_o^e d'engagement et la date courante t :

$$sl(o,t) = d_o - \max(t, t_o^e + p_o) \tag{V.20}$$

Marge des opérations de préparation ou d'exécution arrétées ou bloquées. Pour une opération d'exécution ou de préparation arrêtée suite à une panne de ressource, ou bloquée suite à un problème lié à l'ordre de fabrication ou a la préparation, la marge est calculée en fonction du temps restant à faire estimé $p_o^a = \max(0, p_o - t_o^a + t_o^e)$ où t_o^a est la date de début de l'aléa (panne ou blocage), de la date courante t et de la durée p_a estimée de l'aléa (éventuellemet nulle si elle est inconnue) :

$$sl(o,t) = d_o - (t + p_a + p_o^a) \tag{V.21}$$

Si la fin de l'aléa survient à la date t et si l'opération n'a pas été interrompue, on réactualise sa durée en posant $p_o = p_o^a$, sa date de début d'exécution en posant $t_o^e = t$ et le calcul de la marge se fait en utilisant la relation V.20. Si l'opération est interrompue, on réactualise sa durée en posant $p_o = p_o^a$ et sa date de début au plus tôt en posant $r_{ij}(t) = t$. Le calcul de la marge se fait alors en utilisant la relation V.18 si o est une opération de préparation et la relation V.14 si o est une opération d'exécution.

Surveillance de l'admissibilité de la séquence de groupes. Par abus de langage, la marge des opérations de préparation non en cours, des opérations de préparation ou d'exécution en cours, non arrétées et non bloquées ainsi que des opérations de préparation ou d'exécution arrétées ou bloquée est appelée dans la suite marge libre séquentielle.

Pour surveiller l'admissibilité, il est suffisant d'observer les marges libres séquentielles des opérations constituant les différents groupes de tête sur toutes les ressources. Les conditions nécessaires suffisantes d'admissibilité de la séquence de groupes sont définies par :

$$\forall G_t, \forall o \in G_t, sl(o,t) \geq 0 \tag{V.22}$$

V.4.5.b) Aide à la décision dans le respect du contexte planifié

Pour la prise de décision, le respect du contexte planifié est possible et souhaitable tant que la séquence de groupes est admissible. Dans ce cas, le (ou les) décideur(s) exécutent les opérations dans l'ordre des groupes prévu au terme de la procédure de génération de la séquence. Les opérations de préparation, ainsi que les opérations d'exécution appartenant à un groupe composé d'une seule opération, sont exécutées strictement dans l'ordre prévu. L'aide porte uniquement sur le choix d'une opération à engager, dans le cas où un groupe de tête est composé de plusieurs opérations.

Dans ce cas (voir [Thomas, 1980]), il apparaît que la meilleure décision pour préserver la flexibilité de nature temporelle associée à G_t est de sélectionner l'opération de G_t qui a la plus grande marge libre séquentielle et de l'engager immédiatement si elle est disponible.

Pour les opérations de préparation disponibles et les opérations d'exécution d'un groupe à une seule opération, l'engagement conforme au contexte planifié doit être effectué entre la date de début au plus tôt r_o et la date de début au plus tard $d_o - p_o$.

V.4.5.c) Aide à la décision en cas de non-respect du contexte planifié ou en présence d'aléas internes

i) Non respect du contexte planifié Une décision peut être envisagée et même prise par le décideur sans que le système la propose. On est alors dans le cas d'un non-respect volontaire du contexte planifié. Ceci arrive par exemple en fonction des préférences des décideurs ou pour respecter des contraintes non modélisées dans le problème inital. Dans ce cas, l'aide à la décision consiste à estimer la conséquence de l'action envisagée sur l'admissibilité de la séquence, ce qui est réalisé en recalculant les marges des opérations.

ii) Réaction à des aléas internes L'arrivée d'un aléa interne ne doit pas être obligatoirement suivie d'une action corrective immédiate. Cette action corrective et la prise de décision associée peuvent être reportées au plus tard au moment où l'aléa provoque la perte de l'admissibilité de la séquence [Le Gall, 1989]. Ainsi l'aide à la décision face à un aléa consiste d'une part à diagnostiquer par l'intermédiaire des marges la conséquence de cet aléa, et d'autre part, si nécessaire, à aider le décideur dans le choix d'une action corrective.

ii1) Surveillance et diagnostic Si la durée de l'aléa est estimée, on calcule les nouvelles marges par propagation de cette durée. Si toutes les marges sont positives, l'aléa ne pose a priori pas de problème et il n'y a pour l'instant pas lieu de déclencher une action corrective (réaction différée). Si une marge est négative, alors on doit envisager une réaction immédiate.

Dans le cas où on ne détecte pas la nécessité d'une réaction immédiate corrective, ce qui se produit si les marges prévisionnelles sont positives ou si on ne dispose pas d'une durée estimée de l'aléa, alors on suit la dérive pour détecter la date "limite" où une réaction s'imposerait.

Tant que toute opération o d'un groupe de tête est telle que $sl(o,t) \geq 0$, l'admissibilité est toujours garantie par le respect de la séquence de groupes prévue. A partir du moment où il existe une opération o d'un groupe de tête telle que $sl(o,t) < 0$, le respect du contexte planifié ne permet plus d'assurer les conditions d'admissibilité de la séquence de groupes. Dans ce cas, au moins un ordre de fabrication est en retard de $sl(o,t)$ unités de temps. Le décideur peut interroger ARBITRE pour connaître exactement les ordres de fabrication en retard et leur date de fin au plus tôt. Ceci est obtenu par actualisation des dates de début au plus tôt et propagation dans $\mathcal{G}_G$. Si ces retards sont jugés acceptables, le décideur ne recherche pas d'action corrective et les ordres de fabrication concernés prennent comme nouvelles dates de fin au plus tard, leur date de fin au plus tôt. La réactualisation des dates de fin au plus tard des autres opérations par propagation à rebours depuis les sommets h_i concernés rend toutes les marges positives ou nulles. On peut suivre à nouveau la séquence de groupes prévue.

Si certains de ces retards sont jugés inacceptables, la récupération de l'admissibilité peut passer par le retour à l'étape de génération de la séquence de groupe. Le décideur peut juger que ce retour n'est pas nécessaire ou n'est pas souhaitable, en particulier pour ne pas trop modifier la séquence de groupe prévue. Il peut alors tenter de mettre en oeuvre un ensemble de décisions locales correctives, sur la base de propositions effectuées par ARBITRE.

ii2) Recherche d'une décision corrective Une décision corrective consiste essentiellement à changer les modalités prévues d'utilisation des ressources. Pour cela, on distingue la décision d'engagement d'une opération d'exécution sur ses ressources prévues alors qu'elle n'est pas dans un groupe de tête, de la décision de changement de l'affectation prévue pour une opération (d'exécution ou de préparation). Dans les deux cas, on décrit ci-dessous comment ARBITRE détermine l'impact de ces deux décisions.

Décision E. Il est possible d'engager une opération d'exécution disponible qui ne se trouve pas dans un groupe de tête, sans changement d'affectation. Une telle opération est appelée opération d'exécution engageable. Le décideur peut alors demander la recherche des opérations d'exécution engageables sur un ensemble de ressources données. Pour l'aider dans sa décision, il est nécessaire de tester l'impact de l'engagement de chaque opération engageable sur l'admissibilité de la séquence de groupe. On tombe sur un problème d'insertion en tête des ressources, avec éventuellement un problème de préparation associé. Soit (i, j) une opération d'exécution engageable sur un ensemble de ressources donné. Pour effectuer ces tests, le système utilise si nécessaire la procédure HeurInsèrePrépa (voir paragraphe III.6.1) qui génère, sous la forme du graphe d'opérations de préparation $\mathcal{G}_{ij}$, l'activité de préparation nécessaire préalablement à l'engagement de cette opération d'exécution sur les ressources prévues, en optimisant l'utilisation des ressources complémentaires de préparation. Le système teste si l'état des ressources et des opérations permet cet engagement. ARBITRE considère qu'un engagement est possible si sa mise en oeuvre ne nécessite que l'occurence d'une suite de décisions (d'interruption), mais d'aucun événement attendu (comme la fin d'un aléa). Par exemple, suite à la génération de $\mathcal{G}_{ij}$, le système teste automatiquement l'engagement des premières opérations de préparation de ce graphe. Chaque opération de préparation nécessitant éventuellement des ressources complémentaires, cet engagement n'est pas possible si une des lignes de charge des ressources complémentaires requises pour une opération de préparation est en panne, ou si une des opérations de préparation nécessaires est bloquée. Dans ce cas, l'engagement de (i, j) n'est pas envisagé. Par contre, si certaines des ressources requises par (i, j) sont occupées par une opération d'exécution couplée, ARBITRE prend en compte la nécessité d'interrompre ces opérations préalablement à l'engagement de (i, j). Dans ce cas, cet engagement peut être envisagé. L'impact de l'insertion d'un tel graphe et de l'opération d'exécution (i, j) au début du graphe $\mathcal{G}_G$ est évalué. Au préalable, (i, j) est provisoirement enlevée du graphe $\mathcal{G}$ et les dates de fin au plus tard des opérations précédentes dans $\mathcal{G}$ sont recalculées (voir paragraphe IV.2.3).

Décision C. On n'envisage les changements d'affectation que pour des opérations d'exécution ou de préparation situées dans le groupe de tête ou engageables. Le changement d'affectation d'une opération de préparation s ne peut être envisagé que sur des ressources complémentaires. Le changement d'affectation sur les ressources principales n'a pas de sens. Une requète permet au décideur de connaître l'ensemble des changements d'affectation proposé par le système ainsi que l'impact de chaque changement d'affectation sur la tenue des délais. La recherche des changements d'affectation des opérations d'exécution sur les ressources principales et le calcul de l'impact sur l'admissibilité sont effectués par les procédures d'insertion présentées aux paragraphes III.4 et III.6. La recherche des changements d'affectation des opérations de préparation ou d'exécution sur les ressources complémentaires et le calcul de l'impact sur l'admissibilité sont effectués par les procédures d'insertion présentées aux paragraphes III.2 et III.3. De même que pour l'engagement, l'ensemble des décisions préalables au changement d'affectation d'une opération est envisagé. Par exemple, on doit d'abord interrompre une opération couplée avant de la changer d'affectation.

On peut remarquer que la connaissance précise de l'état de l'atelier par le système lui permet de ne proposer que des décisions correctives réalisables. Le système donne la liste des décisions proposées qui amènent un gain en terme d'admissibilité. Le décideur peut choisir la décision de plus grand gain. Dans le cas de la décision d'engagement d'une opération non située dans un groupe de tête, le gain est donné sous la forme d'une marge d'engagement. Si aucune décision proposée ne permet d'améliorer l'admissibilité, il est préférable de continuer à respecter la séquence planifiée. On doit souligner l'importance de la durée estimée de l'aléa pour le calcul de l'impact des décisions correctives envisagées.

ii3) Réalisation de la décision corrective Si le décideur choisit d'engager une opération (i,j) qui n'est pas dans un groupe de tête sans changer son affectation, le système est en mesure de lui proposer :

- l'ensemble des décisions éventuelles d'interruption des opérations d'exécution et de préparation utilisant les ressources requises par (i,j),
- la mise en oeuvre des éventuelles opérations de préparation des ressources principales de (i,j) (graphe $\mathcal{G}_{ij}$ inséré) au moyen d'un ensemble de ressources complémentaires de préparation proposées.

Si le décideur choisit d'effectuer un changement d'affectation, le système est en mesure de lui proposer :

- l'interruption éventuelle de l'opération (de préparation ou d'exécution) avant son changement d'affectation,
- l'ensemble des éventuelles activités de préparation des ressources principales induites par le changement d'affectation (anciennes et nouvelles ressources principales utilisées par l'opération).

V.4.5.d) Aide a la décision en cas d'aléa externe (modification en temps réel du programme de production)

Lors de l'arrivée imprévue d'un ordre de fabrication ou de l'arrivée imprévue d'une opération d'exécution dans une gamme, le décideur peut souhaiter prendre en compte cette modification du plan de production par une insertion de cet ordre de fabrication ou de cette opération dans la séquence de groupes existante sans remettre en cause cette séquence et en minimisant l'impact de l'insertion sur l'admissibilité. Pour chacune des opérations à insérer, on utilise directement les procédures d'insertion décrites au chapitre III.

Insertion d'une opération d'exécution dans une gamme Dans le cas où une opération d'exécution (i,j) doit être ajoutée à la gamme de l'ordre de fabrication i déjà inclus dans la séquence de groupes, on est exactement dans les conditions d'insertion décrites au paragraphe III.1. Le seul problème est la possibilité, si les gammes sont non linéaires, d'existence d'un chemin dans le graphe entre certaines opérations de l'ensemble $succ^i_{ij}$ et certaines opérations de l'ensemble $prec^i_{ij}$ pour lesquelles une relation de précédence n'était pas imposée dans la gamme avant la demande d'insertion. Dans ce cas, pour pouvoir insérer l'opération, on peut d'abord oter du graphe toutes les opérations suivantes dans la gamme g_i de chaque opération de l'ensemble $prec^i_{ij}$. Les opérations sont ensuite insérées une par une dans un ordre cohérent avec la gamme g^*_i incluant l'opération (i, j).

Insertion d'un ordre de fabrication imprévu Dans [Le Gall, 1989] et dans [Billaut *et al.*, 1996], une procédure d'insertion d'un ordre de fabrication imprévu a été présentée. Le choix de cette méthode est d'autoriser l'insertion uniquement sans remettre en cause la tenue des délais des ordres de fabrication déja présents. Dans certains cas, l'ordre de fabrication inséré peut être rejeté ainsi à la fin de la séquence. Ceci peut être en contradiction avec le fait que l'insertion en temps réel est souvent motivée par le caractère urgent de l'ordre de fabrication imprévu.

Ce choix n'a pas été retenu dans notre approche qui effectue un compromis entre le respect des dates de fin au plus tard des opérations déjà présentes et les dates de fin au plus tard estimées des opérations de l'ordre de fabrication inséré. Les opérations de l'ordre de fabrication sont insérées une par une comme pour la procédure de génération alternative de la séquence initale présentée en V.2.4. On donne à chaque opération insérée une date de fin au plus tard estimée par distribution de la marge de l'ordre de fabrication. L'insertion de l'ordre de fabrication est ensuite ajustée par un processus interactif. A l'issue de l'insertion la liste des ordres de fabrication décalés peut être consultée, ainsi que la date de fin au plus tôt de l'ordre de fabrication inséré. S'il juge que les décalages des ordres de fabrication déja présents sont trop importants le décideur peut annuler l'insertion, augmenter la date de livraison de l'ordre de fabrication et le ré-insérer à nouveau. Inversement, si l'ordre de fabrication est trop pénalisé, sa date de livraison peut être diminuée. A l'issue de ce processus, le décideur peut parvenir à un compromis satisfaisant.

V.5 Intégration dans le logiciel ORDO^R

V.5.1 Presentation du logiciel [Villaumié, 1997a]

Le logiciel d'Ordonnancement en Temps réel d'Atelier ORDO^R développé et commercialisé par la société CABINET VILLAUMIE SA est bâti autour de l'algorithme ORABAID développé au LAAS.

ORDO est disponible sur système mini-informatique de type Digital (VAX-VMS, ALPHA-OPEN VMS, ALPHA-OSF1), Hewlett-Packard (HP9000-HP-UX), IBM (RS6000-AIX).

ORDO est composé de différents modules, décrits dans le paragraphe V.5.1.a) et est intégré au sein d'une plateforme de logiciels dans l'optique d'un Management Dynamique des Flux^R, concept créé par Cabinet Villaumié SA, qui est décrit dans le paragraphe V.5.1.b).

V.5.1.a) Les modules d'ORDO^R

Le Parc est l'ensemble des données techniques de base utilisé par le logiciel pour décrire l'atelier sur lequel portera l'ordonnancement. Parmi ces données on distingue :

- Les ressources (voir paragraphe II.2.2.a)). On distingue la désignation H pour Homme, M pour machine (ressources disjonctives) E pour Equipe (ressource cumulative), S pour sous-traitant (modélisé comme une ressource à capacité illimitée).
- Les pools de ressources (voir paragraphe II.2.2.a)).
- Les sections d'atelier. L'atelier peut être décomposé en sections. Chaque section regroupe un ensemble de ressources placées sous la responsabilité d'une personne.
- Les calendriers. Dans les cas réels, les ressources ne sont pas disponibles en permanence. Un calendrier de fonctionnement est associé à chaque ressource. Un calendrier est défini par un ensemble de périodes et un nombre de lignes de charge (au maximum 1 pour une ressource disjonctive et $E_k > 1$ pour une ressource cumulative) utilisables par période. Dans [Billaut, 1993] et [Artigues *et al.*, 1996], la prise en compte des calendriers dans la méthode ORABAID est décrite.
- Les opérateurs. Les opérateurs sont recensés afin de pouvoir éditer un journal des activités opérateurs issu du suivi de fabrication.
- les informations liées aux activités de préparation (voir paragraphe II.2.2.c)). Il s'agit de la description des temps de montage et de démontage, de changement de type ainsi que des ressources complémentaires requises pour chaque mode étendu de préparation.
- les gammes de fabrication, les ordres de fabrication et les opérations d'exécution (voir paragraphe II.2.2.b). ORDO^R peut fonctionner de différentes façons : en autonomie complète, par création directe des ordres de fabrication, ou bien par création des ordres de fabrication à partir d'une gamme, ou encore interfacé avec un logiciel de GPAO de type MRP avec descente des ordres de fabrication gammés dans ORDO^R. Il est possible de constituer des réseaux d'ordres de fabrication représentés par un ensemble de contraintes de précédence entre opérations d'ordres de fabrication différents (voir [Le Gall, 1989], [Billaut, 1993]).
- La grille des implantations. Il s'agit d'une matrice de temps de transit séparant deux opérations consécutives dans un même ordre de fabrication. Un temps de transit positif représente un temps de transport, un temps de transit négatif représente un chevauchement. La prise en compte des temps de transport et du chevauchement des opérations dans la méthode ORABAID a été présentée dans [Billaut, 1993].

Le Planning Prévisionnel à capacité illimitée réalise une étape essentielle qui consiste à analyser l'adéquation des capacités des postes de charges (ressources), par rapport aux charges matérialisées par les ordres de fabrication à réaliser, dans les délais demandés. Face à une situation de surcharge, le décideur a la possibilité de lisser la charge en ajoutant des heures de travail, en fractionnant les ordres de fabrication, en sous-traitant ou en négociant les dates de livraison.

```
 Ordo v4                          SUIVI D'ATELIER  Page :  1      02/07/97  08H48

Ressource : R1                             Capa : L Etat : LI Liberte :     -1
----------------T--T-------------------------------------------------T----------
marge    OF     |ev|  libelle      op    tta    tdp   qte      nb etat|  compl.
----------------+--+-------------------------------------------------+----------
  -1  OF707063  |  |37002-01      10    2.00   0.00     1200  1 TP   |R1
  -1  OF707057  |  |37002-01      10    2.00   0.00     1200  1 TP N|R1
  -1  OF707022  |  |37002-01      10    2.00   0.00     1200  1 TP N|R1
   0  P718101394|  |R ROUGE JAUNE 10    5.00   0.00        1  2 TP N|R1
  -3  OF707045  |E |3116-01       10    2.00   0.00     1200  1 TP N|R1
   0  P718101399|  |R JAUNE VERT  10    2.00   0.00        1  2 TP N|R1
  -3  OF707098  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
  -3  OF707033  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
  -3  OF707015  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
   0  P718101403|  |R VERT BLEU   10    1.50   0.00        1  2 TP N|R1
  -3  OF707104  |  |15006-01      10    2.00   0.00     1200  1 TP N|R1
  -3  OF707003  |  |15006-01      10    2.00   0.00     1200  1 TP N|R1
----------------T--T-------------------------------------------------T----------
panne,tit,tir :            Duree panne :

   OF             op
                                                     Option (V,A,M)  : v
                  PF2 : Ecran precedent, PF3 : Ecran suivant
```

```
 Ordo v4                          SUIVI D'ATELIER  Page :  1      02/07/97  08H51

Ressource : R1                             Capa : L Etat : AC Liberte :     -1
----------------T--T-------------------------------------------------T----------
marge    OF     |ev|  libelle      op    tta    tdp   qte      nb etat|  compl.
----------------+--+-------------------------------------------------+----------
  -1  P718101448|__|R ROUGE JAUNE 10    5.00   0.00        1  2 EC   |4.99
  -2  OF707045  |  |3116-01       10    2.00   0.00     1200  1 DI N|
   0  P718101449|  |R JAUNE ROUGE 10    1.00   0.00        1  2 TP N|R1
  -2  OF707063  |  |37002-01      10    2.00   0.00     1200  1 TP N|R1
  -2  OF707057  |  |37002-01      10    2.00   0.00     1200  1 TP N|R1
  -2  OF707022  |  |37002-01      10    2.00   0.00     1200  1 TP N|R1
   0  P718101399|  |R ROUGE VERT  10    4.00   0.00        1  2 TP N|R1
  -3  OF707098  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
  -3  OF707033  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
  -3  OF707015  |  |1257-01       10    2.00   0.00     1200  1 TP N|R1
   0  P718101403|  |R VERT BLEU   10    1.50   0.00        1  2 TP N|R1
  -3  OF707104  |  |15006-01      10    2.00   0.00     1200  1 TP N|R1
----------------T--T-------------------------------------------------T----------
panne,tit,tir :            Duree panne :

   OF             op
                                                     Option (V,A,M)  :
                  PF2 : Ecran precedent, PF3 : Ecran suivant
```

FIG. V.9 – *Exemple d'écran de suivi d'une file d'attente*

Le Planning Réel à capacité limitée exécute le module ORABAID de génération d'une séquence de groupes initiale, présenté en V.2 et le module ORABAID d'amélioration de cette séquence présenté en V.3, intégrant les fonctionnalités développées dans cette étude.

La Distribution du Travail dans l'atelier s'effectue principalement par le suivi des files d'attentes (voir ci-après). Il est aussi possible d'éditer tous les documents nécessaires à la constitution du dossier de fabrication, à savoir le plan de charge, les ordres de fabrication planifiés, les fiches suiveuses, les bons de travail et les bons divers (sous-traitance, magasin).

Le Suivi Temps Réel est la partie utilisée dans l'atelier. Chaque module de suivi est un client connecté au serveur ARBITRE (module ORABAID présenté en V.4). Conversationnel, son utilisation est capitale puisque les déclarations faites sur l'état des ressources et des ordres de fabrication alimentent le recalage temps réel de la séquence de groupes et de la distribution du travail dans les files d'attente par ARBITRE.

Dans la figure V.9, des écrans de suivi des files d'attente d'une ressource principale disjonctive $R1$ sont représentés. La liste des opérations à réaliser dans l'ordre préférentiel est donnée (repère 1). On distingue les opérations des ordres de fabrication notées ($OF[...]$) des opérations de préparation notées $P[...]$. Le nombre de ressources requises par l'opération (incluant la ressource suivie) est indiqué (repère 2) ainsi que l'état de l'opération (repère 3). La marge V.4.5.a) de chaque opération d'exécution ou de préparation est indiquée (repère 4). Pour les opérations non situées dans le groupe de tête (non situées en première position de la file d'attente), cette marge est la marge d'engagement. L'indicateur donné dans le repère 5 permet de savoir s'il est intéressant d'engager une opération non prévue dans le groupe de tête. Dans cette exemple tous les indicateurs sont à N (non), ce qui signifie que l'engagement d'une autre opération que celle prévue (à savoir l'opération 10 de l'ordre de fabrication 707063) augmentera le retard vrai d'au moins un ordre de fabrication. Ceci est vérifiable par la présence de marges d'engagement négatives. La colonne ev permet au décideur de déclarer les événements et les décisions concernant chaque opération. Dans l'exemple représenté, le décideur s'apprête à demander l'engagement de l'opération (707045,10). Cette opération est de type "$JAUNE$", comme l'indique le libellé des opérations de préparation qui l'encadrent, alors que la ressource $R1$ est préparée pour exécuter des opérations du type "$ROUGE$". Dans l'écran du bas de la figure V.9, en réponse à cette décision d'engagement, le système a proposé l'engagement d'une opération de changement du type $ROUGE$ vers le type $JAUNE$ préalable à l'engagement de l'opération (707045,10) ainsi que la génération d'opérations de changement de type à prévoir suite à la perturbation de la séquence prévue.

En plus des écrans de suivi des files d'attente des ressources, sont disponibles des écrans de suivi des ordres de fabrication présentant pour l'ordre de fabrication sélectionné toutes les informations relatives aux opérations terminées, en-cours ou non encore exécutées et aux délais et un tableau de bord qui permet d'avoir une vue générale d'une section de l'atelier en donnant la liste des ressources de la section, le degré de liberté de la ressource, l'opération en cours sur chaque ressource.

Statistiques et Historiques ORDO accepte l'addition d'outils pour générer états, analyses, statistiques au format et à la demande des utilisateurs.

Modules complémentaires Très ouvert, ORDO autorise l'addition d'options complémentaires visant à renforcer sa facilité d'intégration dans l'entreprise. ORDO Graphique permet notamment de visualiser les résultats des plannings prévisionnels et réels, des statistiques et des historiques sous forme graphique sous Windows sur PC.

En outre, des éléments de personnalisation sont intégrables dans la plupart des modules présentés ci-dessus.

V.5.1.b) Ordo au sein du Management Dynamique des Flux^R [Villaumié, 1997b]

Le Management Dynamique des Flux^R (voir figure V.10) est un concept novateur qui vise à synchroniser et à piloter en temps réel l'ensemble des flux de l'entreprise. Il repose sur deux principes essentiels :

- l'intégration des fonctions de l'entreprise réalisée par un ensemble de progiciels :
  - IDFLUX pour la gestion commerciale et la gestion de production,
  - Suivi d'activités Opérateurs,
  - IdMAP pour la maintenance des équipements et des outillages,
  - QUAL-Id, pour la maîtrise de la qualité,
  - ADIPROM pour la messagerie et l'administration des procédures
- la définition de boucles d'informations événementielles échangées entre fonctions principalement activées par ORDO. A partir des déclarations ou acquisitions des données, le système propose des analyses et actions portant sur :
  - la convergence synchronisée des flux physiques (recalage de la séquence de groupes, distribution du travail en atelier, listes à servir des matières, départs en sous-traitance, préparation des outillages,...)
  - le contrôle des flux administratifs et financiers (mouvement de personnel, valorisation des stocks et des en-cours, facturations,...)
  - la prise en compte des flux d'informations agissant sur l'ensemble du processus (délais clients, spécifications qualités, téléchargement de programmes).

V.5.1.c) Implantation industrielle

ORDO^R est implanté à l'heure actuelle dans 55 entreprises, parmi lesquelles de grands groupes industriels (Creusot Loire Industrie, Matra, Wavin), ainsi que des PME manufacturières de plus de 40 personnes dans les secteurs de l'automobile, de l'aéronautique, de l'électroménager, de la quincaillerie, de l'électronique, de l'imprimerie (voir exemple de

FIG. V.10 – *Le Management Dynamique des Flux*^{R}

la section V.5.2.b)), de l'horlogerie, etc. A titre d'exemple, la figure V.11 permet de juger des gains obtenu par une des entreprises utilisatrice des versions précédentes d'ORDO: la société SIOBRA déjà présentée au paragraphe I.1.1.a).

SOCIETE SIOBRA EN QUELQUES CHIFFRES	INDICATEURS DE GAINS SUR 8 ans
* Chiffre d'affaires 1996 : 60 Mf * Effectif moyen 100 personnes * Nombre de pièces produites : 24,4 millions/an * Nombre de clients : 95 * Nombre de type de pièces : 850/an * Nombre de nouveaux produits : 50/an * Nombre de lignes de commandes : 350/mois * Délais moyens : de 5 à 15 jours (30% pièces partent en sous-traitance 50% pièces font l'objet d'une reprise 100 pièces incorporées gérées sous forme de nomenclature) * Gammes : de 3 à 7 opérations * Ressources de production 21 en fonderie (35 postes en reprise finition) * Nombre d'OF (hors appel de livraison sur commande ouverte) : 300/mois	* Le chiffre d'affaires a doublé * Le nombre de pièces produites a doublé * Progression limitée des en-cours < 50% * Effectif +50% VA/personne : + 45% * Progression du nombre de presses : + 60% * Progression du nombre d'OF : de 200 à 350 OF/mois * Horizon de commandes : 2mois à 3 semaines * Retard moyen ramené de 3,5 j à 0-1 jour *Effectif Ordonnancement : 2 personnes -> 1 personne * Budget de maintenance informatique : env. 200 Kf/an

FIG. V.11 – *Gains obtenus suite à l'utilisation d'ORDO d'après [Laurent, 1997]*

V.5.2 Etude de cas réels

Dans ce paragraphe, deux exemples de sites industriels comportant des caractéristiques liées à la préparation dans un contexte multi-ressources sont décrits. La nouvelle version d'ORDO sera prochainement installée sur ces sites

V.5.2.a) Le cas X

Il s'agit de fabrication de tubes en PVC. Les tubes sont fabriqués par extrusion. La matière est tirée à partir d'une tête de machine qui définit le diamètre du tube (de 50 à 800 mm), son épaisseur et son type (plein ou alvéolé). L'atelier comporte m machines qui fonctionnent 24 heures sur 24.

Le changement de diamètre du tube fabriqué provoque le changement de toute la tête qui prend une journée entière. Le changement d'épaisseur prend quelques heures, la tête devant être uniquement ajustée mais non changée. Le changement de longueur ne prend que quelques minutes. Ces préparations sont effectuées par une unique équipe d'opérateurs qui n'est là que pendant la semaine, de 5 heures à 21 heures. L'extrusion est l'unique opération de la gamme de tous les ordres de fabrication.

Pour modéliser ce cas, seul le changement de type est nécessaire. Les têtes n'étant pas partagées entre les machines le montage et le démontage ne sont pas nécessaires. Par contre, l'utilisation des ressources complémentaires de changement de type est indispensable. L'équipe de préparateurs, modélisée par une ressource disjonctive unique est en effet une ressource critique lorsque de nombreuses préparations sont à effectuer, et doit être explicitement prise en compte. Le type associé aux opérations d'exécution contient l'information agrégée du diamètre, de l'épaisseur et du type du tube.

La production se faisant pour stock, les délais associés aux ordres de fabrication sont relativement souples. Par contre pour des raisons de coûts de fabrication, les décideurs peuvent restreindre le nombre de machines fonctionnant simultanément, sans décider explicitement quelles machines doivent être arrêtées. Pour prendre en compte la limitation volontaire du nombre $n_M(t)$ de machines fonctionnant simultanément à un instant t, un pool $P1$ contenant les m machines d'extrusion et un pool $P2$ contenant m ressources fictives sont définis. Toute opération d'exécution nécessite une ressource du pool $P1$ et une ressource du pool $P2$. En fonction du nombre $n_M(t)$, les calendriers des ressources fictives sont tels que seules $n_M(t)$ ressources soient disponibles à chaque instant t.

V.5.2.b) Le cas Y

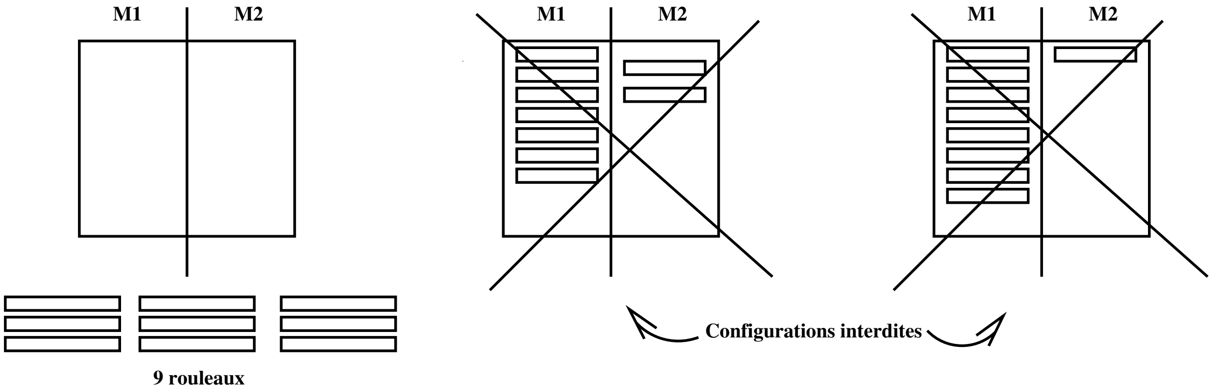


FIG. V.12 – *Machine à imprimer*

Il s'agit d'imprimerie d'emballages pour la grande distribution. Environ 400 ordres de fabrication sont fabriqués par mois, à la commande. Il est primordial de respecter les délais. Les gammes de fabrication sont longues (jusqu'à 15 opérations). La deuxième opération de la gamme est une opération d'impression de l'emballage et utilise une machine à imprimer. Une machine à imprimer est composée de deux machines identiques et indépendantes $M1$ et $M2$. Chaque opération d'exécution n'utilise qu'une des deux machines mais est définie par un certain nombre de couleurs différentes (9 au maximum). Chaque couleur nécessite la mise en place d'un rouleau. Les seules configurations interdites pour le partage des rouleaux par les machines $M1$ et $M2$ sont 7 rouleaux sur une des deux machines et 1 ou 2 rouleau(x) sur l'autre et 8 rouleaux sur une des deux machines et 1 rouleau sur l'autre (voir figure V.12). Toute opération utilisant au moins 7 rouleaux occupe obligatoirement les deux machines. La préparation consiste à la fois à enlever ou ajouter des rouleaux (en fonction du nombre de couleurs requises) et à changer les encriers (en fonction des couleurs requises). Les autres opérations des gammes (massicotage, emporte-pièce, vernissage, emballage) ne nécessitent pas de prise en compte de la préparation.

Dans cet exemple, il s'agit d'effectuer des regroupements technologiques des opérations nécessitant le même nombre de rouleaux et/ou les mêmes couleurs tout en respectant les délais. Pour les opérations d'impression, on définit un ensemble de ressources principales (la machine à imprimer et un ensemble de rouleaux). Toute opération utilisant plus de 6 rouleaux est définie comme utilisant 9 rouleaux pour éviter la planification des configurations interdites. La préparation consiste d'une part au montage et au démontage des rouleaux (ressources partagées par $M1$ et $M2$) et d'autre part au changement de type de la machine à imprimer (changement d'encrier) en fonction des couleurs précédentes et suivantes.

Conclusion

Le travail présenté dans ce mémoire concerne une méthode pour la prise en compte des temps de préparation des ressources dans les problèmes d'ordonnancement d'atelier en temps réel. Il repose sur la définition d'un modèle permettant une prise en compte fine de la préparation des ressources. Pour cela, l'activité de préparation est décomposée en trois opérations élémentaires. L'opération de montage et l'opération de démontage font appel à la notion d'association de ressources principales et permettent notamment d'exprimer les cas de partage d'outils communs à plusieurs machines. L'opération de changement de type correspond à des cas plus classiques de préparation dépendant de la séquence des opérations d'exécution sur une ressource principale isolée, ou sur un ensemble de ressources principales associées. Pour ces trois opérations de préparation élémentaires ainsi que pour l'opération d'exécution, des ressources complémentaires spécifiques peuvent être requises, permettant notamment la prise en compte d'équipes d'opérateurs spécialisés dans la préparation ou dans l'exécution. Les ressources principales, sur lesquelles s'effectue la préparation, sont supposées de type disjonctif. Par contre, les ressources complémentaires d'exécution ou de préparation peuvent être de type disjonctif ou cumulatif. Un état de l'art de la littérature concernant ce genre de problèmes met en évidence une lacune dans la prise en compte de ces caractéristiques. Pourtant, la considération de plusieurs cas industriels réels en démontre la nécessité. Pour faciliter la définition et la maintenance des données, l'utilisation des pools de ressources, déja validée en milieu industriel a été retenue. Toutefois, la flexibilité du modèle a été accrue par la définition de modes étendus pour l'exécution et la préparation de façon à exprimer, si le besoin s'en fait sentir, la spécialisation d'opérateurs par rapport aux machines ou bien la définition d'un nombre variable de ressources pour une même opération.

La caractérisation d'un ensemble d'ordonnancements par une séquence de groupes d'opérations permutables définie dans des travaux antérieurs du LAAS est étendue au contexte de cette étude. Des opérations d'exécution ne peuvent appartenir au même groupe que si elles utilisent les mêmes ressources principales et complémentaires et sont du même type. Ainsi, la constitution de groupes est-elle compatible avec les regroupements technologiques, qui visent à réduire les temps improductifs liés à la préparation. L'utilisation de graphes potentiels-tâches étendus au contexte de la préparation permet de calculer simplement l'admissibilité d'une séquence de groupes, c'est-à-dire la tenue des délais des ordres de fabrication.

Préalablement à la définition des procédures de génération de la séquence de groupe, une procédure d'insertion d'une opération d'exécution dans un ordonnancement unique est définie. L'objectif est d'insérer l'opération en minimisant l'impact sur l'admissibilité de la séquence, c'est-à-dire sur la violation des dates de fin au plus tard, sans déplacer les opérations déjà présentes. Ce problème est résolu de façon optimale dans plusieurs cas particuliers du modèle par des algorithmes de complexité polynomiale, grâce à l'utilisation de règles de dominance. Les règles de dominance sont d'abord définies dans le contexte multi-ressources cumulatif mais sans préparation. Elles permettent d'éviter l'explosion combinatoire qui résulterait de l'énumération des positions d'insertion, due à la présence de ressources cumulatives d'une part et de possibilités d'affectation multi-ressources (pools) d'autre part. L'introduction de la préparation des ressources principales permet de conserver ces relations de dominance sur la base d'hypothèses restrictives mais réalistes sur les temps de préparation (inégalités triangulaires) et si les opérations de préparation ne nécessitent pas de ressources complémentaires. Dans le cas général, une procédure heuristique est définie pour insérer progressivement les opérations de préparation générées par l'insertion sur les ressources complémentaires.

Cette procédure d'insertion est ensuite utilisée comme un outil générique et possède diverses applications. Elle est d'abord intégrée dans une méthode de type tabou d'amélioration d'un ordonnancement dans un contexte sans préparation. Les tests de cette méthode sur des problèmes d'ordonnancement de projet à moyens limités avec mode simple et modes multiples trouvés dans la littérature sont encourageants et montrent la souplesse du modèle d'ordonnancement proposé. De plus les résultats montrent qu'une approche basée sur les graphes peut être adaptée à la résolution de problèmes cumulatifs. Une autre méthode tabou utilisant deux types de voisinage est définie pour l'amélioration d'un ordonnancement dans les problèmes comportant des activités de préparation.

La méthode ORABAID d'Ordonnancement d'Atelier Basé sur une AIde à la Décision est étendue au contexte de cette étude. Une séquence de groupes initiale est construite au moyen d'un algorithme de liste cherchant à effectuer un compromis entre les regroupements technologiques et la tenue des délais. L'intérêt de l'utilisation de la procédure d'insertion étendue à une séquence de groupes dans cet algorithme est démontré à la fois pour réduire l'énumération des affectations possibles et pour compenser les défauts des algorithmes de liste classiques en présence de temps de préparation. Une procédure d'amélioration de la séquence de groupes initiale vis à vis des délais des ordres de fabrication est définie. Elle est basée sur l'application itérative de transferts d'opérations utilisant la procédure d'insertion proposée et de scissions définies dans les études précédentes. Un système interactif d'aide à la décision pour l'ordonnancement en temps réel basé sur l'exploitation de la séquence de groupes est présenté. Un ensemble d'états des entités de l'atelier, de décisions pertinentes et d'événements attendus et inattendus est identifié. Pour permettre au système de connaître précisément l'état de l'atelier, un réseau de Pétri coloré décrivant les enchaînements des états des entités en fonction de ces décisions et de ces événements est défini. L'ensemble des décisions envisageables à tout moment est ainsi identifié. L'aide a la décision concerne l'utilisation des ressources par les opérations d'exécution et de préparation. Elle permet ainsi d'estimer en temps réel le moment où il devient nécessaire pour la tenue des délais d'engager une opération de préparation plutôt que de continuer à engager des opérations d'exécution utilisant la préparation courante d'un ensemble de ressources principales. Elle permet aussi de gérer les ressources complémentaires de préparation dont la disponibilité peut jouer sur cette décision. Une nouvelle procédure d'insertion d'un ordre de fabrication en temps réel est définie. Cette procédure effectue de façon interactive un compromis entre le délai de l'ordre de fabrication inséré et les délais des ordres de fabrication déja ordonnancés. Enfin, la nouvelle version du logiciel ORDO^R commercialisée par la société CABINET VILLAUMIE SA (ORDOR v4.0) est présentée. Cette version intègre déjà certaines des caractérisiques faisant l'objet de cette étude, dont les opérations de changement de type, les ressources complémentaires de préparation et d'exécution et la nouvelle procédure d'insertion. La sortie de cette version a permis au logiciel ORDO^R d'intéresser de nouveaux clients concernés par les problèmes de préparation.

De nombreuses autres caractéristiques, liées ou non à la préparation, devraient être intégrées dans notre approche pour répondre à d'autres besoins. La présence dans l'atelier de ressources de type traitement thermique, qui suppose des regroupements d'opérations pour l'organisation de fournées pourrait être prise en compte, par exemple en définissant une préparation sur une ressource cumulative dont la capacité représenterait la capacité du four. La prise en compte des magasins d'outils nécessite aussi une légère adaptation du modèle et surtout une adaptation des méthodes de résolution. La prise en compte de l'existence de délais de péremption qui imposent un décalage temporel maximal entre certaines opérations d'un même produit nécessite l'introduction de nouvelles contraintes qui permettrait une ouverture vers l'industrie pharmaceutique et chimique. Enfin on peut envisager la possibilité de fractionnement d'un lot de produits, afin de réagir en temps réel à une situation anormale, ou de faciliter l'ordonnancement prévisionnel de l'atelier. Pour une prise en compte réaliste de cette dernière caractéristique, il est indispensable de considérer les opérations de préparation.

Bibliographie

[Adams *et al.*, 1988] J. Adams, E. Balas, D. Zawack. "The shifting bottleneck procedure for job shop scheduling", Management Science, 34:391-401, 1988.

[Aghezzaf *et al.*, 1995] E. A. Aghezzaf, A. Artiba, S. E. Elmaghraby, "Hybrid flowshops: an LP based heuristic for the planning level problems", ETFA, Paris, 551-559, octobre 1995.

[Aguilera, 1993] L.M. Aguilera, "Ordonnancement de production avec coûts de changement dépendant de la séquence", Thèse de doctorat, Institut National Polytechnique de Grenoble, 1993.

[Aguilera *et al.*,1996] L.M. Aguilera, Z. Binder, F. Hanada, J. J. Guimares Ramos, "Non-identical parallel machines scheduling with sequence-dependent changeover-costs in an industrial application", CESA'96 IMACS Multiconference, Computational Engineering in Systems Applications, Symposium on Discrete Events and Manufacturing Systems, Lille, 559-564, 1996.

[Ahn et Hyun, 1990] B.-H. Ahn, J. H. Hyun, "Single facility multi-class job scheduling", Computers and Operations Research, 17, 265-272, 1990.

[Alvarez-Valdes et Tamarit, 1989] R. Alvarez-Valdes, J.M. Tamarit, 'Heuristic algorithms for resource-constrained project scheduling: a review and an empirical analysis",Advances in Projet Scheduling, R. Slowinski, J. Weglarz, (Amsterdam: Elsevier), 113-134, 1989.

[Armentano et Mazzini, 1997] V. A. Armentano, R. Mazzini, "Genetic algorithms for minimizing tardiness on parallel machines with setups", IFAC/IFIP Conference on Management and Control of Production and Logistics, Campinas, SP, Brazil, 319-324, 1997.

[Arthanari et Ramamurthy, 1971] T. S. Arthanari, K. G. Ramamurthy, "A branch-and-bound algorithm for sequencing $n-$jobs and $m-$parallel processors", Opsearch, 7, p 147, 1971.

[Artigues *et al.*, 1996] C. Artigues, F. Roubellat, J.-C. Billaut, "Characterization of a set of schedules in a resource constrained multi-project scheduling problem with multiple modes", Extended Abstract, Workshop on Production Planning and Control, FUCaM-EIASM-ICM, 251-254, Mons (Belgium), september 9-11, 1996, à paraître dans International Journal of Industrial Engineering, Special Issue on Project Management.

[Artigues et Roubellat, 1996] C. Artigues, F.Roubellat, “Characterization of a set of schedules in a job shop with sequence-dependent setup times”, CESA'96 IMACS Multiconference, Computational Engineering in Systems Applications, Symposium on Discrete Events and Manufacturing Systems, Lille, 455-460, 1996.

[Artigues et Roubellat, 1997a] C.Artigues , F.Roubellat, “Un algorithme polynomial d'insertion d'une opération dans un ordonnancement cumulatif multi-ressource”, Rapport LAAS N97002, 1997.

[Artigues et Roubellat, 1997b] C.Artigues , F.Roubellat, “An operation insertion procedure in a multi-resource schedule based on dominance rules”, International Conference on Industrial Engineering and Production Management, Lyon, 304-313, 1997

[Artigues et Roubellat, 1997c] C. Artigues, F. Roubellat, “Reinsertion principles for multi-resource shop scheduling with sequence-dependent setup times”, IFAC/IFIP Conference on Management and Control of Production and Logistics (MCPL'97), Campinas (Brésil), 365-371, 1997.

[Artigues et Roubellat, 1997d] C. Artigues, F. Roubellat, “Multi-resource shop scheduling with sequence-dependent setup times”, Rapport LAAS No97251, 3rd Workshop on Models and Algorithms for Planning and Scheduling Problems, Cambridge, 1997.

[Artigues et Roubellat, 1997e] C. Artigues, F. Roubellat, “Aide à la décision pour l'ordonnancement d'atelier en présence de temps de préparation dépendant de la séquence”, 2ème Congrès International Franco-Québecois, Le Génie Industriel dans un Monde sans Frontières, actex sur CD-Rom, Albi , 1997.

[Assad *et all*, 1995] A. A. Assad, M. O. Ball, R W Dahl, “Sleeping beauties: scheduling the production of mattresses with sequence-dependent set ups”, proceedings, Symposium on Emerging Technology and Factory Automation, INRIA/IEEE, 229-239, 1995.

[Baker, 1974] K. R. Baker, “Introduction to sequencing and scheduling”,Wiley, New-York, 1974.

[Baker, 1990] K. R. Baker, “Scheduling groups of job in the two-machine flow shop”, Journal of Mathematical and Computer Modelling”, 13, 29-36, 1990.

[Baptiste, 1985] P. Baptiste, “Contribution à la conception d'un atelier flexible : définition de la base de données techniques, ordonnancement de tâches à temps de réglage variables”, Thèse de doctorat, Institut National des Sciences Appliquées de Lyon, 1985.

[Baptiste et Le Pape, 1997] P. Baptiste, C. Le Pape, “Adjustments of release and due dates for cumulative scheduling problems”, à paraître dans International Conference on Industial Engineering and Production Management, Lyon, 1997.

[Bard, 1988] J.F. Bard, “A heuristic for minimising the number of tool switches on a flexible machine”, IIE transactions, 20(4), 382-391, 1988.

[Barnes et Vanston, 1981] J.W. Barnes, L.K Vanston, “Scheduling jobs with linear delay penalties ans sequence dependent setup costs”, Operations Research, 29(1), 146-160, 1981.

[Beldiceanu *et al.*, 1996] N. Beldiceanu, E. Bourreau, D. Rivreau, H. Simonis, "Solving resource-constrained project scheduling with CHIP", proceedings 5th International Workshop on Project Management and Scheduling, EURO PMS, (Poznan, Poland), 35-38, 1996.

[Bensana, 1987] E. Bensana, "Utilisation de techniques d'intelligence artificielle pour l'ordonnancement d'atelier", Thèse de Doctorat, Ecole Nationale Superieure de l'Aeronautique et de l'Espace, Toulouse, 1987.

[Berge, 1980] C. Berge, "Les graphes d'intervalles", Regards sur La Théorie des Graphes, actes du colloque Cerisy, presses polytechniques Romandes, 1980, 1-9.

[Bernardo et Lin, 1994] J.J. Bernardo, K.-S. Lin, "An interactive Procedure for Bi-Criteria PRoduction Scheduling", Computers and Operations Research, 21, 677-688, 1994.

[Billaut, 1993] J. C. Billaut, "Prise en compte des ressources multiples et des temps de préparation dans les problèmes d'ordonnancement en temps réel",Thèse de doctorat, Université Paul Sabatier, Toulouse, 1993.

[Billaut *et al.*, 1996] J. C. Billaut, F. Roubellat, C.Artigues, "Aide à la décision pour la prise en compte d'aléas en ordonnancement d'atelier", 5ème Congrès International de Génie Industriel (GI5), pp.129-139, Grenoble, 1996.

[Billaut *et al.*, 1997] J.C. Billaut, F. Roubellat, C.Artigues, "Significant states and decision making for real time workshop scheduling in a multiple resource context", Rapport LAAS No97107, 7th Mini Euro Conference "Decision Support Systems Groupware Multimedia Electronic Commerce", Bruges, 1997.

[Billaut et Roubellat, 1996] J.C. Billaut, F. Roubellat, "A new method for workshop real time scheduling", International Journal of Production Research, 34(6), 1555-1579, 1996.

[Billaut et Roubellat, 1997] J.C. Billaut, F. Roubellat, "Un système interactif pour l'ordonnancement en temps réel d'atelier", dans "Concepts et Outils pour les systèmes de production", Cépaduès, 133-160, 1997.

[Bitran et Gilbert, 1990] "Sequencing production on parallel machines with two magnitudes of sequence-dependent setup cost", Journal of Manufacturing and Operations Management, 3, 24-52, 1990.

[Blazewicz *et al.*, 1996] Blazewicz J., Ecker K.H., Schmidt G., Weglarz J., "Scheduling in Computer and Manufacturing Systems", Second, revised edition, Springer-Verlag.

[Boctor, 1996] F. Boctor,"Resource-constrained project scheduling by simulated annealing", Int. J. Prod. Res., 34(8), 2335-2351, 1996

[Botta et Guinet, 1996] V. Botta, A. Guinet, "Scheduling flowshops with precedence constraints and time lags", Workshop on Production Planning and Control, Mons, 16-19, Septembre 1996.

[Botta, 1996] V. Botta, "Planification et ordonnancement d'une succession d'ateliers avec contraintes", thèse de doctorat, Université Claude Bernard - Lyon 1, 1996.

[Bovet et Petrini, 1980] D. P. Bovet, M. Petrini, “Evaluation of scheduling algorithms for resources with high set-up time”, Europpean Journal of Operational Research (5), 182-192, 1980.

[Brucker et Schlie, 1990] P. Brucker, R. Schlie, “Job-shop scheduling with multi-purpose machines”, Computing 45, 369-375.

[Brucker et Thiele] P. Brucker, O. Thiele “A branch and bound method for the general-shop problem with sequence dependent setup-times”, To appear in: OR Spektrum.

[Bruno et Downey, 1978] J. Bruno, P. Downey, “Complexity of task sequencing with deadlines, set-up times and changeover costs”, SIAM Journal Computing, 7(4), 393-404, 1978.

[Burstall, 1966] R. M. Burstall, 1966, “A heuristic method for a job-scheduling problem”, Operational Research Quarterly, 18, 75-82, 1966.

[Buzacott et Dutta, 1971] J. A. Buzacott, S. K. DUTTA, “Sequencing many jobs on multi-purpose facility”, Naval Research Logistics Quarterly, 18, 75-82, 1971.

[Camalot et Esquirol, 1997] J.-P. Camalot, P. Esquirol, “Aide à la décision et à la négociation dans un problème de gestion de production distribuée”, 2ème Congrès International Franco-Québécois, Le génie Industriel dans un Monde sans Frontières, Albi, 1997.

[Cao et Bedworth, 1992] J. Cao, D. D. Bedworth, “Flow shop scheduling in serial multi-product processes with transfer and set-up times”, International Journal of Production Research, 30, 1819-1830, 1992.

[Carlier, 1982] J. Carlier, “The one-machine sequencing problem”, European Journal of Operations Research, 11, 42-47, 1982.

[Carlier et Pinson, 1989] J. Carlier, E. Pinson, “An algorithm for solving the job-shop problem” MAnagement Science, 35, 2, 164-176, 1989.

[Caseau et Laburthe, 1995] Y. Caseau, F. Laburthe, “Improving branch and bound for job-shop scheduling with constraint propagation”, In M. Deza, R. Euler, et Y. Manoussakis, editeurs, Proc. of the 8th Franco-Japanese-4th Franco-Chinese Conference of Combinatorics and Computer Science, Brest, 1995.

[Cattrysse *et al.*, 1993] D. Cattrysse, M. Salomon, R. Kuik, L. N. Van Wassenhove, “A dual ascent and column generation heuristic for the discrete lotsizing and scheduling problem with setup times”, Management Science, 39(4), 477-486, 1993.

[Cetinkaya, 1994] F. C. Cetinkaya, “Lot Streaming in a two-stage flow shop with set-up, processing and removal times separated”, Journal of the Operational Research Society, 45(12), 1445-1455, 1994.

[Chen, 1993] Z. L. Chen, “A better heuristic for preemprive parallel machine scheduling with batch setup times”, SIAM Journal on Computing, 22, 1303-1318, 1993.

[Chen, 1997] Z. L. Chen,“Scheduling with batch setup times and earliness-tardiness penalties”,European Journal of Operational Research 96, 518-537, 1997.

[Cheng et chen, 1994] T. C. E. Cheng, Z. L. Chen, "Parallel machine scheduling with batch setup time", Operations Research, 42, 1171-1174, 1994.

[Cho, 1989] C. H. Cho, "Structuration des données et caractérisation des ordonnancements admissibles des systèmes de production", Thèse de doctorat, Institut National des Sciences Appliquées de Lyon, 1989.

[Coffman *et al.*, 1989] E.G. Coffman, J. A. Nozari, M. Yannakakis, "Optimal scheduling of products with two subassemblies on a single machine", Operations Research, 37, 326-336, 1989.

[Collinot *et al.*, 1988] A. Collinot, C. Le Pape, G. Pinoteau, "SONIA: a knowledge-based scheduling system", Artificial Intelligence in Engineering, 3, 86-94, 1988.

[Conway *et al.*, 1967] R. W. Conway, W. L. Maxwell, L. W. Miller, "Theory of scheduling", Reading, MA, Addison-wesley Publishing Company, 1967.

[Corwin et Esogbue, 1974] B. D. Corwin, A.O. Esogbue, "Two machine flowshop scheduling problems with sequence dependent setup times: a dynamic programming approach", Naval Research Logistics Quarterly, 21, 515-524, 1974.

[Crama *et al.*, 1994] Y. Crama, A.W.J. Kolen, A.G. Oerlemans, F.C.R. Spieksma, "Minimizing the number of tool switches on a flexible machine", International Journal of Flexible Manufacturing Systems, 6(1), 33-54, 1994.

[Crouhy, 1983] M. Crouhy, "La gestion informatique de la production industrielle", Editions de l'usine édition, 1983.

[Daniels et Mazzola, 1993] , R. L. Daniels, J. B. Mazzola, "A tabu-search heuristic for the flexible-resource flow shop scheduling problem", Annals of Operations Research, 41, 207-230, 1993.

[Daoud et Purcheck, 1981] Z. A. Daoud, G. F. K. Purcheck, "Multi-tool job sequencing for tool change reduction", International Journal of production Research, 19(4), 425-435, 1981.

[Das *et al.*, 1995] S. R. Das, J. N. D. Gupta, B. M. Khumawala, "A saving index heuristic algorithm for flowshop scheduling with sequence dependent set-up times", Journal of the Operational Research Society, 46, 1365-1373, 1995.

[Dauzère Péres et al, 1996] S. Dauzère Péres, W. Roux, J.B. Lasserre, "Multi-resource shop scheduling with resource flexibility", rapport LAAS 96178, 1996.

[Deane et White, 1975] R. H. Deane, E. R. White, "Balancing workloads and Minimizing Set-up Costs in the Parallel Processing Shop", Operational Research Quaterly, 26(1), 45-53, 1975.

[Demeulemeester *et al.*, 1996] Demeulemeester E., De Reyck B., Herroelen W., "Optimal and suboptimal procedures for the discrete time/resource trade-off problem in project networks", 5th International Workshop on Project Management and Scheduling, April 11-13, , Poznan,1996.

[Demeulemeester et Herroelen, 1992] E. Demeulemeester and W. S. Herroelen, "A branch-and-bound procedure for the multiple resource-constrain ed project scheduling problem", Management Science, 38, 1803-1818.

[Demmou, 1977] R. Demmou, “Etudes de familles remarquables d’ordonnancements, en vue d’une aide à la décision”, Thèse de Docteur Ingénieur, Université Paul Sabatier, Toulouse, 1977.

[Desrochers *et al.*, 1990] M. Desrochers, J. K. Lentra, M. W. P. Savelsberg, “A classification schema for vehicule routing and scheduling problems”, European Journal of Operational Research, 1990, 46, 322-332, 1990.

[De Terssac *et al*, 1996] G. De Terssac, J. Erschler, I. Bazet, M. J. Huguet, “De l’analyse des décisions en situation de coopération à leur modélisation : approche par contraintes”, rapport LAAS 96367.

[Dietrich, 1989] B. L. Dietrich, “A 2-phase heuristic for scheduling parallel unrelated machines with set-ups”, Report RC 14330. IBM US Research Center, 1989.

[Dobson *et al.*, 1987] G. Dobson, U.S. Karmarkar, J.L. Rummel, “Batching to minimize flow times on parallel heterogeneous machines”, Management Science, 35, 607-613, 1989.

[Dobson *et al.*, 1989] G. Dobson, U.S. Karmarkar, J.L. Rummel, “Batching to minimize flow times on one machine”, Management Science, 33, 784-799, 1987.

[Driscoll et Emmons, 1977] W. C. Driscoll, H. Emmons“Scheduling production on one machine with changeover costs”, AIIE Transactions, 9 (4), 388-395, 1977.

[Echalier, 1991] F. Echalier, “Problèmes d’ordonnancement sur machines parallèles : apport du recuit simulé”, Thèse de doctorat, Université Claude Bernard - Lyon I, 1991.

[Egbelu, 1991] P. J. Egbelu, “Batch production time in a multi-stage system with material-handling consideration”, International Journal of Production Research, 29, 4, 695-712, 1991.

[Elmaghraby et Guinet, 1992] S. E. Elmaghraby, A. Guinet, “Scheduling on parallel machines : an alternative approach”, 3rd International Workshop on Project Management and Scheduling”,Como, 1992.

[Elmaghraby et Karnoub, 1995] S. E. Elmaghraby, R. E. Karnoub “Production control in flexible flowshops : an example from textile manufacturing”, OR Report n305, 1995.

[Erschler, 1976] J. Erschler, “Analyse sous contraintes et aide à la décision pour certains problèmes d’ordonnancement”, Thèse de Doctorat D’Etat, Université Paul Sabatier, Toulouse, 1976.

[Erschler *et al.*, 1976] J. Erschler, F. Roubellat, J.P. Vernhes, “Finding some essential characteristics of the feasible solutions for a scheduling problem”, Operations Research, 24(4), 1976.

[Erschler *et al.*, 1982] J. Erschler, G. Fontan, C. Mercé, F. Roubellat, “Applying new dominance concepts to job schedule optimization”, European Journal of Operational Research, 11, 60-66, 1982.

[Erschler *et al.*, 1992] J. Erschler, G. Fontan, F. Roubellat, chapitre “Ordonnancement en ateliers spécialisés”, “Encyclopédie du Management, vol II”, 208-229, Helfer et Orsoni, Vuibert, Paris, 1992.

[Esquirol, 1987] P. Esquirol, “Règles et processus d’inférence pour l’aide à l’ordonnancement de tâches en présence de contraintes”, Thèse de doctorat, Université Paul Sabatier, Toulouse, 1987.

[Esquirol et Lopez, 1997] P. Esquirol, P. Lopez, “Analyse sous contraintes en ordonnancement”, dans “Concepts et Outils pour les systèmes de production”, Cépaduès, 133-160, 1997.

[Farn et Muhlemann, 1979] C.K. Farn, A.P. Muhlemann “The dynamic aspects of a production scheduling problem”, International Journal of Production Research, 17, 15, 1979.

[Fleischmann, 1990] B. Fleishmann, “The discrete lotsizing and scheduling problem”, European Journal of Operational Research, 44(3), 337-348, 1990.

[Fleishmann, 1994] B. Fleishmann, “The discrete lot-sizing and scheduling problem with sequence-dependent setup costs”, European Journal of Operational Research, 75(2), 395-404, 1994.

[Flipo, 1997] C. Flipo, “Scheduling unrelated parallel machines for an industrial production-distribution problem”, IFAC/IFIP Conference on Management and Control of Production and Logistics, Campinas, SP, Brazil, 291-296, 1997.

[Flynn, 1987] B. B. Flynn, “Repetitive lots: the use of sequence-dependent set-up time scheduling procedure in group technology and traditional shop”, Journal of Operations Management, 7 (1,2), 203-216, 1987.

[Fontan, 1980] G. Fontan, “Notion de dominance et application à l’étude de certains problèmes d’ordonnancement”, Thèse de Doctorat d’Etat, Université Paul Sabatier, Toulouse, 1980.

[França *et al.*, 1995] P. M. França, M. Gendreau, G. Laporte, F. M. Müller, “A Tabu search Heuristic for the multiprocessor Scheduling Problem with sequence dependent setup times”, Research report, CRT-95-16, Montreal, 1995.

[Frederickson *et al*, 1978] G. Frederickson, M. S. Hecht, C. E. Kim, “Approximation Algorithm for some routing problems”, SIAM journal on Computing, 7, 178-193, 1978

[Frendewey et Sumichrast, 1988] “Scheduling parallel processors with setup costs and resource limitation”, Decision Sciences, 19, 138-146, 1988.

[Gavett, 1965] J.W. Gavett, “Three heuristic rules for sequencing jobs to a single production facility”,Management Science, 11(8), 166-176, 1966.

[Geoffrion et Graves, 1976] A. M. Geoffrion, G. W. Graves, “Scheduling parallel production lines with changeover Costs: practical application of a quadratic assignment/LP approach”, Operations Research, 24(4), 595-610, 1976.

[Ghosh, 1994] “Batch scheduling to minimize total completion time”, Operation Research Letters, 16, 271-275

[Giard, 1988] V. Giard, “Gestion de la production”, 2ème édition, Economica, 1988

[Giffler et Thompson, 1960] B. Giffler, G.L. Thompson, “Algorithms for solving production scheduling problems”, Operations Research, 8(4), 487-503, 1960.

[Glassey, 1968] "Minimum changeover scheduling of several products on one machine", Operations Research, 16(2), 432-352, 1968.

[GOThA, 1993] GOThA, Groupe de Recherche en Ordonnancement théorique et Appliqué (J. Carlier, P. Chrétienne, J. Ershler, C. Hanen, P. Lopez, A. Munier, E. Pinson, M.-C. Portmann, C. Prins, C. Proust et P. Villon), "Les problèmes d'ordonnancement", RAIRO RO, 27 (1), 77-150, 1993.

[Guinet, 1991] A. Guinet,"Textile Production Systems: a succession of Non-identical Parallel Processor Shops", Journal of Operational Research Society, 42 (8), 665-671, 1991.

[Guinet, 1993] A. Guinet, "Scheduling sequence-dependent jobs on identical parallel machines to minimize maximum completion time criteria", International Journal of Production Research, 31(7), 1579-1594, 1993.

[Guinet, 1995] A. Guinet, "Scheduling Independent jobs on uniform parallel machines to minimize tardiness criteria",Journal of Intelligent Manufacturing, 6, 95-103, 1995.

[Gupta, 1975] J. N. D. Gupta, 1975, "A search algorithm for the generalized flowshop scheduling problem",Computers and Operations Research, 2, 83-90, 1975.

[Gupta, 1984] J. N. D. Gupta "Optimal Schedules for single facility with two job classes", Computers and Operations Research, 11, 409-413, 1984.

[Gupta, 1982] S. K. Gupta, " N jobs and m machines job-shop problems with sequence-dependent set-up times", International Journal of Production Research, 20 (5), 643-656, 1982.

[Gupta, 1986a] J. N. D. Gupta, "Flowshop schedules with sequence dependent setup times", Journal of the Operations Research Society of Japan, 29, 206-219, 1986.

[Gupta, 1988] J. N. D. Gupta, "Single facility scheduling with multiple job classes", European Journal of Operational Research, 33, 42-45, 1988.

[Gupta *et al.*, 1995] J. N. D. Gupta, R. Das, S. Ghosh, "Sequence-dependent flowshop scheduling via branch and bound",Unpublished paper, Department of Management, Ball State University, Muncie, IN 47306, USA, 1995.

[Gupta et Darrow, 1986] J. N. D. Gupta, W.P. Darrow, "The two-machine sequence-dependent flowshop scheduling problem", European Journal of Operationns Research, 24, 439-446, 1986.

[Gupta et Tunc, 1994] J. N. D. Gupta, E. A. Tunc, "Scheduling a two-stage hybrid flowshop with separable setup and removal times", European Journal of Operational Research, 77, 415-428, 1994.

[Ham *et al.*, 1985] I. Ham, K. Hitomi, T. Yoshida, "Group technology: Application to production Management", Kluwer-Nijhoff, Boston, 1985.

[Han et Dejax, 1991] "Une nouvelle heuristique pour le problème d'ordonnancement de type flow-shop avec la prise en compte des temps de montage et démontage d'outils et la présence de machines goulots", 3ème Congrès International de Génie Industriel, Tours, 1991.

[Hansmann et Höck, 1995] K. W. Hansmann, M. Höck, "Application of new heuristics to scheduling with sequence-dependent setup times", Working paper 2, Institut Für Industriebetriebslehre und Organisation, Hamburg, 1995.

[Haudot, 1996] L. Haudot, "Une approche orientée utilisateur pour la conception de systèmes coopératifs en ordonnancement de la production", Thèse de Doctorat, Institut National des Sciences Appliquées, Toulouse, 1996.

[Hayes et Wheelwright, 1979] R. H. Hayes et S. C. Wheelwright, "Le cycle de vie du processus de production (i) et (ii)", Harvard-L'expansion, 23-32, 99-110, été et automne 1979.

[Hershauer, 1970] J. C. Hershauer, "An empirically-derived simulation to explore sequencing decisions", Unpublished PhD thesis, Indiana University.

[Hertz et Widmer, 1995] A. Hertz, M. Widmer, "La méthode TABOU appliquée aux problèmes d'ordonnancement", RAIRO APII 29(4-5), 253-378, 1995.

[Hétreux, 1996] G. Hétreux, "Structures de Décision multi-niveaux pour la planification de la production : robustesse et cohérence des décisions", Thèse de Doctorat, Institut National des Sciences Appliquées, Toulouse, 1996.

[Hu *et al.*, 1987] T. C. Hu, Y. S. Kuo, F. Ruskey, "Some optimum algorithms for scheduling problems with changeover costs", Operations Research, 16(2), 342-352, 1987.

[Huguet, 1994] M.-J. Huguet, "Approches par contraintes pour l'aide à la décision et à la coopération en gestion de production", Thèse de Doctorat, Institut National des Sciences Appliquées, Toulouse, 1994.

[Jackson, 55] J. R. Jackson, "Scheduling a production line to minimize maximum tardiness", Management Science Research Project, UCLA, 1955.

[Johnson, 1954] S. M. Johnson,"Optimal two-and three-stage production schedules with setup times included", Naval Research Logistics Quarterly, 1, 61-68, 1954.

[Jordan et Drexl, 1994] C. Jordan, A. Drexl, "Lotsizing and scheduling by batch sequencing", working paper #343, Christian-Albrecht-Universität zu Kiel, Kiel, 1994.

[Kaplan, 1991] L. Kaplan, "Resource-constrained project scheduling with setup times", Working Paper, Departement of Management Science, University of Tennesee, Knoxville, USA, 1991.

[Kedad. 1997] S. Kedad, "Résolution de problèmes de partitionnement généralisé par des méthodes d'optimisation globale à base de déplacements stochastiques: application à l'ordonnancement à machines parallèles", Thèse de Doctorat, Ecole Centrale de Paris, 1997.

[Kedad *et al.*, 1995] S. Kedad, C. Lecomte, P. Dejax, "Une heuristique en deux phases pour la résolution d'un problème d'ordonnancement à machines parallèles", GI5 - 5ème congrès international de Génie Industriel, Grenoble, (1), 97-103, 1995.

[Kelley, 1963] J. E. Kelley, "The critical path method: resource planning and scheduling". In Industrial Scheduling, J. F. Muth and G. L. Thompson (eds) (Englewood Cliffs, NJ: Prentice-HAll), pp 347-365

[Kim et Bobrowski, 1994] S. C. Kim, P. M. Bobrowski, "Impact of sequence-dependent setup time on job shop scheduling performance", International Journal of Production Research, 32 (7), 1503-1520, 1994.

[Kim et Bobrowski, 1995] S. C. Kim, P. M. Bobrowski, " Evaluating order release mechanisms in a job-shop with sequence-dependent setup times", Production and Operations Management, 4 (2), 163-180, 1995.

[Kirby et Scobey, 1970] M. J. L. Kirby, P. F. Scobey, "Production Scheduling on n identical machines", Canadian Operations Research Society Journal, 8, 14-27.

[Kolish, 1995] R. Kolish, "Project scheduling under resource constraints", Physica-Verlag, 1995.

[Kolish et Drexl, 1996] R. Kolish, A. Drexl, "Local Search for Nonpreemptive Multi-Mode Resource-Constrained Project Scheduling", Research Report, Institut fur Betriebswirtschaftslehre, Christian-Albrechts-Universitat zu kiel, 1996.

[Kolish et Sprecher, 1996] R. Kolish, A. Sprecher, "PSPLIB - A project scheduling problem library", Research Report 396, Institut fur Betriebswirtschaftslehre, Christian-Albrechts-Universitat zu kiel, 1996.

[Ivens et Lambrecht, 1994] "Extending the shifting bottleneck procedure to real-life scheduling applications". In Fourth International Workshop on Project Management and Scheduling, 210-213, Leuven, Belgium, 12-15 juillet 1994.

[Lamothe, 1996] J. Lamothe, "Une approche pour l'ordonnancement dynamique d'un atelier de traitement de surface", Thèse de Doctorat, Ecole Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, 1996.

[Laurent, 1997] R. Laurent, "Gérer la réactivité commerciale et les contraintes de production", conférence au salon "solutions GPAO 1997", Paris, mars 1997.

[Lawler *et al.*, 1985] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy kan, D. B. Shmoys, "The Traveling Salesman Problem", John Wiley, New York, 1985.

[Lawler *et al*, 1989] E. L. Lawler, J. L. Lenstra, A. H. G. Rinnooy Kan, D. B. Schmoys, "Sequencing and scheduling: algorithms and complexity", Rapport BS-R8909, Centre for Mahematics and Computer Science, Amsterdam, 1989.

[Lecomte, 93] C. Lecomte, "Un système à base de règles d'aide à l'ordonnancement d'un atelier travaillant à la commande", Thèse de doctorat, Ecole Centrale de Paris, 1993

[Lee et Kim, 1993] Y. H. Lee, S. Kim, "Neural Network Application for scheduling jobs on parallel machines", Computers and Industrial Engineering, 25, pp. 227-230, 1993.

[Lee et Kim, 1996] J-K. Lee, T-D. Kim, "Search Heuristics for Resource Constrained Project Scheduling", Journal of the Operational Research Society, 47, 678-689, 1996.

[Legait, 1992] A. Legait, "L'intégration : quels impacts sur la spécification d'un système de gestion de production assistée par ordinateur?", Thèse de Doctorat, Université de Lyon I-INSA, 1992.

[Le Gall, 1989] Le Gall A., "Un système interactif d'aide a la décision pour l'ordonnancement et le pilotage en temps réel d'atelier", Thèse de doctorat, Université Paul Sabatier, Toulouse, 1989.

[Le Moal et Tarondeau, 1979] P. Le Moal, J. C. Tarondeau, "Un défi à la fonction de production", Revue Française de Gestion, 9-14, Janvier-Février 1979.

[Lepape, 88] C. Lepape, "Des systèmes d'ordonnancement flexibles et opportunistes", Thèse de Doctorat en Sciences, Université de Paris-Sud, centre d'Orsay, 1988.

[Levner *et al.*, 1995] E. Levner, K. Kogan, O. Maimon, "Flowshop scheduling of robotic Cells with job-dependent transportation and set-up effects", Journal of the Operational Research Society, 46, 1447-1455, 1995.

[Levy, 1996] M.-L. Levy "Méthodes par décomposition temporelle et problemes d'ordonnancement", Thèse de Doctorat, Institut National Polytechnique de Toulouse, 1996.

[Li, 1996] S. Li, "A hybrid two-stage flowshop with part family, batch production, major and minor setups", à paraître dans European Journal of Operations Research.

[Lockett et Muhlemann, 1972] A. G. Lockett, A. P. Muhlemann, "A scheduling problem involving sequence dependent changeover times", Operations Research, 20, 895-902, 1972.

[Logendran et Sriskandarajah, 1993] R. Logendran, C. Sriskandarajah, "Two-machine group scheduling problem with blocking and anticipatory setups",European Journal of Operational Research, 69, 467-481, 1993.

[Magnanti et Vachani, 1990] T. L. Magnanti, R. Vachani, "A strong cutting-plane algorithm for production scheduling with changeover costs", Operations Research, 38(3), 456-473, 1990.

[Manier et Baptiste, 1994] M. A. Manier, P. Baptiste, "A survey, the hoist scheduling problem", APII, 28(1), 7-35, 1994.

[Marsh et Montgomery, 1973] J. D. Marsh, D.C. Montgomery, "Optimal procedures for scheduling jobs with sequence-dependent changeover times on parallel processors", AIIE Technical Papers, 279-286., 1973.

[Mason et Anderson, 1991] "Minimizing flow time on a single machine with job classes and setup times", Naval Research Logistics, 38, 333-350, 1991.

[Mitsumori, 1972] S. Mitsumori, "Optimal production schedules on multi-commodity in flow line", IEEE Transactions on Systems, Man and Cybernetics, 2(4),86-493, 1972.

[Monma et Potts, 1989] C. L. Monma, C. N. Potts, "On the complexity of scheduling with batch setup times", Operations Research, 37(5), 798-804, 1989.

[Monma et Potts, 1993] C. L. Monma, C. N. Potts, "Analysis of Heuristics for preemptive parallel machine scheduling with batch setup times", Operations Research, 41(5), 981-993, 1993.

[Nabeshima et Maruyama, 1983] I. Nabeshima, S. Maruyama, "Note on the two machine flowshop scheduling problem with separated setup and clean-up times, times lags and transportation times",Rep. Univ. Electro-Comm, 34(1) 29-31 (1983).

[Ovacik et Uzsoy, 1993] I.M. Ovacik, R. Uzsoy, "Worst-case error bounds for parallel machine scheduling problems with bounded sequence-dependent setup times",Operations Research Letters, 14, 251-256, 1993.

[Ovacik et Uzsoy, 1994a] I.M. Ovacik, R. Uzsoy, "Rolling Horizon procedures for a single-machine dynamic scheduling problem with sequence-dependent setup times",International Journal of Production Research, 32, 1243-1263, 1994.

[Ovacik et Uzsoy, 1994b] I.M. Ovacik, R. Uzsoy, "Exploiting shop floor status information to schedule complex job shops",Journal of Manufacturing Systems, 13, 73-84, 1994.

[Ovacik et Uzsoy, 1995] I.M. Ovacik, R. Uzsoy, "Rolling Horizon procedures for dynamic parallel machine scheduling with sequence-dependent setup times",International Journal of Production Research, 33(11), 3173-3192, 1995.

[Parker *et al.*, 1977] , R. G. Parker, R. H. Deane, R. A. Holmes, "On the Use of a Vehicle Routing Algorithm for the Parallel Processor Problems with Sequence Dependent Changeover Costs", AIIE Transactions, 9(12), 155-160.

[Parthasarathy et Rajendran, 1997] S. Parthasarathy, C. Rajendran, "A simulated annealing heuristic for scheduling to minimize mean weighted tardiness in a flowshop with sequence-dependent setup times of jobs-a case study",Production Planning and Control, 8 (5),475-483, 1997.

[Pellet, 85] X. Pellet, "Sur la hiérarchisation des décisions. Application à la conduite d'ateliers", Thèse de Doctorat, Institut National Polytechnique de Grenoble, 1985.

[Picart et Queyranne, 1977] J. C. Picard, M. Queyranne, "The Time-dependent Traveling Salesman Problem and Its Application to the tardiness Problem in One-Machine Scheduling", Operations Research, 26 (1), 86-110, 1977.

[Pinedo, 1995] M. Pinedo, "Scheduling: theory, algorithms and systems", Partie I: "Deterministic models", Prentice Hall, Englwood Cliffs, 1995.

[Pinson et al., 1994] E. Pinson, C. Prins ,F. Ruller , "Using Tabu Search for Solving the Resource-Constrained Project scheduling problem", proceedings 4th International Workshop on Project Management and Scheduling, EURO WG-PMS, (Leuven, Belgium), 102-106, 1994.

[Portmann, 1988] M.-C. Portmann, "Méthodes de décompositions spatiale et temporelle en ordonnancement de la production", RAIRO-APII, 22(5), 439-451, 1988.

[Portmann, 1996] M.-C. Portmann, "Genetic Algorithms and Scheduling: a State of the Art and some Propositions", Workshop on Production Planning and Control, Mons, i-xxiv, 1996.

[Portmann et Bolzoni, 1994] M.-C. Portmann, V. Bolzoni, "Specific Developments or General Software Adaptation in order to solve some scheduling or production manufacturing problems?", 4th international workshop on Project Management and Scheduling, Leuven, 192-197, 1994.

[Potts, 1991] C. N. Potts, "Scheduling two job classes on a single machine", Computers and Operations Research, 18, 411-415, 1991.

[Prabhakar, 1974] , "A production scheduling problem with sequencing considerations", Management science, 13(8), 454-464, 1974.

[Presby et Wolfson, 1967] J.T. Presby, M. L. Wolfson, "An algorithm for solving job sequencing problems", Management Science, 13(8), 454-464, 1967.

[Prins, 1994] Prins Ch., "Algorithmes de graphes", Eyrolles , 1994

[Proust *et al.*, 1988] C. Proust, M. Drogou, J. M. Foucher, E. Foucheyrand, "Une heuristique pour le problème d'ordonnancement statique de type $n/m/$flowshop, avec prise en compte des temps de montage et démontage d'outils", Revue d'Automatique, d'Informatique et de Recherche Opérationnelle APII",22, 453-470, 1988.

[Proust *et al.*, 1991] C. Proust, J. N. D. Gupta, V. Deschamps, "Flowshop scheduling with set-up, processing and removal times separated", International Journal of Production Research, 29 (3), 479-493, 1991.

[Proust et Grunenberg, 1995] C. Proust, E. Grunenberg, "Planification de production dans un contexte de flow-shop hybride à deux étages: conception et interprogrammation d'ARIANE2000", RAPA, 8(5), 715-734, 1995.

[Psaraftis, 1980] "A dynamic programming approach for scheduling groups of identical jobs", Operations Research, 28, 1347-1359, 1980.

[Rajgopal et Bidanda, 1991] J. Rajgopal, B. Bidanda, "On scheduling parallel machines with two setup classes", International Journal of Production Research, 29(12), 2443-2458, 1991.

[Randhawa et Smith, 1995] S. U. Randhawa, T. A. Smith, "An experimental investigation of scheduling non-identical, parallel processors with sequence-dependent set-up times and due-dates", International Journal of Production Research, 33(1), 59-69, 1995.

[Rìos-Mercado et Bard, 1996] R. Z. Rìos-Mercado,J. F. Bard , 1996, "New Heuristics for the Flow Line Problem with Setup Costs", à paraître dans European Journal of Operational Research

[Roy et Sussman, 1964] B. Roy et B. Sussman, "Les problèmes d'ordonnancement avec contraintes disjonctives", Note DS No. 9 bis, SEMA, Paris, 1964.

[Rubin et Ragatz, 1995] P. A. Rubin, G. L. Ragatz, "Scheduling in a sequence dependent setup environment with genetic search", Computers and Operations Research, 22(1), 85-99, 1995.

[Sahney, 1972] V. K. Sahney, "Single-server, two-machine sequencing with switching times", Operations Research, 20, 26-36, 1972.

[Salomon *et al.*, 94] M. Salomon, M. M. Solomon, L. N. Van Wassenhove, Y. Dumas, S. Dauzère-pérès, "Discrete Lotsizing and scheduling with sequence dependent setup times and setup costs", Management report series no. 183, Erasmus University/Rotterdam School of Management, Rotterdam, 1994.

[Santos et França, 1997] H. C. M. Santos, P. M. França, "Scheduling with sequence-dependent setup times and early-tardy penalties", IFAC/IFIP Conference on Management and Control of Production and Logistics, Campinas, SP, Brazil, 281-286, 1997.

[Schrague et Baker, 1978] L. Schrague, K. Baker, "Dynamic programming solution of sequencing problems with precedence constraints", Operations Research, 26, 444-449, 1978.

[Schutten, 1996] M. Schutten, "Shop Floor Scheduling with Setup Times", Thèse, University of Twente, 1996.

[Schutten *et al.*, 1996] M. Schutten, S.L. van de Velde, W.H. M. Zijm, "Single-machine scheduling with release dates, due dates and family setup times", Management Science, 42(8), 1165-1174, 1996.

[Sherali *et al.*, 1990] H. D. Sherali, S. C. Sarin, M. S. Kodialam, "Models and algorithm for a two-stage production process", Production Planning and Control, 1(1), 27-39, 1990.

[Sielken, 1976] "Sequencing with set-up costs by zero-one mixed integer linear programming", AIIE transactions, 8(3), 369-371, 1976.

[So, 1990] K.C. So, "Some heuristics for scheduling jobs on parallel machines with setups",Management science, 36(4), 467-475, 1990.

[Sprecher *et al.*, 1995] A. Sprecher, R. Kolish, A. Drexl, "Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem", European Journal of Operational Research, 80, 94-102, 1995.

[Srikar et Ghosh, 1986] B.N. Srikar, S. Ghosh, A. Mild,"Model for the n-job, M stage flowshop with sequence dependent set-up times", International Fournal of Production Research, 24(6), 1459-1474, 1986.

[Stafford et Tseng, 1990] E. F. Stafford, F. T. Tseng, "On the Srikar-Ghosh MILP model for he NxM SDST flowshop problem", International Journal of Production Research, 28, 1817-1830, 1990.

[Sule, 1982] D.R Sule, "Sequencing n jobs on two machines with setup, processing and removal times separated", Naval Research Logistics Quarterly, 29, 517-519, 1983.

[Sule et Huang, 1983] D.R Sule, K.Y. Huang, "Sequency on two and three machines, with setup, processing and removal time separated", International Journal of Production Research, 21, 723-732, 1983.

[Sumichrast et Baker, 1987] R. T. Sumichrast, J. R. Baker, "Scheduling parallel processors: an integer linear programming based heuristic for minimizing setup time", International Journal of Production Research, 25(5), 761-771, 1987.

[Strusevich, 1990] , V. A. Strusevich "Two machine flow shop no-wait scheduling problem with set-up, procesing and removal times separated", Research Report 9094/A, Econometric Institute, Erasmus University, Rotterdam, Netherlands, 1990.

[Strusevich et Zwaneveld, 1991] "On non-permutation solutions to some two machine flow shop scheduling problems",Research Report 9150/A, Econometric Institute, Erasmus University, Rotterdam, Netherlands, 1990.

[Szwarc, 1983] "Flowshop problems with time lags", Management Science, 29(4), 477-481, 1983.

[Szwarc et Gupta, 1987] "A flowshop problem with sequence-dependent additive setup times",Naval Research Logistics Quarterly, 34, 619-627, 1987.

[Taha, 1971] H. A. Taha, "Sequencing by implicit ranking and zero-one polynomial programming", AIIE Transactions, 3, p 299, 1971.

[Tang, 1990] C.S. Tang, "Scheduling batches on parallel machines with setups", European Journal of Operations Research, 46, 28-37, 1990.

[Tang et Denardo, 1988] C. S. Tang, E. V. Denardo, "Models arising from a flexible manufacturing machine, Part I: minimization of the number of tool switches", Operations research, 36(5), 767-777, 1988.

[Tang et Wittrock, 1985] C. S. Tang, R. J. Wittrock, "Parallel machine scheduling with major and minor set-ups", RC 11412, IBM T. J. Watson Research Center, Yorktown Heights, New York, 1985.

[Thomas, 1980] V. Thomas, "Aide à la décision pour l'ordonnancemnt d'atelier en temps réel", Thèse de Doctorat, Université Paul Sabatier, Toulouse, 1980.

[Torres, 1996] Torres Ph., "Techniques de propagation de contraintes et problèmes d'ordonnancement", Rapport de Diplôme d'Etudes Approfondies, Institut National des Sciences Appliquées, Toulouse, 1996.

[Ulusoy et Ozdamar, 1994] G. Ulusoy, L. Ozdamar, "A constraint based perspective in resource constrained project scheduling", International Journal of Production Research, 32(3), 693-705, 1994.

[Uskup et Smith, 1975] E. Uskup, S. B. Smith, "A branch-and-bound algorithm for two-stage production-sequencing problems", Operations research,23(1), 118-136, 1975.

[Vaessens et al. 96] R.J.M. Vaessens, E.H.L. Aarts, J.K. Lenstra, "Job-shop scheduling by local search", INFORMS Journal on Computing, 8, 302-317, 1996.

[Valette, 1995] R. Valette, "Les Réseaux de Petri", Cours de l'ENAC, 1995.

[Van Laarhoven *et al.*, 1992] P. J. M. Van Laarhoven, E. H. L. Aarts, J. K. Lenstra, "Job-shop scheduling by simulated annealing", Operations Research, 40, 113-125, 1992

[Vignier, 1997] A. Vignier, "Contribution à la résolution des problèmes d'ordonnancement de type monogamme, multimachine ("Flow-shop hybride")", Thèse de Doctorat, Université de Tours, 1997.

[Villaumié, 1997a] Cabinet Villaumié SA, "ORDO^R Ordonnancement Temps Réel d'Atelier", présentation commerciale du logiciel, 1997.

[Villaumié, 1997b] Cabinet Villaumié SA, "Management Dynamique des Flux", présentation commerciale, 1997.

[Weeda, 1978] P. J. Weeda, A dynamic programming formulation for the one machine sequencing problem", European Journal of Operational Research, 2, p 298-300, 1978.

[White et Wilson, 1977] C.H. White, R.C. Wilson, “Sequence-dependent set-up times and job sequencing”, International Journal of Production Research, 15 (2), 191-202, 1977.

[Widmer, 91] M. Widmer, “Job shop scheduling with tooling constraints: a Tabu search approach”, Journal of the Operational Research Society, 42(1), 75-82, 1991.

[Wilbrecht et Prescott, 1969] “The influence of set-up time on job shop performance”, Management Science, 16(4), 274-280, 1969.

[Wittrock, 1990] R. J. Wittrock, “Scheduling parallel machines with setups”, International Journal of Flexible Manufacturing systems, 2, 329-341.

[Yoshida et Hitomi, 1979] T. Yoshida, K. Hitomi, “Optimal two stage production scheduling with setup times separated”, AIIE transactions, 11, 262-263, 1979.

[Zouhri, 1993] M. Zouhri, “Représentation analytique de familles d'ordonnancements : Apport des arbres PQR”, thèse de doctorat, Institut National des Sciences Appliquées de Lyon, 1993.

Liste des Figures

Index