LE MRP : MATERIAL REQUIREMENTS PLANNING

La planification de la production consiste à définir le niveau global des unités à produire pour satisfaire au mieux les commandes fermes et prévues, tout en respectant les contraintes de capacité, les objectifs généraux de rentabilité, de productivité, de délais de livraison, etc. Ainsi, la planification de la production vise à optimiser l'utilisation des ressources disponibles pour la fabrication d'un ou plusieurs produits.

Il n'existe pas de modèle général d'une telle optimisation, mais certaines approches de ce problème, sans réelles perspectives d'optimisation eu égard à sa complexité. Le modèle de planification probablement le plus répandu est le MRP qui, dans son acception originelle, signifie Material Requirements Planning, c'est-à-dire planification des besoins en composants. Le MRP est une méthode de coordination de la production dans un système productif à multiples étages incorporant plusieurs produits, composants et matières premières. Ainsi, le MRP est une méthode conçue pour gérer la fabrication de produits complexes, i.e. constitués de plusieurs composants.

Dans ce type de système de production, les produits finis sont divisés en sous-ensembles, décomposés à leur tour en groupes, puis en sous-groupes jusqu'au stade final de pièces ou matières achetées. Cette conception modulaire conduit à définir un produit complexe au moyen d'une nomenclature arborescente à plusieurs niveaux auxquels correspondent les divers stades de la décomposition. Concrètement, ces niveaux de décomposition représentent des niveaux de stockage ; aussi un tel système est-il caractérisé par l'existence de stocks multi-échelon.

Ces stocks se subdivisent principalement en deux catégories : les stocks de distribution et les stocks de fabrication. Les articles constituant les stocks de distribution sont des produits finis qui, par définition, font l'objet d'une demande externe appelée demande indépendante. Dans les stocks de fabrication, on trouve les articles qui entrent dans la composition des produits finis. La demande pour ces articles se déduit de celle en produits finis ; à ce titre, elle est qualifiée de demande dépendante. Il en résulte que seules les demandes indépendantes font l'objet de prévisions statistiques.

A partir de ces prévisions, des disponibilités de matières et des ressources, un programme directeur de production est établi. Ce programme de fabrication exprime les décisions de production en articles finis sur un certain nombre de périodes futures, en réponse à une demande prévisionnelle. Connaissant la composition des produits, leurs caractéristiques techniques (délai,...) et le programme directeur de production, il est possible de déterminer les quantités exactes de chacun des composants dont il convient d'assurer l'approvisionnement. Une planification des besoins en composants est ainsi réalisée.

Dans sa version originelle, le MRP, que nous appellerons désormais MRP1, consiste donc à établir un programme directeur de production, à calculer les quantités exactes des composants requis et à définir les lots d'approvisionnement. La détermination de la taille des lots résulte normalement d'un arbitrage entre le coût fixe de production ou de commande, et le coût de stockage. Plusieurs modèles d'approvisionnement existent, mais nous aborderons cette question dans le chapitre 2. Aussi supposerons-nous dans ce chapitre que la taille des lots est fixée de manière exogène.

Par ailleurs, le MRP1 cherche à établir une programmation de la production sans se poser le problème des capacités de production. La démarche qui lui a succédé est connue sous le nom de MRP2 (Manufacturing Resources Planning, i.e. planification des ressources de production). Cette dernière comprend le MRP1 mais de plus, elle confronte la charge souhaitée et la charge disponible pour chaque centre de production, afin d'en déduire une programmation réalisable.

Dans ce chapitre, nous présentons les fondements du MRP, en illustrant les différents concepts introduits par un exemple.

1. Une présentation d'ensemble du MRP

La planification de la production requiert pour sa mise en place un certain nombre de données physiques, ou "ressources". Ces données sont indispensables pour établir le programme directeur de production dont découle la planification des besoins en composants. Les propositions de fabrication qui en résultent conduisent à évaluer les capacités de production requises pour leur réalisation ; une planification des besoins en capacité est ainsi effectuée. Les propositions seront alors acceptées dans la limite des capacités de production disponibles. Schématiquement, les ressources, l'activité et les résultats du MRP2 peuvent être représentés sous la forme de tableaux (cf. Figure 1).

Les données physiques constituent les ressources du MRP2 et se subdivisent en données techniques (Tableau A de la Figure 1) et en données de flux (Tableau B). Ces données sont classées dans des fichiers dont la dénomination apparaît en encadré dans les tableaux A et B. Elles sont fondamentales pour établir le programme directeur de production (Tableau C), la planification des besoins en composants (Tableau D) et la planification des besoins en capacité (Tableau E). C'est ce que nous appellerons "activité" du MRP2. Cette activité donne lieu à des données de résultats consignées dans les tableaux F et G de la Figure 1. Dans les sections qui suivent, nous examinons successivement les ressources, l'activité et les résultats du MRP2, en procédant à l'analyse du contenu des différents tableaux susmentionnés.

2. Ressources du MRP : les données physiques

La planification s'appuie sur des informations relatives aux caractéristiques physiques du système de production. Dans cette section, nous détaillons ces informations qui regroupent les données techniques et les données de flux.

2.1 Les données techniques

Ces données sont organisées en fichiers dont les termes dénominatifs apparaissent dans le Tableau A de la Figure 1. Ici, nous détaillons le contenu de ces fichiers et illustrons notre présentation à l'aide d'un exemple de produit. A ce titre, considérons une entreprise fabriquant un unique modèle de lunettes. A la Figure 2, on donne le schéma des lunettes.

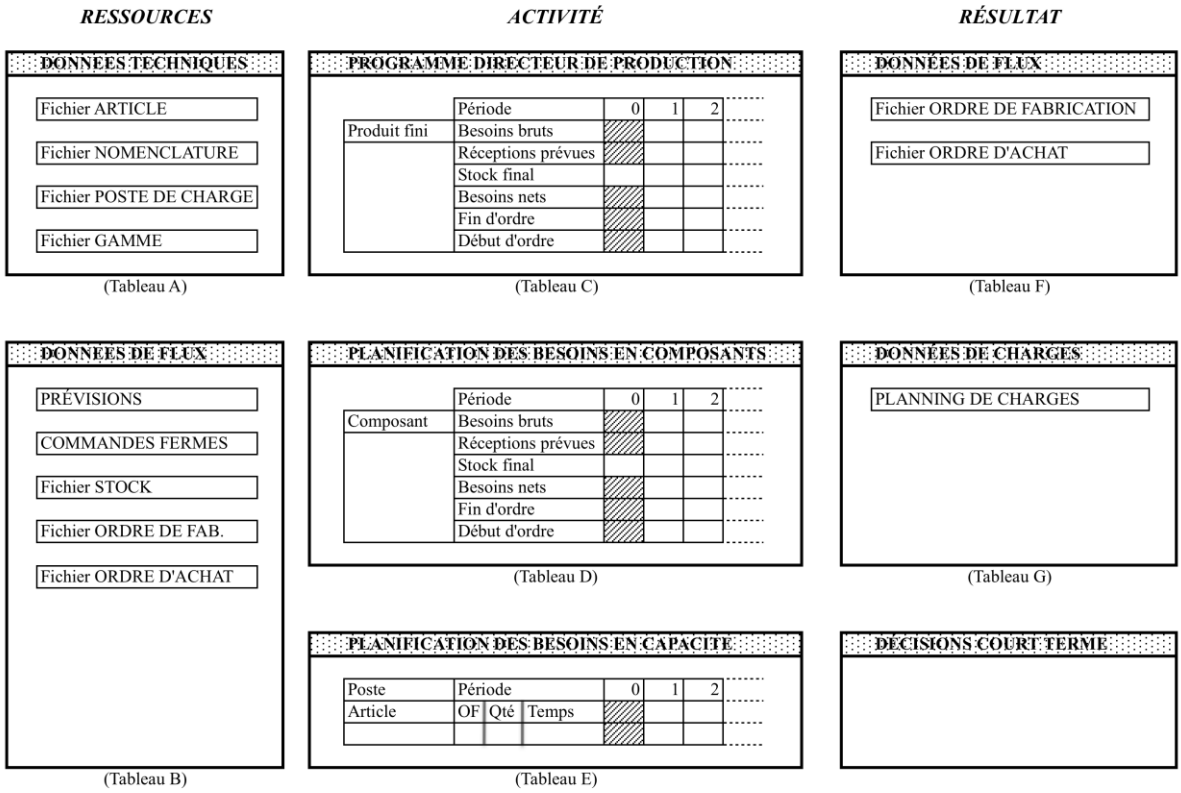


Figure 1. Une présentation d'ensemble du MRP

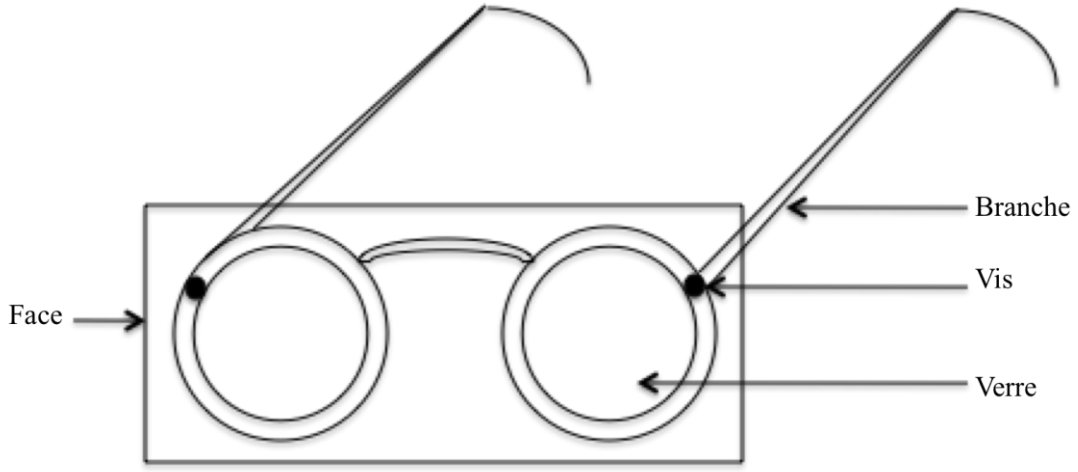


Figure 2. Schéma d'une paire de lunettes

Le détail des fichiers de données techniques se trouve à la Figure 3 et concerne le modèle de lunettes décrit précédemment. Finalement, la Figure 3 est une représentation détaillée du Tableau A de la Figure 1.

Fichier ARTICLE

Code	Désignation	Unité stock	Type	Stock sécurité	Lot mulitple	Règle d'appro.	Délai	Lot standard
LU	Lunettes	1	F (fabriqué)	0	10	Lot pour lot	2	50
VEDEP	Verres	1	F	500	10	Lot pour lot	2	100
MO	Monture	1	F	250	10	Groupage 5j.	1	50
VEBR	Verre brut	1	A (acheté)	1500	10	Groupage 5j.	3	50
FA	Face	1	F	250	10	Groupage 5j.	1	50
BR	Branche	1	F	500	10	Qté fixe = 500	1	50
VS02	Vis	1	A	500	10	Qté fixe = 500	2	50
PL	Plastique	1	A	2500	10	Qté fixe = 500	3	50

Fichier NOMENCLATURE

Composé	Composant	Niveau	Coeff. Lien
LU		0	
	VEDEP	1	2
	MO	1	1
VEDEP		1	
	VEBR	2	4
MO		1	
	FA	2	1
	BR	2	2
	VS02	2	2
FA		2	
	PL	3	8
BR		2	
	PL	3	1

Fichier POSTE DE CHARGE

Poste	Désignation	Capa. Théorique (jours/semaine)	Coeff. Minimum en %	Capacité réelle
MT	Montage	5	20	4
US	Usinage	5	10	4,5

Fichier GAMME

Opération	Désignation	Poste de charge	Temps en jours	
			Temps prépa.	Temps unit.
LU 10	Montage	MT	0,2	0,004
VEDEP 10	Usinage	US	0,15	0,01
MO 10	Montage	MT	0,2	0,008

Figure 3. Les fichiers de données techniques (détail du Tableau A de la Figure 1)

Avant de procéder à la description du contenu de ces fichiers, introduisons quelques définitions pour clarifier le vocabulaire que nous utiliserons dans la suite.

Définitions des principaux types d'articles

Un article ou produit est un objet qui, par l'intérêt qu'il présente pour le traitement de l'information, justifie sa codification, c'est-à-dire son identification formelle par un code. On parle encore de référence

Ainsi, des chutes de bois, objets "conçus" dans une entreprise, ne seront probablement pas codées si elles ne sont ni réutilisées, ni stockées, ni demandées : dans ce cas, on peut imaginer qu'une codification puisse être inutile.

Un produit fini est un article exclusivement destiné à la vente. On dit encore d'un produit qu'il est fini, lorsqu'il ne subit plus aucune transformation au sein de l'entreprise considérée (il

est "fini" pour cette entreprise). Cette précision permet d'éviter les ambiguïtés, dans la mesure où l'on assimile généralement un produit fini à un objet directement utilisable en l'état.

Ainsi, une pièce détachée ou pièce de rechange, objet destiné au remplacement, est un produit fini si elle est exclusivement destinée à la vente. Une matière première est un produit fini pour l'entreprise qui l'achète dans le seul but de la revendre.

Un composant est un article qui entre dans la composition d'un produit plus complexe ou plus élaboré. De façon équivalente, un article est un composant lorsqu'il est utilisé pour l'obtention d'un autre article. Tous les articles sont des composants à l'exception des produits finis.

Le fer, par exemple, entre dans la composition des vis : produits plus élaborés que cette matière première.

Un composant "commun" à plusieurs produits est un article qui entre dans la composition respective de ces derniers. On parle encore de composant à usages multiples, par opposition au composant qui sert à l'obtention d'un unique article.

Dans notre exemple de lunettes, le plastique est un composant commun à la face et aux branches.

Un composé est un article qui résulte de la transformation au sein de l'entreprise d'au moins un composant. Tous les produits sont des composés, à l'exception des articles achetés encore appelés approvisionnements externes.

Un produit semi-fini ou semi-ouvré est un article inachevé, et entreposé en attendant de lui appliquer les dernières opérations qui permettront de l'adapter aux exigences de la clientèle.

Un produit intermédiaire est un article résultant d'une ou plusieurs transformations, et destiné à en subir d'autres ; un produit semi-fini est donc intermédiaire.

Les pièces usinées dans l'entreprise, sous-ensembles et mélanges sont des exemples de produits intermédiaires.

Le fichier article

Le fichier article contient des données relativement permanentes nécessaires à la description des articles.

Dans la première colonne de ce fichier, on trouve le code des produits. La signification de ces codes est reportée dans la colonne désignation.

L'unité de stock est le nombre des localisations en stock des articles. Si un article est stocké dans l'usine et dans six centres de distribution, il lui correspondra 7 unités de stock.

Stock et délai de sécurité

Les stocks de sécurité sont utilisés dans les systèmes MRP lorsque l'incertitude concerne les quantités, par exemple lorsque les rebuts ou les surplus de demande sont fréquents.

Dans un système productif caractérisé par l'existence de stocks multi-échelon, les marges de sécurité, appliquées à un échelon, permettent également de se prémunir contre les incertitudes susceptibles d'apparaître dans les autres échelons. Ainsi, la constitution d'un stock de sécurité au niveau du produit fini peut faciliter l'absorption des erreurs de prévision. Certaines situations obligent à la constitution de stocks à d'autres niveaux que celui du

produit fini. C'est notamment le cas lorsque le taux de rebut d'une production particulière est élevé, ou lorsque les quantités livrées d'une matière achetée sont souvent inférieures aux quantités commandées.

Les délais de sécurité sont utilisés lorsque l'incertitude pèse plutôt sur le temps. S'il arrive fréquemment que l'un des fournisseurs d'une firme ne respecte pas les délais de livraison, l'introduction d'un délai de sécurité est plus pertinente que la constitution d'un stock de sécurité. Concrètement, le délai de sécurité est ajouté au délai "normal" en vue d'exécuter un ordre d'approvisionnement (achat ou production) avant sa date d'exigibilité, de manière à absorber les variations du délai de livraison.

Lot multiple : la taille de l'approvisionnement doit être un multiple de cette quantité. Dans notre exemple (cf. Figure 3), les paires de lunettes doivent être fabriquées par lots de 10 ou un multiple de la dizaine.

Règles d'approvisionnement

Les règles d'approvisionnement indiquent la quantité d'unités constituant un lot d'approvisionnement, pour chaque article considéré isolément. Nous supposerons ici que les lots sont déterminés de façon exogène. Les règles d'approvisionnement que l'on mentionne à titre indicatif dans le fichier article (cf. Figure 3) figurent au rang des techniques les plus simples.

- le lot pour lot consiste à s'approvisionner à chaque période d'une quantité égale à la demande (de la dite période) ;
- le groupage x jours conduit à effectuer des lots dont le nombre d'unités correspond à la demande de x jours ;
- la quantité fixe de commande, comme son nom l'indique, impose un volume donné d'approvisionnement.

Délai en jours : il s'agit du délai moyen d'approvisionnement c'est-à-dire calculé pour un lot d'une taille donnée, mais les lots effectivement lancés en production pourront être d'une taille différente. Ce délai d'obtention se définit comme la somme du temps opératoire (incluant le temps de lancement et le temps de fabrication) et de temps inter-opératoires (temps de transit entre les centres de production par exemple).

Lot standard : le lot standard est un lot virtuel moyen, notamment utilisé pour calculer des coûts moyens.

Le fichier nomenclature

Une nomenclature de produits est une codification exhaustive et bi-univoque de tous les articles d'une entreprise donnée.

Cette liste de codes fait apparaître les relations d'inclusion des produits entre eux. En effet, elle est organisée en niveaux qui représentent les différents stades de décomposition des produits. A un niveau donné n, on dispose les produits inclus dans d'autres, placés quant à eux, au niveau n-1. Aussi parle-t-on généralement de liste hiérarchisée de codes. Dans cette liste, figurent également les quantités respectives des produits inclus dans une unité de l'article dont ils sont les composants. On appelle ces quantités des coefficients techniques, ou coefficients de lien. Par convention, on place les produits finis au niveau zéro puis, au niveau un, les composés qui entrent directement dans la composition des produits finis, etc.

Le fichier nomenclature ou fichier liens de structure contient les relations d'inclusion des composants dans les composés. Ce fichier peut être représenté sous la forme d'un tableau (cf. fichier nomenclature dans le Figure 3). Dans la première colonne, on inscrit le code du composé ; dans une deuxième colonne, on saisit les codes des composants qui entrent directement dans la composition de ce composé, et dans une troisième colonne, on indique les niveaux respectifs de ces articles. Enfin, dans une dernière colonne, on place les coefficients de lien.

A titre d'illustration, considérons le fichier nomenclature du Figure 3. Le composé "LU" (lunettes) est un produit du niveau zéro puisqu'il est fini. Deux unités de l'article "VEDEP" et une unité du produit "MO" sont requises pour l'obtention d'une unité de "LU". Autrement dit, il faut deux verres et une monture pour produire une paire de lunettes.

A partir du fichier liens de structure, il est possible de générer des nomenclatures dans une grande diversité de formes. Le Figure 4 représente la nomenclature décalée ou nomenclature sous forme arborescente des articles de l'entreprise de lunettes.

C	O	D	E	S				Coeff. Lien
L	U							
	M	O						1
		F	A					1
			P	L				8
		B	R					2
			P	L				1
		V	S		0		2	2
	V	E	D	E		P		2
		V	E	B		R		4

Figure 4. Nomenclature décalée des articles de l'entreprise de lunettes

La nomenclature sous forme arborescente est une représentation d'une nomenclature multi-niveau, où chaque composant d'un composé est placé à sa droite et en dessous. Par exemple, les composants MO et VEDEP du composé LU sont disposés en dessous et à la droite de ce dernier. Cette présentation permet d'identifier les sous-ensembles d'un produit fini, et d'en connaître la composition. La nomenclature décalée est l'outil du bureau d'études. De plus, puisqu'elle montre les différentes étapes de l'élaboration du produit fini, elle fournit un bon guide pour la planification des achats et de l'activité des ateliers.

On peut extraire de la nomenclature complète, une nomenclature partielle, relative à un seul produit. Ces nomenclatures partielles peuvent être représentées sous la forme d'un diagramme de structure. La Figure 5représente le diagramme de structure de la paire de lunettes. Les codes des articles sont inscrits à l'intérieur de cases, reliées par des lignes à d'autres cases. Ces lignes symbolisent les liens de composants à composés (lecture ascendante du diagramme) ou de composés à composants (lecture descendante du diagramme). Les chiffres entre parenthèses sont les coefficients de lien.

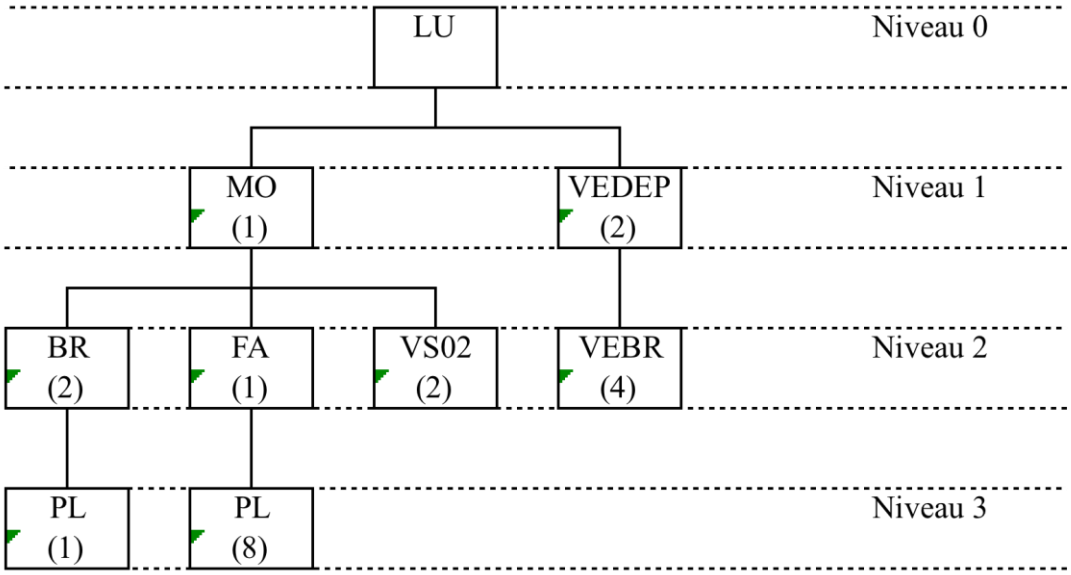


Figure 5. Diagramme de structure des lunettes

Le fichier poste de charge

Un poste de charge est une unité de production virtuelle, constituée d'un ou plusieurs centres d'activité, comprenant des machines et des outillages nécessaires à l'exécution de tâches. Cette unité de production est notamment utilisée pour la planification des besoins en capacité. Dans le fichier poste de charge du Figure 3, on trouve certaines informations que nous détaillons ci-après.

La capacité théorique représente le nombre maximum de jours de fonctionnement, par semaine en l'occurrence.

Le coefficient minimum en pourcent est le pourcentage de temps durant lequel le poste ne fonctionne pas.

La capacité réelle est le nombre de jours de fonctionnement effectif. Cette capacité est égale à (1-coefficient minimum) × capacité théorique.

Le fichier gamme

Préparée par le bureau des méthodes, la gamme d'opération est une liste des différentes opérations nécessaires à la fabrication ou l'assemblage d'une pièce, d'un ensemble ou sous-ensemble.

Dans la colonne opération du fichier gamme (cf. Figure 3), on consigne le code des articles et le numéro d'ordre des opérations à effectuer sur ces articles.

Le temps de préparation est le temps nécessaire pour préparer les machines à l'opération (réglage, nettoyage...).

Le temps unitaire est le temps requis pour la fabrication ou l'assemblage d'un seul article.

2.2 Les données de flux

Comme précédemment, nous allons procéder à l'examen des fichiers de données de flux, en nous appuyant sur le même exemple. La Figure 6 est une représentation détaillée des fichiers du Tableau B de la Figure 1.

PREVISIONS au 1er mars				
LU	Semaine 1	Semaine 2	Semaine 3	Semaine 4
Prévisions/jour	50	10	10	30

COMMANDES FERMES			
Commande	Article	Quantité	Pour le
500	LU	20	01 mars
501	LU	20	05 mars
502	LU	20	08 mars
503	LU	50	10 mars

FICHIER STOCK au 1er mars		
Article	Type	Stock final
LU	F	20
MO	F	60
VEDEP	F	200
etc.		

FICHIER ORDRE DE FABRICATION au 1er mars					
Numéro OF	Article	Quantité	Début	Fin	Statut
10	LU	80	19 février	01 mars	L (lancé)
20	BR	60	20 février	01 mars	L (lancé)

FICHIER ORDRE D'ACHAT au 1er mars					
Numéro OA	Article	Quantité	Début	Fin	Statut
50	VEBR	60	20 février	03 mars	L (lancé)
51	PL	20	20 février	03 mars	L (lancé)

Figure 6. Les fichiers de données de flux

Les prévisions de vente

Seules les demandes indépendantes font l'objet de prévisions statistiques. Les demandes dépendantes, par définition, se déduisent des demandes indépendantes. Par conséquent, il serait incorrect d'établir des prévisions pour ce type de demandes. Dans notre exemple, on prévoit de vendre 50 paires de lunettes chaque jour de la première semaine.

Les commandes fermes

Les données de ce fichier proviennent du carnet de commandes de l'entreprise. Ce carnet contient le volume des commandes qui ne sont pas encore exécutées ou livrées.

Les dates consignées dans le fichier des commandes fermes sont toutes référencées à 8h00 du matin. Par exemple (cf. Figure 6), la commande n° 500 d'une quantité de 20 unités de LU doit être livrable le 1er mars à 8h00 du matin. Notons qu'il y a 20 jours ouvrables par mois, et pour simplifier, le mois ne tient pas compte des samedis et dimanches. Ainsi, le mois commence le 1er et se termine le 20.

Le fichier stock

Ce fichier donne le volume de stock disponible ou stock final à la date mentionnée, pour chaque article. Le stock final est souvent établi par inventaire physique.

Les fichiers ordre de fabrication, ordre d'achat

Un ordre de fabrication est un document donnant instruction à la fabrication de produire une quantité donnée d'article. Ce document déclenche la mise en fabrication et précise l'article à fabriquer, la quantité et la date de fin de fabrication souhaitées. Le fichier ordre de fabrication (OF) rassemble les OF émis. De façon analogue, un ordre d'achat ou bon de commande comprend les quantités d'articles commandés, leur description et leur date de livraison. Ces ordres d'approvisionnement, calculés antérieurement, ont été maintenus et donc lancés afin d'assurer des entrées d'articles en stock. A ce titre, ils ont toujours le statut "L", i.e. lancé.

3. L'activité du MRP : le programme directeur de production et la planification des besoins en composants

Le programme directeur de production est un programme de fabrication exprimant les décisions de production en articles finis, en réponse à une demande prévisionnelle sur un horizon donné. A titre d'illustration, l'entreprise de lunettes établit un programme directeur de production pour les lunettes, en tenant compte des prévisions, de son carnet de commandes ainsi que des ressources disponibles. A partir de ce programme, on déduit successivement les demandes (besoins) pour les articles des niveaux inférieurs. En effet, connaissant la production souhaitée de lunettes, on déduit sans difficulté la production adéquate de branches, de montures, etc., puisqu'il faut deux branches et une monture pour fabriquer une paire de lunettes. Une planification des besoins en composants est ainsi réalisée.

3.1 Le programme directeur de production

Les quantités programmées sont établies pour un certain nombre de périodes futures constituant l'horizon de planification. Le programme directeur de production (PDP, dans la suite) contient principalement cinq informations pour chaque produit fini et chaque période : les besoins bruts, les réceptions prévues, le stock final, les besoins nets, les débuts et fins

d'ordres de fabrication. La Figure 7 est un exemple de programme directeur de production pour les lunettes.

	Période	0	1	2	3	4	5
LU	Besoins bruts	/	50	50	50	50	50
Lot pour lot	Réceptions prévues	/	80	0	0	0	0
Lot multiple = 10	Stock final	20	50	0	0	0	0
Stock sécurité = 0	Besoins nets	/	0	0	50	50	50
Délai = 2	Fin d'ordre	/	0	0	50	50	50
	Début d'ordre	/	50	50	50	0	0

Figure 7. Programme directeur de production des lunettes

Les besoins bruts

Les besoins bruts sont les quantités requises à chaque période pour satisfaire la demande anticipée, et assurer la livraison des commandes fermes. Ces besoins résultent d'un calcul souvent complexe faisant intervenir les délais, les données de commandes fermes, de prévisions et la capacité de production.

Les réceptions prévues

Les réceptions prévues sont des ordres de fabrication ou d'achat déjà lancés, c'est-à -dire en cours de réalisation. Ces quantités résultent de décisions d'approvisionnement antérieures, et sont susceptibles d'assurer une couverture partielle des besoins bruts. Ces réceptions sont importées des fichiers ordre de fabrication et ordre d'achat.

Ainsi, les 80 paires de lunettes que l'on prévoit de recevoir à la première période (cf. Figure 7) sont des quantités qui apparaissent également dans le fichier ordre de fabrication de la Figure 6.

Le stock final, les besoins nets, les fins et débuts d'ordres d'approvisionnement

Soit T, le nombre de périodes de l'horizon de planification et t, une période avec $t = 1,\mathsf{K}\ ,T$. On note, pour un article donné

BB_t, les besoins bruts à la période t
RP_t, les réceptions prévues (appelées aussi livraisons attendues) à la période t
SP_t, le stock projeté à la période t
BN_t, les besoins nets à la période t
x_t, la fin d'ordre d'approvisionnement (ou livraison programmée) c'est-à-dire le volume d'article calculé selon une règle particulière d'approvisionnement et dont on souhaite la livraison à la période t
I_t, le stock disponible à la période t, i.e. le volume de stock à la fin de la période t.

Le stock projeté est égal au stock physique de la période, augmenté des réceptions prévues et diminué des besoins bruts. Ainsi, $SP_t = SP_{t-1} + RP_t - BB_t$.

Les besoins nets correspondent aux quantités qu'il convient de commander ou de fabriquer, compte tenu des besoins bruts et des disponibilités de matières. Les besoins nets se calculent à partir du stock projeté. On a :

$$BN_t = \begin{cases} 0 & \text{si } \max(0, SP_{t-1}) + RP_t \geq BB_t \\ BB_t - RP_t - \max(0, SP_{t-1}) & \text{sinon} \end{cases}$$

A la Figure 8, on donne un exemple de calcul du stock projeté et des besoins nets en lunettes.

	Période	0	1	2	3	4	5
LU	Besoins bruts	/	50	50	50	50	50
	Réceptions prévues	/	80	0	0	0	0
	Stock projeté	20	50	0	-50	-100	-150
	Besoins nets	/	0	0	50	50	50

Figure 8. Calcul du stock projeté et des besoins nets en lunettes

Il s'agit ensuite d'assurer la couverture des besoins nets, en déterminant la séquence des fins d'ordres d'approvisionnement x_t pour $t = 1, \ldots, T$. Cette séquence résultera du choix d'une règle d'approvisionnement. Le stock final I_t nécessite ce calcul préalable, puisqu'il comprend à chaque période t le volume d'approvisionnement x_t. Le stock final I_t s'écrit :

$I_t = I_{t-1} + x_t + RP_t - BB_t$

avec $I_0 = SP_0$

A titre d'illustration, considérons la période 3 du PDP pour les lunettes (cf. Figure 7). On a $BN_3 = 50$ et $I_2 + RP_3 = 0$. Ainsi, un approvisionnement est requis car le stock de la période précédente augmenté des réceptions prévues ne permet pas de satisfaire les besoins nets. Comme la règle d'approvisionnement est le lot pour lot, une quantité de 50 unités doit être commandée à la période 1 (début d'ordre), de sorte à être disponible à la période 3 (fin d'ordre) ; aussi a-t-on $x_3 = 50$ et $I_3 = I_2 + x_3 + RP_3 - BB_3 = 0$.

3.2 La planification des besoins en composants

La planification des besoins en composants consiste à déterminer les quantités de composants nécessaires à la réalisation du programme directeur de production, à partir des nomenclatures et des états de stock. Les quantités requises de composants se calculent selon le principe d'explosion des nomenclatures. Les besoins en composants du niveau 1 se déduisent des débuts d'ordres de fabrication en produits finis. On détermine ensuite les besoins en articles du niveau 2, à partir des débuts d'ordres de fabrication des produits du niveau 1. Ce procédé est réitéré à chaque niveau jusqu'aux niveaux d'utilisation des articles achetés.

La Figure 9 est un exemple de planification des besoins en composants qui entrent dans la fabrication des lunettes.

PROGRAMME DIRECTEUR DE PRODUCTION

	Période	0	1	2	3	4	5	6	7	8	9	10
LUNETTES	Besoins bruts		50	50	50	50	50	10	10	10	10	10
Lot pout lot	Réceptions prévues		80									
Lot multiple = 10	Stock final	20	50	0	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	0	50	50	50	10	10	10	10	10
Délai = 2 périodes	Début d'ordre		50	50	50	10	10	10	10	10	0	0

PLANIFICATION DES BESOINS EN COMPOSANTS

Niveau 1

	Période	0	1	2	3	4	5	6	7	8	9	10
MONTURES	Besoins bruts		50	50	50	10	10	10	10	10	0	0
Lot pout lot	Réceptions prévues											
Lot multiple = 10	Stock final	60	10	0	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	40	50	10	10	10	10	10	0	0
Délai = 1 période	Début d'ordre		40	50	10	10	10	10	10			

	Période	0	1	2	3	4	5	6	7	8	9	10
VERRES	Besoins bruts		100	100	100	20	20	20	20	20	0	0
Lot pout lot	Réceptions prévues											
Lot multiple = 10	Stock final	200	100	0	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	0	100	20	20	20	20	20	0	0
Délai = 2 périodes	Début d'ordre		100	20	20	20	20	20	0	0	0	

Niveau 2

	Période	0	1	2	3	4	5	6	7	8	9	10
BRANCHE	Besoins bruts		80	100	20	20	20	20	20	0	0	0
Lot pout lot	Réceptions prévues		60									
Lot multiple = 10	Stock final	20	0	0	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	100	20	20	20	20	20	0	0	0
Délai = 1 période	Début d'ordre		100	20	20	20	20	20	0			

	Période	0	1	2	3	4	5	6	7	8	9	10
FACE	Besoins bruts		40	50	10	10	10	10	10	0	0	0
Lot pout lot	Réceptions prévues											
Lot multiple = 10	Stock final	50	10	0	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	40	10	10	10	10	10	0	0	0
Délai = 1 période	Début d'ordre		40	10	10	10	10	10	0	0	0	0

	Période	0	1	2	3	4	5	6	7	8	9	10
VIS	Besoins bruts		80	100	20	20	20	20	20	0	0	0
Lot pout lot	Réceptions prévues											
Lot multiple = 10	Stock final	200	120	20	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	0	0	20	20	20	20	0	0	0
Délai = 2 périodes	Début d'ordre		0	20	20	20	20	0	0	0	0	0

	Période	0	1	2	3	4	5	6	7	8	9	10
VERRE BRUT	Besoins bruts		400	80	80	80	80	80	0	0	0	0
Lot pout lot	Réceptions prévues											
Lot multiple = 10	Stock final	500	100	20	0	0	0	0	0	0	0	0
Stock de sécurité = 0	Besoins nets		0	0	60	80	80	80	0	0	0	0
Délai = 3 périodes	Début d'ordre		80	80	80	0	0	0	0	0	0	0

Figure 9. Planification des besoins en composants des lunettes

Pour simplifier, nous avons supposé que les stocks de sécurité sont nuls. La règle d'approvisionnement choisie est le lot pour lot pour chacun des articles. Ainsi, les fins d'ordres de fabrication sont identiques aux besoins nets. Notons également que nous n'avons pas traité les besoins en composants du niveau 3, car le principe demeure le même à chaque

niveau. Les besoins bruts en composants se déduisent des débuts d'ordres de fabrication des articles dans lesquels ils sont inclus. Par exemple, on souhaite une entrée en stock de 10 paires de lunettes à la période 10. Le délai de fabrication étant de 2 périodes, un ordre de fabrication de 10 unités doit être lancé à la période 8. Pour réaliser cette production, on doit disposer de 10 montures et 20 verres à la période 8 : besoins bruts en ces composants à cette période (cf. Figure 9). Les réceptions prévues en composants proviennent des fichiers ordre d'achat et ordre de fabrication.

Notre exemple montre que la date de fin d'un ordre de fabrication d'un composé définit la date de début et de fin des ordres de fabrication de ses composants. Cette technique de détermination des dates de début et fin au plus tard est connue sous le nom de jalonnement amont ou jalonnement au plus tard. Le jalonnement amont est souvent représenté par un diagramme. A la Figure 10, on trouve ce type de diagramme pour les lunettes. Les dates de fin d'ordres d'approvisionnement sont placées en abscisse. Chaque article est symbolisé par un rectangle dont la longueur est proportionnelle à son délai d'obtention.

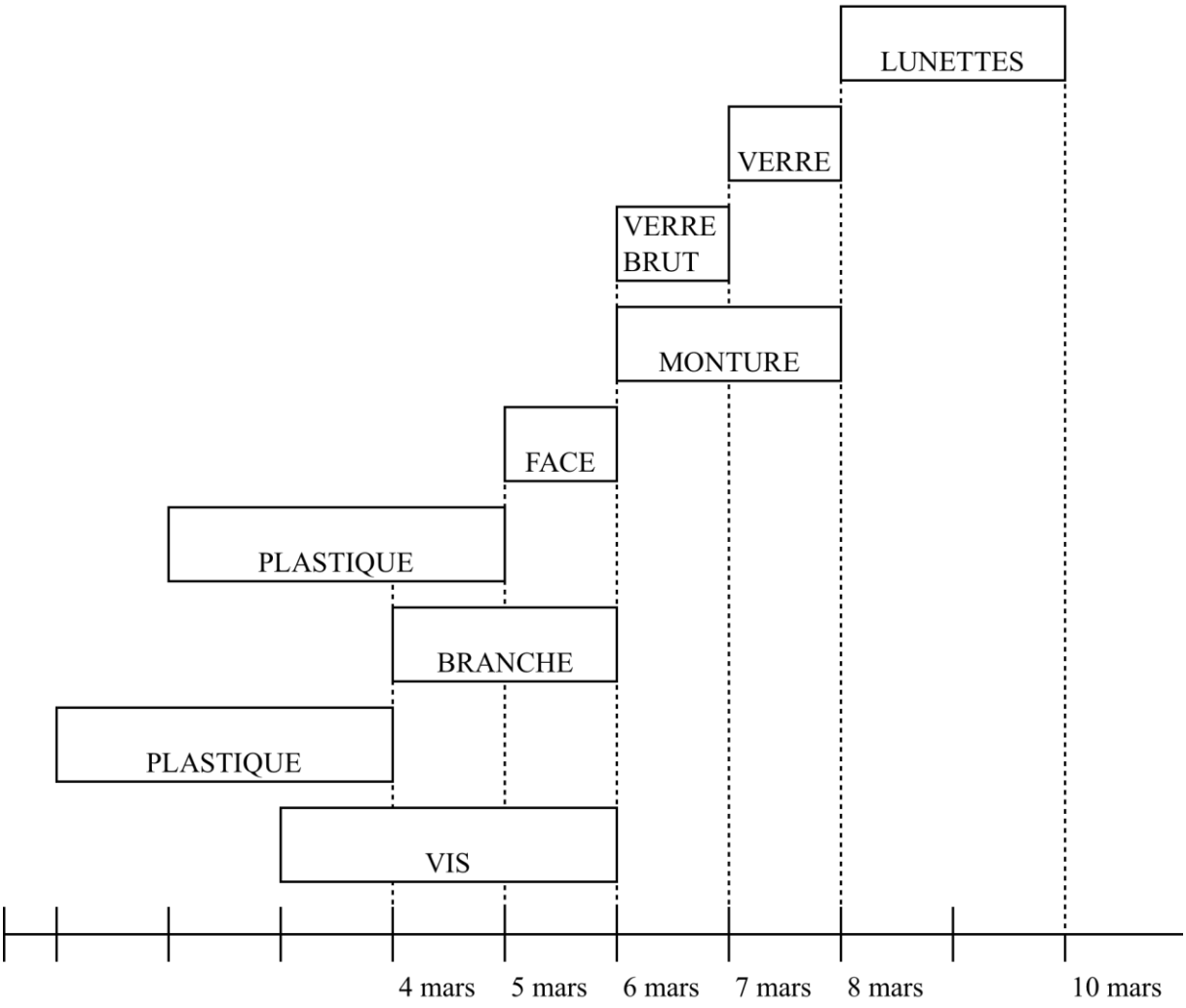


Figure 10. Représentation du jalonnement amont pour les lunettes

Jusque là, nous avons supposé que la taille des lots d'approvisionnement était déterminée de façon exogène. En réalité, pour assurer la couverture des besoins nets, on emploie des règles de détermination des lots qui reposent généralement sur un arbitrage entre le coût fixe de commande ou de fabrication et le coût de stockage. La méthode MRP (MRP1 ou MRP2) qui inclut le recours à de tels modèles d'approvisionnement constitue fondamentalement une alternative à la gestion des stocks multi-échelon par les politiques classiques de gestion des stocks.

Nous sommes maintenant en mesure d'expliquer les raisons de l'inadéquation des politiques traditionnelles de gestion des stocks, au cas des stocks multi-échelon. Dans la section qui suit, nous illustrons les problèmes posés à l'aide de l'exemple, et montrons en quoi la méthode MRP en apporte précisément les réponses.

4. L'ajustement charge - capacité (MRP2)

Jusqu'à présent, nous avons raisonné comme si la capacité de production était infinie, c'est-à-dire que nous avons procédé au calcul des lots sans nous préoccuper de leur réalisabilité au regard de la capacité disponible. Il convient donc dans un premier temps de calculer la charge qu'impliquent les lots et de la confronter à cette capacité disponible.

Pour illustrer le raisonnement, on va supposer qu'il n'y a pas de problème de capacité aux niveaux de nomenclature 0 et 1 et l'on va considérer uniquement les articles du niveau 2, FA et BR qui nécessitent un passage sur le poste d'usinage pour leur fabrication. Les composants VS02 et VEBR étant achetés, ils ne génèrent pas de charge sur ce poste d'usinage. Pour les besoins de l'illustration, nous avons changé l'échéancier de demandes en lunettes et obtenons, en appliquant le lot pour lot les débuts d'ordre de fabrication consignés à la Figure 11. On sait d'après la Figure 3 (fichier Gamme), que le poste d'usinage a une capacité de 4.5 jours par semaine, et les temps d'usinage unitaires respectifs des articles BR et FA sont de 0.003 et 0.007. Pour obtenir la charge sur le poste usinage, il suffit de multiplier ces temps par les débuts d'ordre de fabrication pour chaque article et d'en faire la somme.

	Période	1	2	3	4	5	6	7	8	9	10
BRANCHE	Début d'ordre	200	560	440	300	800	1160	0	0	0	0
Temps unitaire = 0.003	Charge	0.6	1.68	1.32	0.9	2.4	3.48	0	0	0	0
FACE	Début d'ordre	100	280	220	150	400	580	0	0	0	0
Temps unitaire = 0.007	Charge	0.7	1.96	1.54	1.05	2.8	4.06	0	0	0	0
POSTE USINAGE	CHARGE TOTALE	1.3	3.64	2.86	1.95	5.2	7.54	0	0	0	0
	Excédent ou déficit	3.2	0.86	1.64	2.55	**-0.7**	**-3.04**	4.5	4.5	4.5	4.5

Figure 11. Calcul de la charge sur le poste usinage au niveau 1

On constate un déficit en capacité aux période 5 et 6 pour un total de 0.7+3.04=3.74 jours qu'il est possible de résoudre selon les deux méthodes que nous allons présenter dans les sous-sections suivantes.

4.1 Ajustement par les stocks

En anticipant la production des articles aux périodes 4 et 3 où il existe un excédent de capacité. Il convient donc d'utiliser 2.55 jours à la période 4 et le complément, à savoir 3.74-2.55=1.19 jours à la période 3 (ce qui est possible puisque l'excédent de cette période est de 1.64). Cette production anticipée implique un stockage que l'on souhaite le moins onéreux possible ce qui conduit à choisir l'article le moins cher à stocker. Pour cela, on doit évaluer le coût de stockage unitaire des articles FA et BR, partant de l'hypothèse communément admise

que le coût de stockage d'un article est proportionnel à la valeur de celui-ci. Cette valeur, ou coût de revient de l'article i et noté c_i comprend le coût de fabrication de celui-ci et le coût de ses composants. On adopte les notations suivantes. Soit

h_i, le coût unitaire de stockage de l'article i par période, avec $h_i = \alpha c_i$ (on pourra prendre par exemple $\alpha = 0.01$) ;

m_k, le coût de fonctionnement du poste k par période ;

$\tau_{k,i}$, le temps requis sur le poste k pour fabriquer une unité d'article i ;

$P(i)$, l'ensemble des articles inclut directement dans la composition de i ;

$l_{j,i}$, le nombre d'unités de composant j nécessaires à la fabrication d'une unité d'article i (coefficient de lien).

Si $P(i)$ est vide, c'est-à-dire si le composant est acheté, son coût de revient est simplement son coût d'achat. Dans le cas contraire, le coût de revient est donné par

$$c_i = m_k \cdot \tau_{k,i} + \sum_{j \in P(i)} l_{j,i} \cdot c_j$$

Pour effectuer ce calcul, il convient de commencer par les articles achetés pour remonter progressivement les niveaux de nomenclature jusqu'aux produits finis.

Dans notre exemple, les articles BR et FA utilisent le composant PL qui est acheté et dont le coût unitaire d'achat est égal à 9 euros ($c_{PL} = 9$). On suppose que le coût de l'atelier d'usinage par semaine est de 700 euros ($m_{US} = 700$). Le fichier gamme indique $\tau_{US,BR} = 0.003$ et $\tau_{US,FA} = 0.007$. On a $P(BR) = P(FA) = PL$ et $l_{PL,BR} = 1; l_{PL,FA} = 8$. Ce qui donne $c_{BR} = 700 \times 0.003 + 1 \times 9 = 11.1$ et $c_{FA} = 700 \times 0.007 + 8 \times 9 = 76.9$, ce qui n'est pas surprenant compte tenu du fait que la FACE est plus élaborée que la BRANCHE. On va donc anticiper la production de l'article le moins cher à stocker, à savoir l'article BR.

A la période 6, le déficit de capacité est de 3.04 jours (ce qui représente 3.04/0.003=1013 unités à retrancher du début d'ordre en BR à la période 6, soit 1160-1013=147). CE déficit peut être comblé en utilisant l'excédent de capacité de 2.55 jours en période 4, ce qui représente 2.55/0.003=850 unités de plus à produire que prévu initialement ; le début d'ordre corrigé en période 4 est alors de 300+850=1150. Le complément, à savoir 3.04-2.55=0.49 peut être comblé en utilisant l'excédent de capacité à la période 3 (excédent égal à 1.64), ce qui représente 0.49/0.003=163 unités de plus à produire en période 3.

A la période 5, le déficit est de 0.7 ce qui représente 0.7/0.003=233 unités à retrancher du début d'ordre initialement prévu (soit 800-233=567). Ce déficit peut être comblé en utilisant la capacité excédentaire de la période 3 qui est désormais égale à 1.64-0.49=1.15. Tout peut donc se rattraper en période 3 puisque la capacité excédentaire restante (1.15) est supérieure à 0.7. On produire en plus la période 3 ces 233 unités. Le début d'ordre corrigé de la période 3 est finalement égal à 440+163+233=836. La figure 12 fournit un résumé des opérations (on s'arrête à la période 6 au delà de laquelle les débuts d'ordre de fabrication sont nuls).

	Période	1	2	**3**	**4**	**5**	**6**
BRANCHE	Début d'ordre	200	560	**440**	**300**	**800**	**1160**
	Variation			**+(3.04-2.55)/0.003 +0.7/0.003**	**+2.55/0.003**	**-0.7/0.003**	**-3.04/0.003**
	Début d'ordre corrigé	200	560	**836**	**1150**	**567**	**147**
Temps unitaire = 0.003	Charge	0.6	1.68	2.508	3.45	1.701	0.441
FACE	Début d'ordre	100	280	220	150	400	580
Temps unitaire = 0.007	Charge	0.7	1.96	1.54	1.05	2.8	4.06
POSTE USINAGE	CHARGE TOTALE	1.3	3.64	4.048	4.5	4.501	4.501
	Excédent ou déficit	3.2	0.86	0.452	0	-0.001	-0.001

Figure 12. Ajustement charge - capacité sur le poste usinage

4.2 Ajustement par adaptation de la capacité

On peut envisager le recours à du travail temporaire pour accroître momentanément la capacité de production (soit par l'embauche d'intérimaire, soit par le transfert de personnel entre unités productives si les centres de production sont équipés pour accueillir un travailleur supplémentaire).

L'embauche d'un intérimaire doit être économiquement plus rentable que la solution de l'ajustement par les stocks. Dans notre exemple, cela revient à comparer le coût associé à l'embauche d'un travailleur temporaire pour 3.74 jours au coût de stockage de la production qu'on est obligé d'anticiper. En prenant $\alpha = 0.01$, le coût de stockage de cette production anticipée est de $0.01 \times 11.1 \times (850 \times 2 + 163 \times 3 + 233 \times 2) = 294.705$. Pour que l'embauche d'un temporaire constitue une meilleure option, il faudrait que son coût journalier soit inférieur à 294.705/3.74=78.8 euros, ce qui est impossible.

5. Règles d'approvisionnement (lotissement)

Jusqu'à présent nous avons considéré une seule règle d'approvisionnement, le lot pour lot qui consiste à s'approvisionner d'un montant égal aux besoins nets à chaque période.

L'algorithme de Wagner-Whitin (1958) utilise la programmation dynamique pour *optimiser* les lots d'un article considéré isolément (sous-section 5.1). Il existe également des méthodes dites heuristiques, plus simples, qui ne garantissent pas l'obtention d'une solution optimale. Nous présentons deux d'entre elles dans une deuxième sous-section. On adopte au préalable les notations suivantes.

S : coût fixe de lancement ;

h : coût de stockage par unité et par période ;

$D_{t,k} = \sum_{j=t}^{k} d_j$: demande cumulée (besoins bruts) pour les périodes t à k.

Le coût associé à un lot de taille $D_{t,k}$, noté $C(D_{t,k})$ est donné par $C(D_{t,k}) = S + H(D_{t,k}) = S + h\sum_{j=t}^{k}(j-t)d_j$, où $H(D_{t,k})$ est le coût de stockage associé au lot $x_t = D_{t,k}$.

On considère l'échéancier suivant de besoins bruts et l'on suppose $S = 1.668$ et $h = 0.0042$. Pour simplifier, les réceptions prévues et le stock initial sont supposés nuls de sorte que les besoins nets correspondent aux besoins bruts.

Période	1	2	3	4	5	6
d_t (besoins bruts)	46	30	82	90	56	17

5.1 L'algorithme de Wagner-Whitin (WW)

L'algorithme permet d'obtenir la solution optimale au problème de minimisation des coûts d'approvisionnement sur un horizon donné, sous contrainte de satisfaction de la demande (besoins nets). Formellement, le problème s'écrit

$$\min_{x_1,\ldots,x_T} \sum_{t=1}^{T} Sy_t + hI_t,$$

sous les contraintes

$$I_t = I_{t-1} + x_t - d_t$$
$$I_T = 0$$
$$x_t \in N$$
$$y_t = \begin{cases} 1 \text{ si } x_t > 0 \\ 0 \text{ sinon} \end{cases}$$

Comme la fonction de coût est concave, on montre qu'un approvisionnement optimal couvre toujours un nombre entier de demandes futures. Wagner et Within proposent une méthode de résolution du problème par la programmation dynamique. Le principe est le suivant : une politique optimale est nécessairement composée de sous-politiques optimales.

Soit $F(t)$, le coût de la politique optimale pour les périodes $1,2,\mathrm{K},t$. Ce coût s'écrit : $F(t) = \min_{k=0,\mathrm{K},t-1} \{S + H(D_{k+1,t}) + F(k)\}$, avec $F(0) = 0$. Ainsi, le coût de la politique optimale pour les périodes $1,2,\mathrm{K},t$ s'obtient en comparant les coûts associés aux alternatives suivantes.

- Couvrir les demandes des périodes 1 à t par un approvisionnement à la première période
- Couvrir les demandes des périodes 2 à t par un approvisionnement à la période 2 et adopter la politique optimale pour la période 1
- Couvrir les demandes des périodes 3 à t par un approvisionnement à la période 3 et adopter la politique optimale pour les périodes 1 et 2, etc.

L'application de l'algorithme à notre exemple est faite à la figure 13.

Période	1	2	3	4	5	6	S=1.668	h=0.0042
Besoins nets	**46**	**30**	**82**	**90**	**56**	**17**		

t	k	1	2	3	4	5	6	Coût de stock.	Coût de lanc.	F(k)	Total	F(t)
1	**0**	46						0	1.668	0	1.668	1.668
2	**0**	76	0					0.126	1.668	0	1.794	1.794
	1	**46**	30					0	1.668	1.668	3.336	
3	**0**	158	0	0				0.8148	1.668	0	2.4828	2.4828
	1	**46**	112	0				0.3444	1.668	1.668	3.6804	
	2	**76**	**0**	82				0	1.668	1.794	3.462	
4	**0**	248	0	0	0			1.9488	1.668	0	3.6168	3.6168
	1	**46**	202	0	0			1.1004	1.668	1.668	4.4364	
	2	**76**	**0**	172	0			0.378	1.668	3.462	5.508	
	3	**158**	**0**	**0**	90			0	1.668	2.4828	4.1508	
5	**0**	304	0	0	0	0		2.8896	1.668	0	4.5576	
	1	**46**	258	0	0	0		1.806	1.668	1.668	5.142	
	2	**76**	**0**	228	0	0		0.8484	1.668	1.794	4.3104	4.3104
	3	**158**	**0**	**0**	146	0		0.2352	1.668	2.4828	4.386	
	4	**248**	**0**	**0**	**0**	56		0	1.668	3.6168	5.2848	
6	**0**	321	0	0	0	0	0	3.2466	1.668	0	4.9146	
	1	**46**	275					2.0916	1.668	1.668	5.4276	
	2	**76**	**0**	245	0	0	0	1.0626	1.668	1.794	4.5246	4.5246
	3	**158**	**0**	**0**	163	0	0	0.378	1.668	2.4828	4.5288	
	4	**248**	**0**	**0**	**0**	73	0	0.0714	1.668	3.6168	5.3562	
	5	**76**	**0**	**228**	**0**	**0**	17	0	1.668	4.3104	5.9784	

Figure 13. Application de WW

Détaillons ce qui se passe à la période $t=3$. On a $F(3)=\min_{k=0,K,2}\{S+H(D_{k+1,3})+F(k)\}$, soit $F(3)=\min_{k=0,K,2}\{S+H(D_{1,3})+F(0);S+H(D_{2,3})+F(1);S+H(D_{3,3})+F(2)\}$. La première possibilité consiste à placer un lot unique à la période 1 permettant de couvrir les demandes des 3 périodes et adopter la politique optimale en 0. Le coût de cette option est donc égal à $S+H(D_{1,3})+F(0)$, avec $S=1.668$ puisqu'il n'y a qu'un seul lancement, $H(D_{1,3})=0.0042\times(30\times1+82\times2)=0.8148$ et $F(0)=0$. Le coût total associé à cette option est donc égal à 2.4828. La deuxième possibilité consiste à s'approvisionner en période 2 d'un lot couvrant les demandes des périodes 2 et 3 et à adopter la politique qui était optimale lorsqu'on ne considérait qu'une seule période de demande (dans ce cas là, pas beaucoup de choix, il s'agit d'un lot à la période 1 couvrant les besoins de cette période). Enfin, la troisième possibilité est de s'approvisionner à la période 3 pour couvrir la demande de cette période et adopter la politique optimale pour les 2 premières périodes. Cette politique consistait à s'approvisionner à la période 1 pour couvrir les demandes des périodes 1 et 2, pour un coût égal à $F(2)=1.794$. Finalement, quand on arrive à la période 6, on obtient la solution optimale qui consiste à s'approvisionner aux 1 et 3.

5.2 La méthode du moindre coût unitaire ou Least Unit Cost (LUC)

La méthode LUC consiste à choisir le lot $x_t=D_{t,k}$ vérifiant la condition $\frac{C(D_{t,k})}{D_{t,k}}<\frac{C(D_{t,k+1})}{D_{t,k+1}}$ pour $t\le k\le T$.

L'application de la méthode à notre exemple est faite à la figure 14 où l'on donne à la dernière colonne le coût moyen par demande. On sélectionne le lot qui minimise ce coût moyen. On obtient $x_1=D_{1,4}=248$ et l'on a $x_2=x_3=0$. On itère la procédure pour la période d'approvisionnement suivante, à savoir la période 5. On obtient finalement des lots aux périodes 1 et 5 pour un coût total qu'on peut lire à la Figure 13, à savoir 5.3562.

t	k	1	2	3	4	5	6	Coût de stock.	Coût de lanc.	Coût/demande
1	1	46						0	1.668	0.03626087
1	2	76						0.126	1.668	0.023605263
1	3	158						0.8148	1.668	0.015713924
1	4	248						1.9488	1.668	0.014583871
1	5	304						2.8896	1.668	0.014992105
1	6	321						3.2466	1.668	0.01531028
5	5					56		0	1.668	0.029785714
	6					73		0.0714	1.668	0.023827397

Figure 14. Application de la méthode LUC

5.3 La méthode de Silver-Meal

Cette méthode procède de la même logique que la précédente sauf qu'il s'agit ici de minimiser le coût moyen par période, pour les périodes que le lot permet de couvrir. Formellement, il s'agit de choisir le lot choisir le lot $x_t=D_{t,k}$ vérifiant la condition

$\frac{C(D_{t,k})}{k-(t-1)} < \frac{C(D_{t,k+1})}{k+1-(t-1)}$ pour $t \le k \le T$, où $k-(t-1)$ est le nombre de périodes que l'on peut couvrir avec le lot $x_t = D_{t,k}$.

L'application de la méthode à notre exemple consiste est faite à la figure 15. La demande cumulée $D_{1,3}$ vérifie la condition recherchée, on choisit donc $x_1 = D_{1,3} = 158$ et l'on a $x_2 = x_3 = 0$. On itère la procédure pour la période d'approvisionnement suivante, à savoir la période 4.

t	k	1	2	3	4	5	6	Coût de stock.	Coût de lanc.	Coût/période
1	1	46						0	1.668	1.668
1	2	76						0.126	1.668	0.897
1	3	158						0.8148	1.668	0.8276
1	4	248						1.9488	1.668	0.9042
4	4				90			0	1.668	1.668
4	5				146			0.2352	1.668	0.9516
4	6				163			0.378	1.668	0.682

Figure 15. Application de l'heuristique de Silver-Meal

6. Planification en horizon glissant

En horizon glissant, on suppose que la demande est connue avec certitude sur un nombre limité de périodes appelé fenêtre de prévision que l'on note W. Si t désigne la période courante, alors on va appliquer une technique d'approvisionnement donnée aux demandes des périodes $t, \ldots, t+W-1$ et obtenir un échéancier de lots sur cette fenêtre. Mais seul l'approvisionnement de la période courante sera mis en oeuvre. A la période courante suivante, à savoir $t+1$, on connaît les demandes des périodes $t+1$ à $t, \ldots, t+W$ sur lesquelles on applique à nouveau la technique d'approvisionnement choisie. On connaît donc une demande supplémentaire, celle de la période $t+W$ et il se peut alors que l'échéancier de lots diffère de celui obtenu à la période précédente. A nouveau, seul l'approvisionnement de la période courante est lancé.

Pour illustrer le raisonnement, on considère toujours le même exemple que précédemment et l'on suppose que la fenêtre de prévision est de 3 périodes, de sorte qu'à la période courante 1, on connaît seulement les demandes des périodes 1 à 3. Supposons que la technique d'approvisionnement retenue soit l'algorithme de Wagner-Whitin. La Figure 16 illustre la procédure de calcul des lots en horizon glissant.

L'application de WW aux 3 premières demandes conduit à un échéancier de lots égal à (158,0,0). On implémente le lot de la période courante, à savoir 158. A la période courante suivante, c'est à dire à la période 2, la demande de la période 4 est révélée. On recalcule les besoins nets sur la fenêtre de prévision et on obtient (0,0,90) pour ces besoins nets aux périodes 2 à 4. L'application de WW suggère des lots identiques à ces besoins nets mais seul le lot de la période courante est implémenté, à savoir 0. A la période courante 3, la demande de la période 5 est révélée. On recalcule les besoins nets sur la fenêtre (périodes 3 à 5) et on applique WW. On obtient l'échéancier de lots (0,146,0). Seul le lot de la période 3 est implémenté, à savoir 0. Le principe demeure le même à la période 4 ou WW suggère un approvisionnement de 163 unités qui est mis en oeuvre. Notons qu'en raison de la révélation d'une demande supplémentaire à chaque période, les lots suggérés par WW peuvent changer, ce qui est le cas du lot de la période 4, qui prend successivement les valeurs de 90 unités puis 146 puis 163 unités. Si l'on stoppe artificiellement l'horizon à 6 périodes et que l'on calcule le coût total sur cet horizon, on constate que ce coût, 4.5288, diffère du coût que l'on obtient

lorsqu'on applique WW à l'ensemble des 6 demandes (voir Figure 13) et qui s'élève à 4.5246. Il existe donc une perte d'optimalité en horizon glissant lié au fait que les demandes ne sont révélées que progressivement et que des décisions d'approvisionnement sont fixées avant que toute la demande soit connue.

	Période	0	1	2	3
Fenêtre = 3 périodes	**Demande (besoins bruts)**		46	30	82
Coût de lancement = 1.668	Besoins nets		**46**	**30**	**82**
Coût de stockage = 0.0042	Fin d'ordre		158	0	0

	Période	0	1	2	3	4
Fenêtre = 3 périodes	**Demande (besoins bruts)**		46	30	82	90
Coût de lancement = 1.668	Besoins nets		0	**0**	**0**	**90**
Coût de stockage = 0.0042	Fin d'ordre		158	0	0	90

	Période	0	1	2	3	4	5
Fenêtre = 3 périodes	**Demande (besoins bruts)**		46	30	82	90	56
Coût de lancement = 1.668	Besoins nets		0	**0**	**0**	**90**	56
Coût de stockage = 0.0042	Fin d'ordre		158	0	0	146	0

	Période	0	1	2	3	4	5	6
Fenêtre = 3 périodes	**Demande (besoins bruts)**		46	30	82	90	56	17
Coût de lancement = 1.668	Besoins nets		0	**0**	**0**	**90**	**56**	**17**
Coût de stockage = 0.0042	Fin d'ordre		158	0	0	163	0	0
	Stock final		112	82	0	73	17	0
	Coût de lancement		1.668	0	0	1.668		
	Coût de stockage		0.4704	0.3444	0	0.3066	0.0714	0
	Coût total de lancement	3.336						
	Coût total de stockage	1.1928						
	Coût total	4.5288						

Figure 16. Planification en horizon glissant

7. MRP et gestion des stocks multi-échelon

Il existe essentiellement deux catégories de politiques classiques de gestion des stocks : les politiques à révision continue et les politiques à révision périodique.

Une politique à révision continue suppose que le décideur connaît l'état exact des stocks pour passer des commandes à tout moment. Dans les politiques à révision périodique, au contraire, les stocks ne sont évalués qu'à intervalles réguliers et les commandes sont passées à des moments précis.

Ces politiques traditionnelles de gestion des stocks ne permettent pas de gérer de façon satisfaisante les stocks multi-échelon, pour deux raisons essentielles. Ces politiques supposent en effet que les différentes références peuvent être gérées indépendamment parce que les demandes sont indépendantes. En outre, elles s'appliquent à la gestion d'articles dont la demande est relativement stationnaire.

7.1 Gestion de stocks d'articles à demande dépendante

Une gestion indépendante des stocks multi-échelon serait lourde de conséquences en terme de probabilité de rupture de stock. A titre d'illustration, imaginons que l'on accepte une probabilité de rupture de 5%, pour chacun des stocks de composants qui entrent dans la fabrication des lunettes. La probabilité pour que ces 7 composants soient disponibles simultanément est égale à $0{,}95^7 \approx 0{,}70$. Autrement dit, il n'y a que 70% de chance de pouvoir assembler les lunettes. En revanche, le principe même de la méthode MRP implique que les besoins nets en composants sont déterminés suffisamment à l'avance, pour qu'une rupture de stock ne puisse pas se produire.

7.2 Irrégularité de la demande

Les modèles de gestion à point de commande sont satisfaisants pour la gestion du stock des articles dont la demande est stationnaire. Dans le cas des stocks multi-échelon, la demande en composants connaît des fluctuations qui s'amplifient à mesure que l'on descend dans les niveaux de nomenclature. En effet, même si la demande en produits finis est stationnaire, celle en composants ne l'est plus du fait du lancement de la fabrication par lots. A titre d'illustration, considérons la Figure 17.

	Période	1	2	3	4	5	6
LUNETTES	Besoins bruts	50	50	50	50	50	50
	Fin d'ordre	100	0	100	0	100	0

	Période	1	2	3	4	5	6
MONTURES	Besoins bruts	100	0	100	0	100	0
	Fin d'ordre	200	0	0	0	100	0

	Période	1	2	3	4	5	6
BRANCHES	Besoins bruts	400	0	0	0	200	0
	Fin d'ordre	600	0	0	0	0	0

Figure 17. Exemple d'irrégularité de la demande générée par les lots

Pour simplifier, nous supposons que les délais sont nuls. Les fins d'ordres résultent de l'application d'une règle d'approvisionnement donnée. Comme on peut le constater, la demande en lunettes est stationnaire, mais celle en branches est fortement irrégulière.

8. Conclusion

L'objectif de la méthode MRP ne réside pas dans l'obtention d'une solution globalement optimale au problème de minimisation des coûts d'approvisionnement (de tous les articles considérés simultanément), des délais de livraison, etc., notamment sous les contraintes de capacité de production et de satisfaction de la demande.

Elle cherche une solution réalisable qui, contrairement à la solution optimale, peut être obtenue dans un temps raisonnable. En effet, ce problème d'optimisation est d'une complexité telle que sa résolution exacte n'est pas envisageable. De plus, l'instabilité de l'environnement obligerait à reconsidérer sans cesse cet optimum. Néanmoins, ceci n'exclut pas le recours à des méthodes optimales, pour résoudre des problèmes partiels comme l'optimisation des coûts d'approvisionnement d'un seul article.

Le MRP part d'un constat, celui de l'inadéquation des politiques classiques de gestion des stocks au cas des stocks multi-échelon. Une politique de gestion des stocks se définit par les réponses qu'elle apporte aux questions suivantes : quand doit on s'approvisionner, et quel doit être le volume de l'approvisionnement. De nombreux modèles d'approvisionnement ont été conçus pour répondre aux exigences du cadre analytique dans lequel se situe le MRP (demandes discrètes, déterministes, non stationnaires, etc.). Fondamentalement, le MRP utilise des méthodes de groupage des besoins nets qui consistent à minimiser certains critères de coûts, pour chaque article considéré isolément. Ces méthodes, optimales ou non au regard de la gestion d'un seul article, ne sauraient garantir la solution globalement optimale, car elles ignorent les interdépendances des produits. Néanmoins, elles sont susceptibles de fournir des solutions plus satisfaisantes en termes de coût, que les séquences d'approvisionnements obtenues à l'issue de la seule planification des besoins en composants

INTRODUCTION AUX PROBLÈMES D'ORDONNANCEMENT

Le problème d'ordonnancement consiste à organiser dans le temps la réalisation d'un ensemble de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînements, etc.) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises.

Après avoir donné quelques définitions permettant de mieux caractériser les variantes du problème posé (section 1), nous nous intéresserons à l'ordonnancement de projets (section 2), puis nous abordons les problèmes d'ordonnancement en ateliers spécialisés (section 3). Dans une troisième section nous aborderons les problèmes d'ordonnancement de projets.

1. Quelques définitions

Un problème d'ordonnancement consiste à guider la conception et le contrôle d'une *réalisation* pouvant être de nature très variée. Il peut s'agir d'un grand ensemble (projet immobilier, navire, etc.), de l'organisation d'un emploi du temps, etc.

On appelle tâche j (ou job) un élément de réalisation, avec t_j la date de début de la tâche ; C_j, sa date de fin (completion time) et p_j, son temps opératoire (processing time). Chaque tâche peut être caractérisée par des éléments additionnels comme :

- une date de début au plus tôt $\underline{t}_j$ (release date)
- une date de fin souhaitée d_j (due date) ;
- une date de fin obligatoire D_j (deadline).

Les tâches peuvent être soumises à des contraintes qu'il est nécessaire de respecter pour trouver un ordonnancement réalisable, c'est-à-dire un ordre dans lequel ces tâches pourront être exécutées sans qu'il y ait violation de ces contraintes. Ces contraintes peuvent être de plusieurs natures.

Les contraintes de type potentiel sont celles qui peuvent se mettre sous la forme $t_j - t_i \geq a_{ij}$ où a_{ij} est une constante. Elles permettent de traduire par exemple les contraintes suivantes.

- Succession de deux tâches. La tâche j ne peut débuter qu'après l'exécution de la tâche i. Ce qui s'écrit $t_j \geq t_i + p_i$, soit $t_j - t_i \geq p_i$ pour retrouver la forme précédente. Cette contrainte est encore appelée contrainte de succession temporelle (par exemple, on ne peut faire les fondations si le terrassement n'est pas fini).
- Succession de deux tâches sans interruption. La tâche j commence dès que la tâche i est terminée, ce qui s'écrit $t_j = t_i + p_i$.
- Succession de deux tâches avec un intervalle de battement noté θ_{ij} : $t_j \geq t_i + p_i + \theta_{ij}$, soit $t_j - t_i \geq p_i + \theta_{ij}$.
- La tâche j ne peut commencer avant que la tâche i ait atteint un degré d'avancement suffisant. Soit $t_j \geq t_i + \alpha p_i$, avec $0 < \alpha < 1$.

- La tâche j ne peut commencer avant une date donnée : $t_j \geq \underline{t}_j + \theta$
- La tâche j doit être terminée avant une date donnée : $t_j + p_j \leq d_j$

Les deux dernières contraintes sont encore appelées contraintes de localisation temporelles.

Les contraintes de type disjonctif imposent à certaines tâches d'être exécutées pendant des intervalles de temps disjoints. Ces contraintes apparaissent lorsque deux tâches distinctes d'une même réalisation nécessitent la présence d'un moyen unique. Par exemple une seule grue sur un chantier, une seule fraiseuse dans un atelier, etc. Ces contraintes s'écrivent $t_j - t_i \geq p_i$ ou bien $t_i - t_j \geq p_j$ (il n'y a pas de contrainte de succession).

Les contraintes de type cumulatif associent à chaque tâche j les besoins en matériel, en budget ou en main d'oeuvre requis pour sa réalisation. Ces besoins peuvent varier au cours du temps. Ces contraintes sont généralement assez souples puisqu'un budget peut être dépassé (dans la limite du raisonnable) et que l'on peut recourir à une main d'oeuvre temporaire en cas d'insuffisance de la main d'oeuvre permanente. C'est la souplesse de ces contraintes cumulatives qui permet de les distinguer des contraintes disjonctives. Si par exemple un ouvrier ne peut accomplir deux tâches simultanément, la prise en compte de limitations de main d'oeuvre doit s'exprimer à l'aide de contraintes disjonctives.

Ainsi, les ressources disjonctives ne peuvent exécuter qu'une tâche à la fois tandis que les ressources cumulatives peuvent exécuter un nombre limité d'opérations simultanément.

Ordonnancer, c'est trouver un ordre de traitement réalisable des tâches, c'est-à-dire qui respecte l'ensemble des contraintes associées au problème posé. Cet ordre n'est pas nécessairement unique et le choix d'un ordonnancement particulier parmi l'ensemble des ordonnancements réalisables peut se faire en s'appuyant sur différents critères comme

- Le retard algébrique (lateness) : $L_j = C_j - d_j$ (différence entre la date de fin effective et la date de fin souhaitée, cette différence pouvant être négative dans le cas ou la tâche s'achève avant la date souhaitée) ;
- Le retard absolu (tardiness) : $T_j = \max(0, C_j - d_j)$;
- L'avance (earliness) : $E_j = \max(0, d_j - C_j)$;
- La pénalité de retard $U_j = 0$ si $C_j \leq d_j$ et $U_j = 1$ sinon ;
- La durée de séjour dans l'atelier : $F_j = C_j - \underline{t}_j$.

2. Ordonnancement de projets

Le problème consiste à déterminer l'ordre dans lequel doivent s'enchaîner les tâches de sorte à minimiser la durée totale d'exécution du projet, sous différentes contraintes de type potentiel, disjonctif ou cumulatif comme nous les avons décrites dans la section précédente. Un problème d'ordonnancement avec des contraintes de localisation temporelle et de succession temporelle est appelé problème central d'ordonnancement.

2.1 Formulation du problème central d'ordonnancement et représentation Potentiel-Tâches

On rappelle que t_j désigne la date de début de la tâche j avec $j = 1,\mathsf{K},n$ et p_j, son temps opératoire (processing time) ou durée d'exécution. En associant artificiellement la tâche 0 à l'événement "début du projet" et la tâche $n+1$ à l'événement "fin du projet", alors la fonction objectif s'écrit $\min t_{n+1} - t_0$ et les contraintes de succession d'un ensemble S de tâches concernées s'écrivent $t_j \geq t_i + p_i \quad \forall (i,j) \in S$. On dit que la tâche i est un ancêtre de la tâche j. On note $\Gamma^{-1}(j)$ l'ensemble des ancêtres de la tâche j et $\Gamma(j)$ l'ensemble des tâches successeurs de j.

Le projet peut se représenter sous la forme d'un graphe dont les noeuds sont les tâches et dont les arcs (flèche reliant un noeud à un autre) symbolisent les relations de succession entre couples de tâches concernées. Ainsi, l'ensemble S représente les arcs du graphe. Tout arc orienté (i,j) est valué par la durée d'exécution p_i de la tâche i, encore appelée longueur de l'arc (i,j)

Prenons l'exemple de projet dont les données figurent dans le Tableau 1 ; les durées de chaque tâche étant exprimées en jours.

Tâche j	Désignation	Durée p_j	Ancêtres i
1	Terrassement	5	-
2	Fondations	4	1
3	Colonnes porteuses	2	2
4	Charpente	2	3
5	Couverture	3	4
6	Maçonnerie	5	3
7	Plomberie	3	2
8	Coulage dalle	3	7
9	Chauffage	4	8 et 6
10	Plâtre	10	9 et 5
11	Finitions	5	10

Tableau 1. Un exemple de projet

2.1.1. Classement des tâches par niveaux

Pour tracer correctement le graphe potentiel-tâches, c'est-à-dire pour affecter une position appropriée dans l'espace à chacune des tâches de sorte à avoir une représentation claire, on affecte ces tâches à des niveaux, qui représentent des étapes de réalisation du projet selon **l'algorithme de classement des tâches par niveau** constitué des étapes suivantes.

- **Etape 1**. Placer au niveau $k = 0$ les tâches qui n'ont aucun ancêtre (tâches j telles que $\Gamma^{-1}(j) = \varnothing$). Supprimer ces tâches de la liste de tâches à affecter. Poser $k := k+1$
- **Etape 2.** Affecter au niveau k les tâches ayant pour ancêtres les tâches ayant été affectées au niveau $k-1$. Supprimer ces tâches de la liste de tâches à affecter. Poser $k := k+1$
- **Etape 3.** S'il reste des tâches dans la liste des tâches à affecter, retourner à l'étape 2. Sinon la décomposition des tâches en niveaux est terminée.

L'application de l'algorithme à notre exemple du Tableau 1 est faite à la Figure 1. La tâche qui n'a pas d'ancêtre est la tâche 1 que l'on place au niveau 0. A l'itération suivante, parmi les tâches restant à classer, on sélectionne celles qui ont la tâche 1 comme ancêtre. La tâche

2 est donc affectée au niveau 2, puisque $\Gamma^{-1}(2)=1$. A l'itération 3, les tâches restant à classer et ayant la tâche 2 comme ancêtres sont les tâches 3 et 7 ($\Gamma^{-1}(3)=\Gamma^{-1}(7)=2$) qui viennent donc se placer au niveau 2. A l'itération 4, les tâches 4 et 6 ont la tâche 3 pour ancêtre et la tâche 8 à l'ancêtre 7. Les tâches 4, 6 et 8 se placent dont au niveau 3, etc.

Itération 1	Tâches à affecter	1	2	3	4	5	6	7	8	9	10	11
	Ancêtres		1	2	3	4	3	2	7	8 6	9 5	10
	Tâches Niveau 0	**1**										
Itération 2	Tâches à affecter		2	3	4	5	6	7	8	9	10	11
	Ancêtres		**1**	2	3	4	3	2	7	8 6	9 5	10
	Tâches Niveau 1		**2**									
Itération 3	Tâches à affecter			3	4	5	6	7	8	9	10	11
	Ancêtres			**2**	3	4	3	**2**	7	8 6	9 5	10
	Tâches Niveau 2			**3**				**7**				
Itération 4	Tâches à affecter				4	5	6		8	9	10	11
	Ancêtres				**3**	4	**3**		**7**	8 6	9 5	10
	Tâches Niveau 3				**4**		**6**		**8**			
Itération 5	Tâches à affecter					5				9	10	11
	Ancêtres					**4**				**8** **6**	9 5	10
	Tâches Niveau 4					**5**				**9**		
Itération 6	Tâches à affecter										10	11
	Ancêtres										**9** **5**	10
	Tâches Niveau 5										**10**	
Itération 7	Tâches à affecter											11
	Ancêtres											**10**
	Tâches Niveau 6											11

Figure 1. Exemple d'application de l'algorithme de classement des tâches par niveau

Pour obtenir le graphe potentiel-tâches, on commence par placer les tâches du niveau 0, puis celles du niveau 1 etc., jusqu'au niveau le plus élevé. Puis, à l'aide de la colonne indiquant les ancêtres de chaque tâche, on relie les tâches à leurs ancêtres par un arc valué par le temps d'exécution de la tâche. Dans notre exemple, on obtient le graphe potentiel-tâches de la Figure 2.

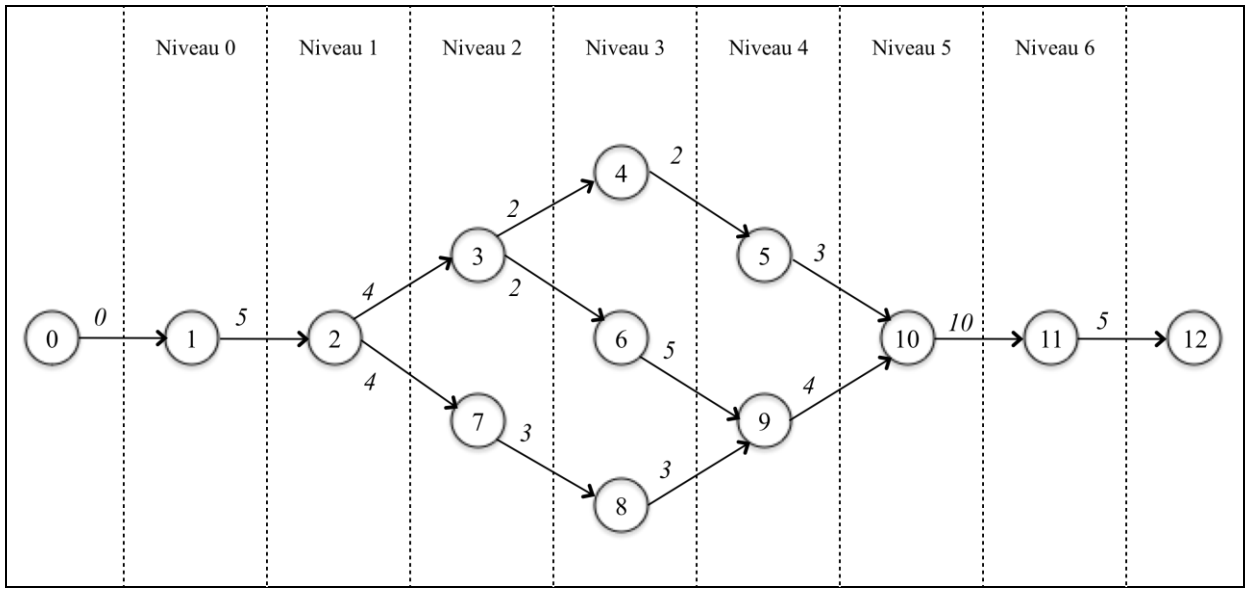


Figure 2. Un exemple de graphe Potentiel-Tâches

2.1.2. Représentation d'autres types de contraintes

Supposons que la tâche 3 ne puisse commencer qu'à la date 10. Ceci s'écrit $t_3 \geq 10$, soit $t_3 \geq t_0 + 10$. On représente cette contrainte par un arc joignant les noeuds 0 à 3 et valué par 10.

Si la tâche 5 doit être commencée avant la date 40, c'est-à-dire si $t_5 \leq 40$, soit $t_5 \leq t_0 + 40$ alors on représente cette contrainte par un arc joignant les noeuds 5 à 0 et valué par -40.

Enfin, supposons que la tâche 9 doivent débuter au plus tard 5 jours après le début de la tâche 8, c'est-à-dire $t_9 \leq t_8 + 5$. On représente cette contrainte par un arc joignant le noeud 9 au noeud 8 et valué par -5.

La Figure 3 donne la représentation de ces contraintes supplémentaires.

Plus généralement, avec a,b,c trois entiers strictement positifs, on a le tableau suivant.

Contrainte	Représentation
$t_j \geq a \Leftrightarrow t_j \geq t_0 + a$	Arc $(0,j)$ valué par a
$t_j \leq b \Leftrightarrow t_j \leq t_0 + b$	Arc $(j,0)$ valué par $-b$
$t_j \leq t_i + c$	Arc (j,i) valué par $-c$

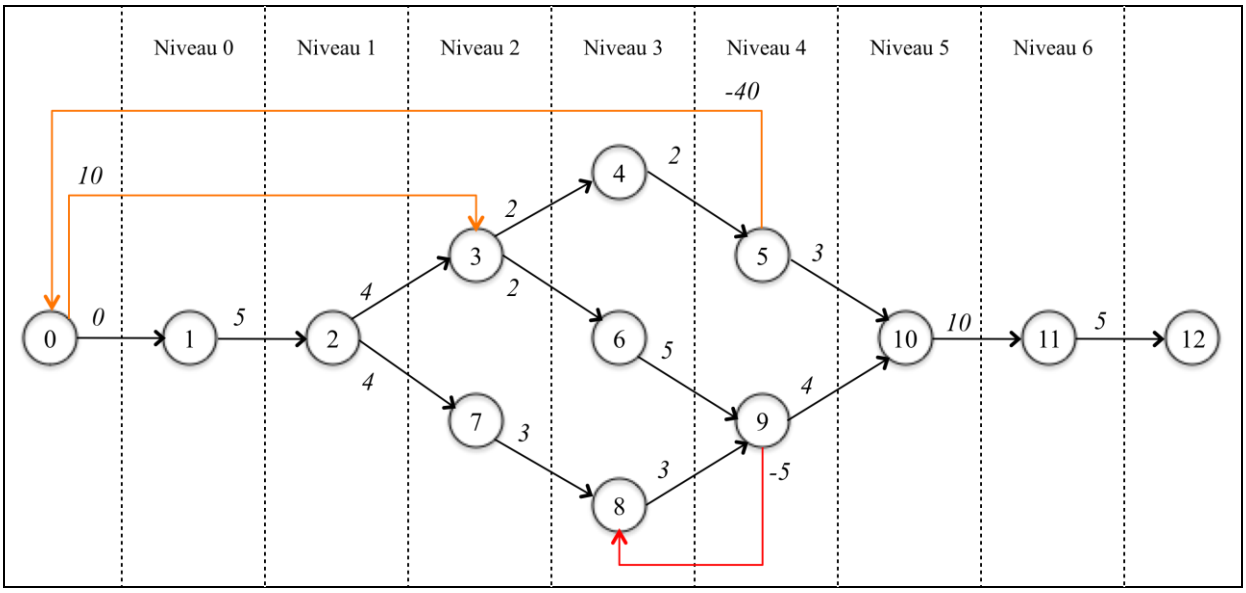
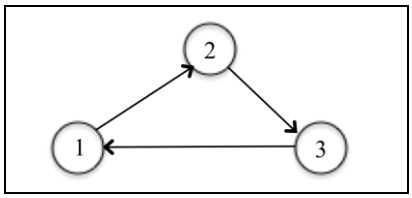


Figure 3. Autres types de contraintes

2.1.3. Conditions de réalisabilité d'un ordonnancement

On peut montrer que les contraintes temporelles sont compatibles entre elles si et seulement si le graphe associé ne comporte aucun circuit de longueur positive. Un circuit est un sous-ensemble d'arcs partant du noeud i pour revenir au noeud i. La longueur du circuit est simplement la somme des longueurs des arcs le constituant.

Pour illustrer le raisonnement, supposons que nous ayons la situation suivante. La tâche 1 qui dure p_1 jours doit être terminée avant que la tâche 2 ne commence. Ceci s'écrit $t_2 \geq t_1 + p_1$ et se représente par un noeud 1 relié au noeud 2 par un arc de longueur p_1. La tâche 2 qui dure p_2 jours doit être terminée avant que la tâche 3 commence. Graphiquement, on a un arc de longueur p_2 reliant les noeuds 2 et 3. La tâche 3 qui dure p_3 jours doit être terminée avant que la tâche 1 commence. On a un arc de longueur p_3 reliant 3 à 1. Cette situation est représentée à la figure suivante.



Ecrivons les contraintes correspondantes : $t_1 + p_1 \leq t_2$; $t_2 + p_2 \leq t_3$; $t_3 + p_3 \leq t_1$. En sommant et en simplifiant on obtient $p_1 + p_2 + p_3 \leq 0$.

En revanche, la présence d'un circuit de longueur négative ne pose pas de problème, comme c'est le cas à la Figure 3 pour le cycle (8,9,8) de longueur -2. Les contraintes correspondantes s'écrivent $t_8 + 3 \leq t_9$ et $t_9 - 5 \leq t_8$. En les combinant on obtient par exemple $t_9 - 5 \leq t_8 \leq t_9 - 3$ ce qui constitue une condition réalisable.

2.2 Détermination de l'ordonnancement au plus tôt et au plus tard, marges libres et chemin critique

Ordonnancement au plus tôt. Partant du noeud 0 qui marque le début du chantier, la date de début au plus tôt $\underline{t}_j$ d'une tâche j se définit par $\underline{t}_j = \max_{i\in\Gamma^{-1}(j)} \{\underline{t}_i + p_i\}$, pour $j = 0,\ldots,n+1$. Ainsi une tâche peut démarrer au plus tôt lorsque la plus longue des tâches qui la précède est terminée, et lorsque toutes ces tâches ancêtres ont elles-mêmes démarré au plus tôt. On note que $\underline{t}_{n+1}$ donne la durée d'exécution du projet pour un ordonnancement au plus tôt.

L'application de cette règle à notre exemple de la Figure 2 donne $\underline{t}_1 = \underline{t}_0 + p_0 = 0$, puis $\underline{t}_2 = \underline{t}_1 + p_1 = 5$, puis $\underline{t}_3 = \underline{t}_2 + p_2 = 9$; $\underline{t}_7 = \underline{t}_2 + p_2 = 9$; $\underline{t}_4 = \underline{t}_6 = \underline{t}_3 + p_3 = 11$; $\underline{t}_8 = \underline{t}_7 + p_7 = 12$; $\underline{t}_5 = \underline{t}_4 + p_4 = 13$. La tâche 9 a deux prédécesseurs $\Gamma^{-1}(9) = \{6,8\}$, donc $\underline{t}_9 = \max\{\underline{t}_6 + p_6,\ \underline{t}_8 + p_8\}$ c'est-à-dire $\underline{t}_9 = \max\{11+5, 12+3\} = 16$. Puis $\underline{t}_{10} = \max\{\underline{t}_5 + p_5,\ \underline{t}_9 + p_9\} = 20$; $\underline{t}_{11} = \underline{t}_{10} + p_{10} = 30$. Enfin $\underline{t}_{12} = \underline{t}_{11} + p_{11} = 35$. La durée totale d'exécution du projet est donc de 35 jours lorsqu'on effectue un ordonnancement au plus tôt.

Ordonnancement au plus tard. On note $\bar{t}_j$ la date de début au plus tard de la tâche j. Il s'agit de la date limite à laquelle on peut encore commencer une tâche sans que cela affecte la durée totale d'exécution du projet résultant de l'ordonnancement au plus tôt. Ainsi, certaines tâches peuvent être retardées sans que cela allonge la date de fin du projet. Pour déterminer les dates de début au plus tard des tâches, on commence par la fin du projet pour remonter au début. Il est clair que pour la tâche fictive 12 symbolisant la fin du projet, on a $\bar{t}_{12} = \underline{t}_{12} = 35$ car retarder la tâche fictive "fin du projet", revient à allonger la durée totale d'exécution du projet. La tâche 11 peut s'achever au plus tard à la date où on peut faire démarrer au plus tard la tâche 12, c'est-à-dire $\bar{t}_{11} + p_{11} = \bar{t}_{12}$, soit $\bar{t}_{11} = 35 - 5 = 30$. De même, pour la tâche 10, on a $\bar{t}_{10} + p_{10} = \bar{t}_{11}$, soit $\bar{t}_{10} = 30 - 10 = 20$. La tâche 5 qui précède la tâche 10 doit s'achever au plus tard quand on peut commencer la tâche 10 au plus tard, c'est-à-dire $\bar{t}_5 + p_5 = \bar{t}_{10}$ soit $\bar{t}_5 = 20 - 3 = 17$. Pour la tâche 9 on obtient $\bar{t}_9 = 20 - 4 = 16$. Un raisonnement similaire pour les tâches 4, 6 et 8 conduit à $\bar{t}_4 = 17 - 2 = 15$; $\bar{t}_6 = 16 - 5 = 11$; $\bar{t}_8 = 16 - 3 = 13$. La tâche 3 a deux tâches qui lui succèdent : la tâche 4 qui peut démarrer au plus tard en 15 et la tâche 6 qui peut démarrer au plus tard à la date 11. Il faut donc que la tâche 3 se termine au plus tard à la date 11, c'est-à-dire $\bar{t}_3 + p_3 = \min(\bar{t}_4, \bar{t}_6)$, soit $\bar{t}_3 = \min(15,11) - 2 = 9$. Par un raisonnement similaire, on poursuit la détermination de toutes les dates de début au plus tard des tâches.

La généralisation donne $\bar{t}_j = \min_{i\in\Gamma(j)} \{\bar{t}_i\} - p_j$, pour $j = n+1,\ldots,0$ et $\bar{t}_{n+1} = \underline{t}_{n+1}$.

Tâche critique, marge libre et chemin critique. On appelle tâche critique toute tâche dont le retard affecte la fin au plus tôt du projet. Toute tâche critique j est telle que sa date de début au plus tard correspond à sa date de début au plus tôt, c'est-à-dire $\bar{t}_j = \underline{t}_j$. La marge libre m_j d'une tâche j se définit comme la différence entre sa date de début au plus tard et sa date de début au plus tôt, soit $m_j = \bar{t}_j - \underline{t}_j$. Cette marge correspond à la valeur maximale du retard autorisé pour débuter une tâche sans que cela retarde la fin du projet. Cette marge est nulle pour les tâches critiques, et positive pour celles qui ne le sont pas. Enfin, on appelle chemin critique, l'ensemble des tâches critiques qui représentent les tâches à surveiller si l'on veut respecter le délai minimum de réalisation du projet. La somme des longueurs des arcs constituant le chemin critique donne la durée minimum d'exécution du projet.

Dans notre exemple, la tâche 5 peut commencer au plus tôt à la date $\underline{t}_5 = 13$ et doit commencer au plus tard à la date $\bar{t}_5 = 17$, on peut donc retarder le début de la tâche de 4 jours

au maximum par rapport à sa date de début au plus tôt. On dispose donc d'une marge de 4 jours pour cette tâche. En revanche, la tâche 9 qui doit s'achever au plus tard à la date 20 pour commencer au plus tard la tâche 10 à cette même date est une tâche critique car elle s'achève à la date 20. Sa marge libre est nulle. La Figure 4 représente le chemin critique.

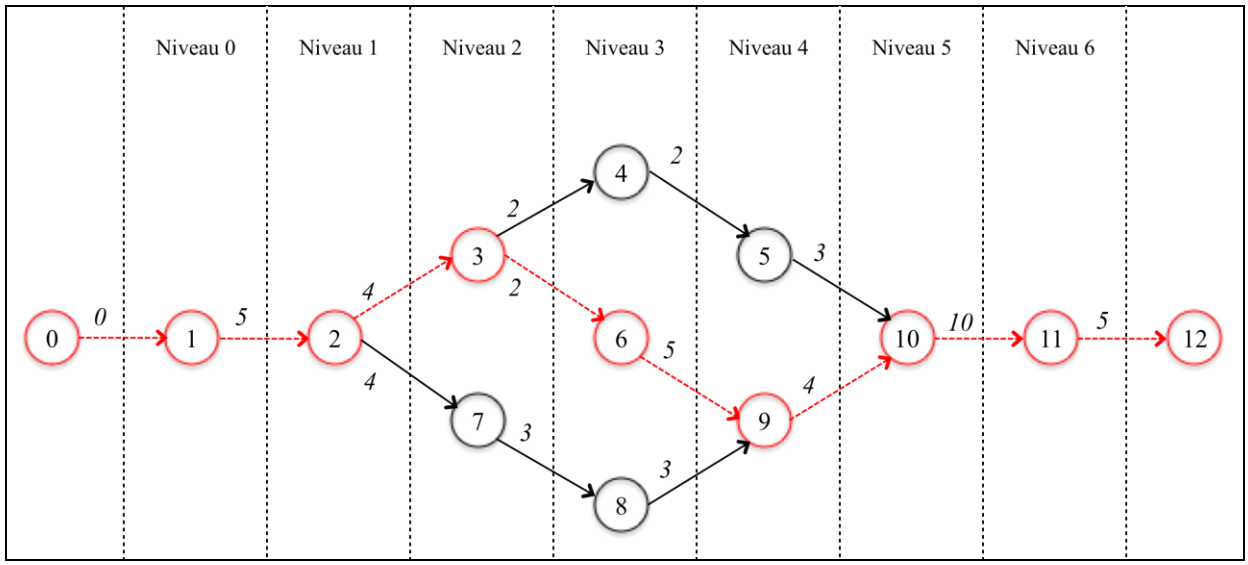


Figure 4. Chemin critique

2.3 L'ordonnancement par la méthode P.E.R.T. (Program Evaluation Review Technique)

La méthode PERT s'est développée parallèlement à la méthode Potentiel-Tâches et s'en distingue par le fait que les tâches ne sont plus associées aux noeuds mais aux arcs du réseau (on appelle réseau tout graphe orienté). L'algorithme de résolution pour la méthode PERT est très semblable à celui de la méthode Potentiel-Tâches. La différence majeure réside donc dans la construction du graphe PERT, souvent plus difficile que celui de la méthode Potentiel-Tâches.

Dans la méthode PERT, on associe à chaque tâche un arc de longueur égale à sa durée d'exécution. Les noeuds sont utilisés pour traduire les relations de succession temporelle. La construction du graphe PERT pose divers problèmes qui amènent à ajouter des arcs fictifs qui ne correspondent à aucune tâche. Le cas se rencontre pour les contraintes de succession temporelle. Supposons par exemple que la tâche 1 précède les tâches 2 et 3 et que la tâche 4 précède la tâche 3. On peut donner une représentation de ce cas à la Figure 5. Cependant le graphe de la Figure 5 dit que la tâche 4 précède la tâche 2, ce qui n'est pas le cas.

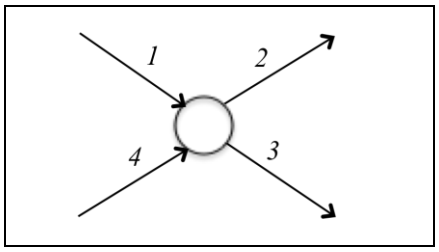


Figure 5. PERT incohérent

Pour résoudre ce problème, on doit ajouter un noeud fictif permettant de dissocier la tâche 2 de la tâche 4 et on doit ajouter un arc fictif entre le noeud existant et le noeud fictif pour traduire le lien entre les tâches 1 et 3. Cet arc fictif qui est donc une tâche fictive a une durée d'exécution nulle. Ceci est représenté à la Figure 6.

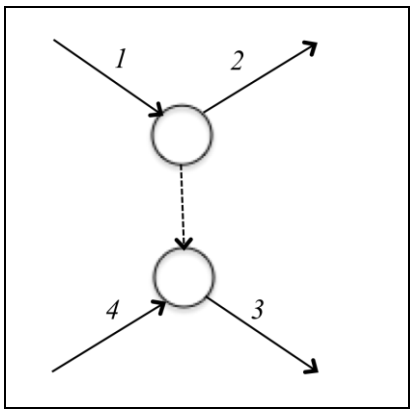


Figure 6. PERT cohérent

Outre sa lisibilité plus complexe, le graphe PERT nécessite souvent l'introduction de nombreux arcs fictifs. On préfèrera la méthode Potentiel-Tâches.

2.4 Prise en compte des ressources mobilisées et minimisation des coûts

Jusqu'à présent on a considéré la durée de chaque tâche comme une donnée. Or la durée d'une tâche particulière peut varier en fonction des ressources qu'elle mobilise, comme l'emploi de travailleurs supplémentaires, l'achat ou la location de matériel additionnel. En général, il est donc possible de diminuer la durée d'exécution d'une tâche moyennant la mobilisation de ressources supplémentaires qui supposent un coût additionnel. Nous allons examiner ici comment il est possible de réaliser un arbitrage entre la durée des tâches et leur coût.

Supposons que toute tâche j possède une durée minimum d'exécution (incompressible), notée $\underline{p}_j$ et une durée maximum notée $\overline{p}_j$. Si l'on admet que le coût associé à la durée est une fonction décroissante et linéaire de cette durée, alors la fonction de coût s'écrit

$$c_j(p_j) = c_j(\underline{p}_j) + \alpha_j(p_j - \underline{p}_j),$$

ce qui donne pour la durée maximale

$$c_j(\bar{p}_j) = c_j(\underline{p}_j) + \alpha_j(\bar{p}_j - \underline{p}_j).$$

La pente de la fonction de coût est donnée par

$$\alpha_j = \frac{c_j(\bar{p}_j) - c_j(\underline{p}_j)}{\bar{p}_j - \underline{p}_j}$$

L'objectif de minimisation des coûts associés aux durées des tâches s'écrit

$$\min z = \sum_{j=1}^{n} \left[c_j(\underline{p}_j) + \alpha_j(p_j - \underline{p}_j) \right] = K + \sum_{j=1}^{n} \alpha_j p_j,$$

Le terme K étant constant et en remplaçant α_j par sa valeur, on obtient finalement

$$\min z = \sum_{j=1}^{n} \frac{c_j(\bar{p}_j) - c_j(\underline{p}_j)}{\bar{p}_j - \underline{p}_j} \cdot p_j,$$

Les p_j étant cette fois-ci des variables pour le problème. Les variables du problème sont donc les suivantes

- t_j la date de début de la tâche j pour $j = 1, \ldots, n+1$ (on rappelle que t_{n+1} est la date de fin du projet, donc la durée d'exécution totale du projet) ;
- p_j, la durée d'exécution de la tâche j.

Le problème de minimisation des coûts des tâches se formule donc comme suit

$$\min z = \sum_{j=1}^{n} \frac{c_j(\bar{p}_j) - c_j(\underline{p}_j)}{\bar{p}_j - \underline{p}_j} \cdot p_j$$

$$s.c. \begin{cases} t_j \geq t_0 & \text{(localisation temporelle)} \\ t_j \geq t_i + p_i & \text{(succession temporelle)} \\ t_{n+1} \geq t_j + p_j & \text{(fin de projet)} \\ \underline{p}_j \leq p_j \leq \bar{p}_j & \text{(bornes sur la durée)} \\ t_{n+1} \leq T & \text{(borne sur la fin de projet)} \end{cases}$$

La borne sur la fin de projet a du être ajoutée sans quoi, étant donné l'objectif de minimisation des coûts associés aux durées, le processus d'optimisation conduirait à proposer $p_j = \bar{p}_j, \quad \forall j$.

3. Ordonnancement en ateliers spécialisés

Un centre de production désigne une usine, un département, un atelier, un groupe de machines ou une machine. Il peut être toute autre chose si la réalisation n'est pas un produit fabriqué mais une prestation de service : caisse de supermarché, centre de tri postal.

Une classification répandue des problèmes d'ordonnancement en ateliers se fonde sur les différentes configurations des centres de production (machines). Les modèles les plus connus sont ceux d'une machine unique, de machines parallèles, d'un atelier à cheminement unique ou d'un atelier à cheminement multiple.

3.1 Typologie des centres de production

Machine unique. L'ensemble des tâches nécessite pour leur exécution le passage sur un centre de production unique (Figure 7). Ce problème prend son intérêt lorsque le centre de production constitue un goulot d'étranglement qui peut influencer l'ensemble du processus de production. Le problème peut alors se restreindre à l'étude de ce centre uniquement.

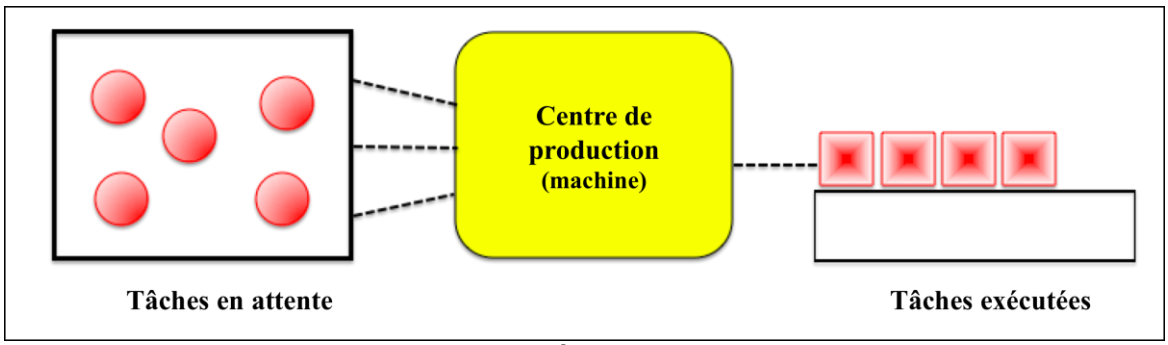


Figure 7. Ordonnancement de n tâches sur un unique centre de production

Machines parallèles. Dans cette configuration (Figure 8), on dispose de m centres de production identiques. L'ordonnancement s'effectue alors en deux phases : la première phase consiste à affecter les tâches aux machines et la deuxième phase consiste à établir la séquence de réalisation de ces tâches sur chaque machine.

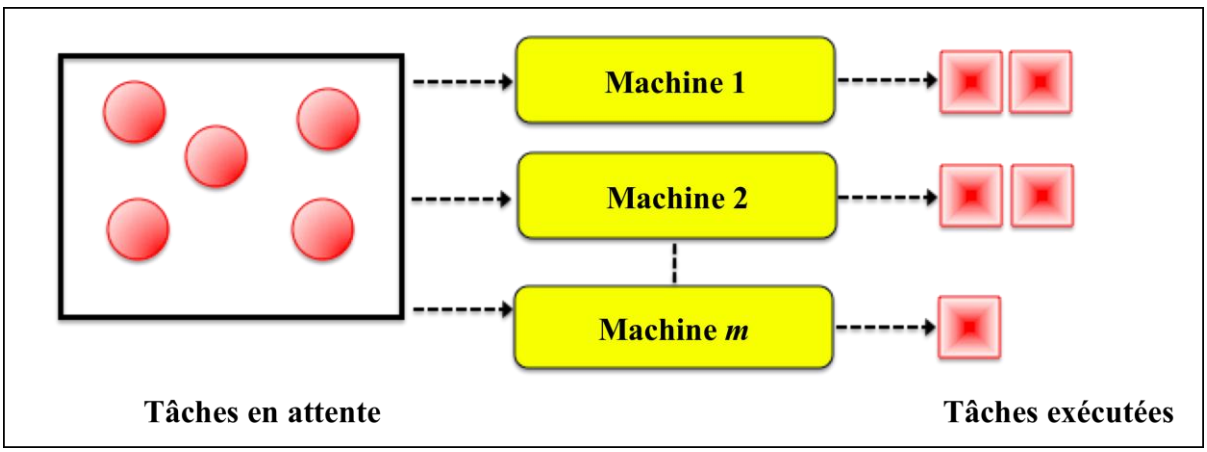


Figure 8. Ordonnancement de n tâches sur m machines parallèles identiques

Atelier à cheminement unique (Flow Shop). Il s'agit d'un atelier où les étapes de transformation sont identiques pour tous les produits fabriqués (voir Figure 9). Les machines peuvent être dédiées à une opération précise, et sont implantées en fonction de leur séquence d'intervention dans la gamme de production. Ce type d'atelier se caractérise par une grande productivité et une faible flexibilité et permet la fabrication de produits faiblement diversifiés.

Dans ce type d'atelier, l'objectif généralement retenu consiste à trouver une séquence de tâches qui minimise le temps total de production.

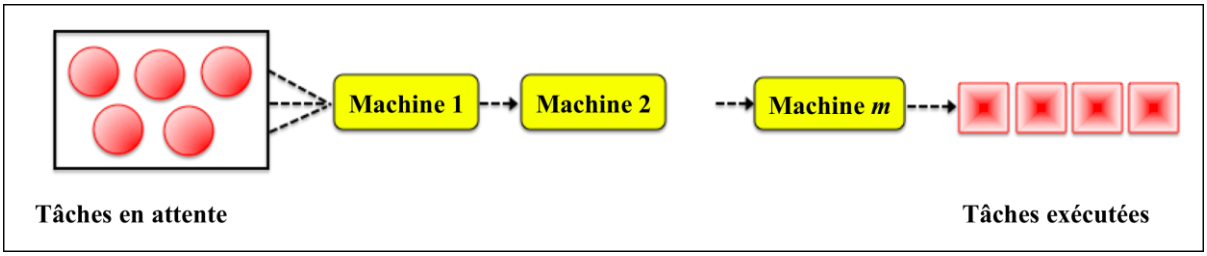


Figure 9. Ateliers à cheminement unique - Flow Shop

Une "généralisation" de ce type d'ateliers est donnée par **les flow shops hybrides** ou « atelier à cheminement unique avec machines en exemplaires multiples » qui comme leur nom l'indique font coexister plusieurs machines du même type en parallèle.

Ateliers à cheminements multiples (Job Shop). Les ateliers à cheminements multiples traitent une variété de produits dont la production requiert plusieurs types de machines dans des séquences variées mais selon un ordre défini d'opérations à exécuter pour la fabrication de chaque type de produit. Ils sont adaptés à la fabrication de produits diversifiés pour lesquels une demande faible s'exprime. Ils se caractérisent ainsi par une variabilité dans les opérations et un mix de production variable dans le temps. Il est donc nécessaire que le système soit flexible (susceptible de répondre au mieux aux variations de la demande). Comme précédemment, l'objectif le plus considéré est la minimisation du temps total de production.

La Figure 10 montre un exemple d'un atelier à cheminements multiples avec quatre travaux et six machines.

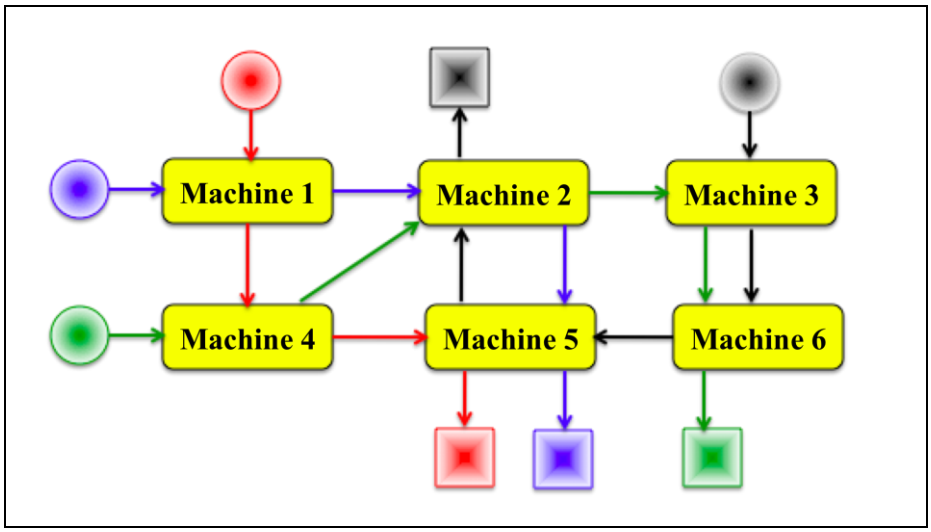


Figure 10. Atelier à cheminements multiples (Jop Shop)

Ce type d'ateliers se distingue des **ateliers à cheminement libre (open shop)** où chaque produit à traiter doit subir un ensemble d'opérations sur un ensemble de machines, mais dans un ordre totalement libre.

3.2 Ordonnancement de n tâches nécessitant un seul centre de production

Nous sommes dans le cas de la Figure 7. Considérons l'exemple consigné dans le Tableau 2 où les temps opératoires p_j associés aux jobs (tâches) j sont exprimés en centièmes d'heure. Il est clair que quel que soit l'ordre des tâches retenu, le temps opératoire total reste le même à savoir $\sum_j p_j$.

Tout ordonnancement associe à chacune des tâches un numéro d'ordre. Il s'agit donc d'une application O qui, à la position h dans la séquence fait correspondre une tâche unique $j = \mathrm{O}(h)$. On présente dans le tableau 1 un ordonnancement possible (ligne "Job programmé"). On a ainsi $\mathrm{O}(1) = 3,\ \mathrm{O}(2) = 4$ etc. L'application inverse $\mathrm{O}^{-1}(j) = h$ associe naturellement à toute tâche j sa position h (ou numéro d'ordre) dans la séquence, avec $h = 1,\mathrm{K}\ ,n$. L'avant dernière ligne indique pour chaque tâche sa date de fin C_j effective. A la dernière ligne, nous donnons la date effective de début de la tâche. Dans la suite, nous ne ferons pas de distinction entre la position h dans la séquence et la tâche j affectée à cette position.

Job j	1	2	3	4	5
Temps opératoire p_j	50	150	80	200	30
Job programmé	3	4	1	5	2
Date de fin effective C_j	80	280	330	360	510
Date de début effective t_j	0	80	280	330	360

Tableau 2. Un exemple d'ordonnancement

Notons que le temps d'attente d'une tâche située en position h dans la séquence est donné par t_h, sa date de début effective. Si les tâches se succèdent sans encombre, comme c'est le cas ici, alors le temps d'attente d'une tâche est égal à la somme des temps opératoires de toutes les tâches la précédant dans la séquence. Ainsi $t_h = \sum_{k=1}^{h-1} p_k$, où p_k désigne le temps opératoire de la tâche située en position k dans la séquence. Le temps total d'attente est égal à $\sum_{h=1}^{n} t_h = \sum_{h=1}^{n}\sum_{k=1}^{h-1} p_k$, ce qui en développant donne $\sum_{h=1}^{n} t_h = np_1 + (n-1)p_2 + \mathrm{L}\ + p_n$. Le temps moyen d'attente par tâche, noté $\overline{A}$, est simplement la moyenne arithmétique des temps d'attente, à savoir $\overline{A} = \frac{1}{n}\sum_{h=1}^{n} t_h = p_1 + \frac{n-1}{n} p_2 + \mathrm{L}\ + \frac{1}{n} p_n.$

Dans l'exemple du Tableau 2, on obtient $\overline{A} = 1050/5 = 210.$

3.2.1. Minimisation du temps moyen d'attente (règle T.O.M.)

Puisque le critère du temps opératoire total ne permet pas de discriminer parmi les $n!$ ordonnancements possibles, on va considérer un autre critère, celui du temps moyen d'attente.

D'après la définition de ce temps, il est évident pour le minimiser qu'il faut ordonner les tâches par ordre croissant de temps d'exécution puisque le plus gros coefficient (à savoir 1) est affecté au temps d'exécution de la première tâche, et le plus faible (à savoir $1/n$) à la dernière tâche. Cette règle est connue sous le nom de Shortest Processing Time rule (SPT rule) ou règle T.O.M. (Temps Opératoire Moyen).

Dans notre exemple, l'ordonnancement qui permet de minimiser ce temps moyen d'attente est défini par la séquence de jobs $\{5,1,3,2,4\}$.

Un autre critère pouvant être retenu est celui du retard qui se définit comme la différence entre la date de fin effective C_j et la date de fin souhaitée d_j. Nous prenons ici comme date de fin souhaitée "théorique" pour chaque tâche, sa date de fin effective si cette tâche avait été programmée en premier. Dans ce cas, on a donc $d_j = p_j \ \forall j = 1,\mathsf{K}\ ,n$. Avec cette définition de la date de fin souhaitée, on note que $C_j - d_j > 0 \ \forall j = 1,\mathsf{K}\ ,n$ de sorte que retard algébrique (lateness) et retard absolu (tardiness) correspondent toujours. Le Tableau 3 indique le retard de chaque tâche pour l'ordonnancement retenu dans le Tableau 1.

Position h	1	2	3	4	5	
Job programmé $j = \mathrm{O}(h)$	3	4	1	5	2	
Date de fin effective C_j	80	280	330	360	510	$\sum = 1560$
Date de fin souhaitée $d_j = p_j$	80	200	50	30	150	$\sum = 510$
Retard $L_j = C_j - d_j$	0	80	280	330	360	$\sum = 1050 = 1560 - 510$

Tableau 3. Retard algébrique.

On cherche donc à minimiser $L = \sum_{j=1}^{n}\left(C_j - d_j\right) = \sum_{j=1}^{n} C_j - \sum_{j=1}^{n} p_j$. Comme la somme des temps opératoires ne dépend pas de l'ordonnancement, cela revient à minimiser la somme des dates de fin effectives : $\sum_{j=1}^{n} C_j$. En rappelant que p_h désigne le temps opératoire du job placé en position h dans la séquence, alors $C_h = \sum_{k=1}^{h} p_h$ et l'on cherche à minimiser $\sum_{h=1}^{n} C_h = \sum_{h=1}^{n}\sum_{k=1}^{h} p_h$, ce qui en développant donne $np_1 + (n-1)p_2 + \mathsf{L} + p_n$. Ainsi, au temps opératoire de la première tâche est affectée le plus gros coefficient multiplicateur, à savoir n et à la dernière tâche, le plus faible. A nouveau, comme pour le temps moyen d'attente, il est évident qu'il faut ordonner les tâches par ordre croissant de leur temps opératoire pour minimiser le retard.

Une application à notre exemple de la règle T.O.M. qui revient donc à minimiser le retard algébrique "théorique" est donnée dans le Tableau 4. Le retard total est égal à 1090 et le retard moyen vaut 1090/5=218.

Job j	1	2	3	4	5
Temps opératoire p_j	50	150	80	200	30
Job programmé	5	1	3	2	4

$j = \mathrm{O}(h)$						
Date de fin effective C_j	30	80	160	310	510	$\sum = 1090$

Tableau 4. Application de la règle T.O.M.

3.2.2. La règle de la date de fin minimale (Early Due Date, ou E.D.D.).

Cette règle consiste simplement à séquencer les tâches par ordre croissant de leur date de fin souhaitée. On montre que cet ordonnancement minimise le retard algébrique vrai (lateness) $\bar{L} = \frac{1}{n}\sum_{h=1}^{n}(C_h - d_h)$, où les d_h correspondent aux "vraies" dates de fin souhaitée et non aux temps opératoires. On introduit dans le Tableau 5 ce type de date et l'on donne l'ordonnancement correspondant à ce critère E.D.D.

Job j	1	2	3	4	5
Temps opératoire p_j	50	150	80	200	30
Date de fin souhaitée d_j	100	300	410	400	200
Job programmé $j = \mathrm{O}(h)$	1	5	2	4	3

Tableau 5. Application de la règle E.D.D.

3.2.3. La règle First Come First Serve (F.C.F.S) et comparaison des Trois règles

La règle F.C.F.S conduit à exécuter les tâches dans leur ordre d'arrivée. On supposera dans notre exemple que l'ordre d'arrivée des jobs correspond à celui du Tableau 2, à savoir $\{3,4,1,5,2\}$. Nous allons comparer la performance de ces 3 règles à l'aide des critères suivants:

- Le temps moyen d'attente $\bar{A} = \frac{1}{n}\sum_{h=1}^{n} t_h$
- Le retard algébrique moyen "vrai" (lateness) $\bar{L} = \frac{1}{n}\sum_{h=1}^{n}(C_h - d_h)$, où , rappelons-le, les d_h correspondent aux "vraies" dates de fin souhaitée et non aux temps opératoires.
- Le retard absolu (tardiness) moyen $\bar{T} = \frac{1}{n}\sum_{h=1}^{n}\max(0, C_h - d_h)$;
- L'avance (earliness) moyenne $\bar{E} = \frac{1}{n}\sum_{h=1}^{n}\max(0, d_h - C_h)$;

La Figure 11 donne les ordonnancements selon les 3 règles ainsi que la valeur des critères précédents pour chacun d'eux. On constate que la règle F.C.F.S conduit à une avance moyenne maximum dans notre cas, mais ce résultat est fortuit et l'avantage d'une telle règle ne se mesure pas à l'aide des critères retenus ici : cette règle est souvent employée pour sa simplicité. La règle T.O.M. permet de minimiser le temps moyen d'attente. De ce point de vue elle constitue la meilleure règle. Mais elle ne minimise pas le retard algébrique vrai, ce

que fait la règle E.D.D. On note que l'avance moyenne est plus faible avec E.D.D qu'avec T.O.M. ce qui a pour conséquence un temps de stockage des tâches achevées plus faible.

Job	1	2	3	4	5		
Temps opératoire	50	150	80	200	30		
Date fin souhaitée	100	300	410	400	200		
Ordo. F.C.F.S.	**3**	**4**	**1**	**5**	**2**		
Temps opératoire p_h	80	200	50	30	150		
Date fin souhaitée d_h	410	400	100	200	300	Moyenne	
Date début effectif t_h	0	80	280	330	360	**210**	(Temps moyen d'attente)
Date fin effective C_h	80	280	330	360	510	**312**	(Temps d'achèvement moyen)
Retard algébrique, C_h-d_h	-330	-120	230	160	210	**30**	(Retard algébrique vrai moyen)
Retard absolu, max(0,C_h-d_h)	0	0	230	160	210	**120**	(Retard absolu moyen)
Avance, max (0,d_h-C_h)	330	120	0	0	0	**90**	(Avance moyenne)
Job	1	2	3	4	5		
Temps opératoire	50	150	80	200	30		
Date fin souhaitée	100	300	410	400	200		
Ordo. T.O.M.	5	1	3	2	4		
Temps opératoire p_h	30	50	80	150	200		
Date fin souhaitée d_h	200	100	410	300	400	Moyenne	
Date début effectif t_h	0	30	80	160	310	**116**	(Temps moyen d'attente)
Date fin effective C_h	30	80	160	310	510	**218**	(Temps d'achèvement moyen)
Retard algébrique, C_h-d_h	-170	-20	-250	10	110	**-64**	(Retard algébrique vrai moyen)
Retard absolu, max(0,C_h-d_h)	0	0	0	10	110	**24**	(Retard absolu moyen)
Avance, max (0,d_h-C_h)	170	20	250	0	0	**88**	(Avance moyenne)
Job	1	2	3	4	5		
Temps opératoire	50	150	80	200	30		
Date fin souhaitée	100	300	410	400	200		
Ordo. E.D.D.	1	5	2	4	3		
Temps opératoire p_h	50	30	150	200	80		
Date fin souhaitée d_h	100	200	300	400	410	Moyenne	
Date début effectif t_h	0	50	80	230	430	**158**	(Temps moyen d'attente)
Date fin effective C_h	50	80	230	430	510	**260**	(Temps d'achèvement moyen)
Retard algébrique, C_h-d_h	-50	-120	-70	30	100	**-22**	(Retard algébrique vrai moyen)
Retard absolu, max(0,C_h-d_h)	0	0	0	30	100	**26**	(Retard absolu moyen)
Avance, max (0,d_h-C_h)	50	120	70	0	0	**48**	(Avance moyenne)

Figure 11. Comparaison des règles T.O.M., E.D.D. et F.C.F.S.

3.3 Ordonnancement de n tâches sur deux centres de production

Chaque tâche nécessite pour son exécution le passage sur deux machines A et B . On note p_{jA} et p_{jB} les temps opératoires de la tâche j sur les machines A et B respectivement. Le critère d'ordonnancement retenu est celui de la minimisation du temps total d'exécution de toutes les tâches sur les deux machines. On distingue deux cas :

- Toutes les tâches sont exécutées d'abord sur A puis sur B (elles ont toutes le même ordre de passage sur les deux machines). Il s'agit d'un cas particulier de la Figure 9 où $m = 2$ (cheminement unique ou Flow Shop).
- Les tâches n'ont pas toutes le même ordre de passage sur les deux machines. Ce cas s'inscrit dans celui de la Figure 10, avec $m = 2$ (cheminement multiple ou Job Shop).

3.3.1 Le cas de tâches s'effectuant toutes selon le même ordre sur chaque machine (algorithme de Johnson) - Flow Shop

Le raisonnement à suivre est illustré à l'aide des données de temps opératoire consignées dans le Tableau 6.

Job j	1	2	3	4	5
Temps opératoire p_{jA}	50	150	80	200	30
Temps opératoire p_{jB}	60	50	150	70	200

Tableau 6. Exemple pour deux machines - Flow Shop

L'algorithme de Johnson (1954) permet de déterminer l'ordonnancement qui minimise le temps opératoire total.

- **Etape 1.** Rechercher la tâche j dont le temps opératoire p_{jm} est le plus faible, quelle que soit la machine m
- **Etape 2.** Si $m = A$, placer la tâche j à la première place disponible dans la séquence d'ordonnancement. Sinon, placer cette tâche à la dernière place disponible
- **Etape 3.** Supprimer la tâche j de la liste des tâches restant à ordonnancer. S'il reste plus d'une tâche dans la liste, retourner à l'étape 1. Sinon, placer la dernière tâche à la seule position disponible restante dans la séquence d'ordonnancement.

En appliquant l'algorithme à notre exemple, la tâche 5 vient en première position. La tâche 2 est affectée à la dernière position. La tâche 1 est placée en deuxième. La tâche 4 est placée en quatrième position. Il ne reste plus que la troisième place pour la tâche 3. L'ordonnancement optimal est donc $\{5,1,3,4,2\}$.

Le temps opératoire total est donné par la date de fin de la dernière tâche sur la dernière machine, à savoir C_{nm} dans le cas du flow shop à n tâches et m machines. Pour déterminer C_{nm}, on doit déterminer préalablement les dates de fin de chaque tâche j sur chaque machine k, C_{jk}. Pour que la tâche j puisse débuter sur la machine k, il faut à la fois que cette tâche soit terminée sur la machine précédente $k-1$ et que la machine k soit disponible, ce qui est le cas lorsque la machine a terminé d'exécuter ses opérations sur la tâche précédente $j-1$. Donc la tâche j ne peut débuter sur la machine k qu'en date $\max(C_{j-1,k};C_{j,k-1})$ où $C_{j-1,k}$ est la date de disponibilité de la machine k et $C_{j,k-1}$ est la date de fin de la tâche j sur la machine $k-1$. Sur la première machine, seule la date d'achèvement de la tâche précédente donne la date de début de la tâche suivante. Et pour la première tâche, seule sa date d'achèvement sur la machine précédente compte. On a finalement

$$C_{j,k} = \max(C_{j-1,k};C_{j,k-1}) + p_{j,k} \quad \text{pour } k = 2,\ldots,m \text{ et } j = 2,\ldots,n$$
$$C_{j,k} = C_{j-1,k} + p_{j,k} \quad \text{pour } k = 1 \text{ et } j = 2,\ldots,n$$
$$C_{j,k} = C_{j,k-1} + p_{j,k} \quad \text{pour } k = 2,\ldots,m \text{ et } j = 1$$
$$C_{1,1} = p_{1,1}$$

3.3.2. Le diagramme de Gantt

Le calcul du temps opératoire total est facilité par une représentation de l'ordonnancement par un diagramme de Gantt où figure

- en abscisse : le temps ; chaque tâche étant représentée par sa date de début et sa date de fin ;
- en ordonnée : les moyens de production utilisés (chaque centre de production).

A la Figure 12 on trouve le diagramme de Gantt associé à l'ordonnancement optimal.

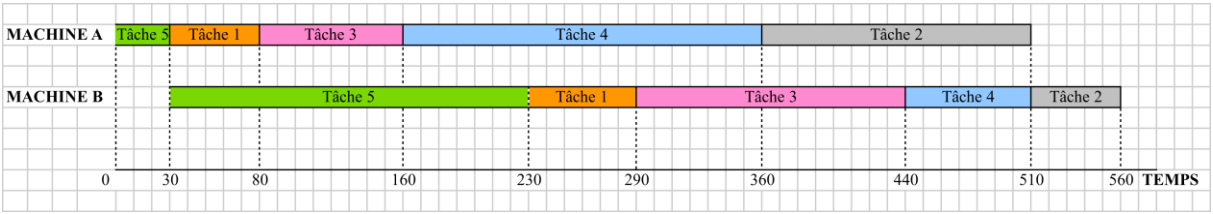


Figure 12. Diagramme de Gantt associé à l'ordonnancement optimal

Si l'on avait retenu pour ce cas l'ordonnancement selon la règle T.O.M, à savoir $\{5,1,3,2,4\}$, nous aurions obtenu le diagramme de Gantt de la Figure 13, où il existe un temps d'inoccupation de la machine B (repéré par la lettre Z sur le diagramme) de 20 centième d'heure. En effet, la tâche 4 ne se termine qu'en 510 sur la machine A alors que la machine B est disponible dès la date 490.

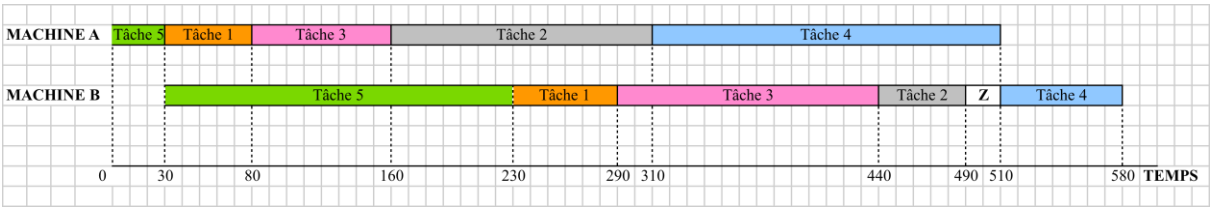


Figure 13. Diagramme de Gantt associé à l'ordonnancement T.O.M.

3.3.3 Le cas de tâches ne s'effectuant pas dans le même ordre (algorithme de Jackson) - Job Shop

Dans le cas le plus général, certaines tâches ne nécessitent leur passage que sur l'une des deux machines, d'autres sur les deux machines, dans un ordre ou dans un autre. L'ordonnancement qui minimise le temps opératoire total s'obtient par l'algorithme de Jackson qui résulte d'une adaptation de l'algorithme de Johnson. Nous allons présenter cet algorithme en l'illustrant simultanément à l'aide de l'exemple consigné dans le Tableau 7.

	Tâches sur A puis sur B						Tâches sur B puis sur A				
Job j	1	2	3	4	5	6	7	8	9	1	1

										0	1
Temps opératoire p_{jA}	50	80	10	50	30	70	90	20	10	40	10
Temps opératoire p_{jB}	30	60	30	0	0	0	70	30	100	0	0

Tableau 7. Illustration de l'algorithme de Jackson

Etape 1. Faire une partition de l'ensemble des tâches en :

- L'ensemble J_A des jobs ne passant que sur la machine A ; $J_A = \{4,5,6\}$
- L'ensemble J_B des jobs ne passant que sur la machine B ; $J_B = \{10,11\}$
- L'ensemble J_{AB} des jobs passant d'abord sur la machine A puis sur la machine B ; $J_{AB} = \{1,2,3\}$
- L'ensemble J_{BA} des jobs passant d'abord sur la machine B puis sur la machine A ; $J_{BA} = \{7,8,9\}$

Etape 2. Calculer un ordonnancement pour chaque sous-ensemble

- Déterminer l'ordonnancement optimal avec l'algorithme de Johnson pour chacun des sous-ensembles de tâches passant sur les deux machines. Pour J_{AB}, on obtient $O_{AB} = \{3,2,1\}$ et pour J_{BA}, on a $O_{BA} = \{9,8,7\}$
- Pour les tâches ne nécessitant le passage que sur l'une des deux machines, on retombe dans le cas de l'ordonnancement de n tâches sur une machine, où la séquence n'a pas d'influence sur le temps opératoire total. On pourra donc choisir la règle T.O.M par exemple. Dans notre illustration, la règle T.O.M conduit pour $J_A = \{4,5,6\}$ à la séquence $O_A = \{5,4,6\}$ et pour $J_B = \{10,11\}$ à la séquence $O_B = \{11,10\}$.

Etape 3. Combiner les sous-ensembles de séquences pour obtenir l'ordonnancement final en remarquant que l'on a intérêt à débuter le plus vite possible sur la machine A les tâches qui devront ensuite être traitées sur la machine B et à mettre en dernier sur cette machine A les tâches qui doivent d'abord aller sur B. Cela revient à combiner les résultats de la manière suivante.

- Pour la machine A, la séquence finale est $O_{AB} \cup O_A \cup O_{BA}$, ce qui donne dans l'exemple $\{3,2,1,5,4,6,9,8,7\}$
- Pour la machine B, la séquence finale est $O_{BA} \cup O_B \cup O_{AB}$, ce qui donne dans l'exemple $\{9,8,7,11,10,3,2,1\}$

On peut comme précédemment utiliser un diagramme de Gantt pour représenter l'ordonnancement optimal.

3.4 Ordonnancement de n tâches sur m machines avec ordre identique de passage - Flow Shop (heuristique CDS)

Il existe dans ce cas $(n!)^m$ ordonnancements possibles. La formulation générale de ce problème peut être résolue à l'optimum à l'aide de la programmation dynamique ou de la programmation linéaire en nombre entiers. Mais au delà d'un certain nombre de tâches et de machines, la taille du problème devient telle qu'il est impossible de trouver une solution optimale dans un temps raisonnable.

L'une des approches heuristiques existantes pour ce type de problème est proposée par Campbell, Dudek et Smith (1970), (CDS dans la suite). Cette méthode consiste à générer $m-1$ solutions en appliquant l'algorithme de Johnson sur deux machines fictives. La première regroupe les k premières machines, la deuxième regroupe les k dernières ; avec k variant de 1 à $m-1$. Les temps opératoires de chaque tâche j sur ces deux machines fictives sont la somme des temps opératoires sur les machines qu'elles regroupent. Ils se définissent donc de la manière suivante

$$P_{j1}=\sum_{h=1}^{k}p_{jh},\quad P_{j2}=\sum_{h=m-k+1}^{m}p_{jh}\quad \text{pour } j=1,\ldots,n \text{ et pour } k=1,\ldots,m-1.$$

La meilleure des $m-1$ solutions c'est-à-dire celle qui fournit le temps total d'exécution le plus faible est retenue comme la solution au problème à m machine.

Illustrons le raisonnement à l'aide d'un exemple à $m=3$ machines et $n=7$ tâches. Les données de temps opératoire sont fournies dans le Tableau 8.

	Tâche j						
Machine h	1	2	3	4	5	6	7
1	20	12	19	16	14	12	17
2	4	1	9	12	5	7	8
3	7	11	4	18	18	3	6

Tableau 8. Temps opératoires de 7 tâches sur 3 machines

On va donc générer $m-1=2$ solutions, en appliquant Johnson sur deux couples de machines fictives. Le premier couple sera constitué de la machine 1 d'une part et du groupement des machines 2 et 3 d'autre part. Le deuxième couple sera constitué du groupement des machines 1 et 2 d'une part et de la machine 3 d'autre part. Les temps opératoires à prendre en compte figurent dans le Tableau 9.

	Tâche j						
Groupement Machine	1	2	3	4	5	6	7
2 et 3	11	12	13	30	23	10	14
1 et 2	24	13	28	28	19	19	25

Tableau 9. Temps opératoires de 7 tâches sur les groupements de machines

L'application de l'algorithme de Johnson sur les machines fictives (1) et (2,3) conduit à la séquence $\{5,4,7,3,2,1,6\}$ dont la durée totale d'exécution est de 120. Sur les machines fictives (1,2) et (3), on obtient $\{4,5,2,1,7,3,6\}$ ou $\{5,4,2,1,7,3,6\}$ dont les durées d'exécution totale sont aussi de 120.

Propriétés particulières du cas de 3 machines.

On dit d'une machine M_1 qu'elle est dominée par la machine M_2, si $\max_{j=1,\ldots,n} p_{j,M_1} \leq \min_{j=1,\ldots,n} p_{j,M_2}$, c'est-à-dire si le plus grand temps opératoire sur M_1 (toutes tâches confondues) n'excède pas le plus petit temps opératoire sur M_2. Dans ce cas, il n'est pas nécessaire d'évaluer deux solutions comme précédemment. Il suffit en effet d'appliquer l'algorithme de Johnson sur les deux groupements de machines constitués par la machine dominée et les deux autres, à savoir (M_1, M_2) et (M_1, M_3).

Dans notre exemple, on obtient

Machine h	$\max_{j=1,\ldots,n} p_{j,h}$	$\min_{j=1,\ldots,n} p_{j,h}$
1	20	12
2	12	1
3	18	3

La machine 2 est donc dominée par la machine 1. On applique alors l'algorithme de Johnson sur les groupements de machine $(2,1)$ et $(2,3)$ et l'on obtient la séquence $\{5,4,7,3,2,1,6\}$.

Il a été démontré que lorsque :

$$\max_{j=1,\ldots,n} p_{j,M_1} \leq \min_{j=1,\ldots,n} p_{j,M_2} \text{ ET } \max_{j=1,\ldots,n} p_{j,M_1} \leq \min_{j=1,\ldots,n} p_{j,M_3},$$

c'est-à-dire lorsque la machine M_1 est à la fois dominée par M_2 et M_3 alors l'algorithme de Johnson appliqué sur les groupements de machines (M_1, M_2) et (M_1, M_3) fournit la solution optimale.

Dans notre exemple, la machine 2 n'est pas dominée par la machine 3, donc l'algorithme de Johnson ne garantit pas l'optimalité de la solution trouvée.

3.5 Ordonnancement de n tâches sur un seul centre de production et dont l'ordre de passage influe sur les coûts de lancement

Nous présentons ici un exemple concret de ce type de situation et en montrons l'analogie avec le problème du voyageur de commerce. Nous montrons comment formuler ce problème par un programme linéaire en nombre entiers. Nous examinons ensuite le principe de l'algorithme de Branch and Bound (encore appelé procédure de séparation évaluation) qui sert à résoudre les programmes linéaires en nombre entiers.

3.5.1 Le cas d'un atelier de peinture et analogie avec le problème du voyageur de commerce

Considérons par exemple un atelier de peinture où, pour passer d'une couleur de produit à une autre, il est nécessaire de nettoyer les pistolets, ce qui requiert une quantité de solvant et un temps de travail qui dépendent à la fois de la couleur utilisée précédemment et de la couleur courante que l'on va utiliser. En effet, passer par exemple du blanc au rouge mobilise moins de ressources que passer du rouge au blanc. Dans le Tableau 10, on consigne les coûts

(solvant et temps de travail) associés à tout couple de couleur, pour quatre couleurs différentes.

		Nouvelle couleur			
		Blanc	Jaune	Rouge	Bleu
Couleur initiale	Blanc	0	5	6	7
	Jaune	6	0	4	5
	Rouge	8	6	0	7
	Bleu	7	8	6	0

Tableau 10. Coûts associés aux changements de teinte

Imaginons que lors d'une journée de production, on doive produire des lots de produits de chacune de ces quatre couleurs. Il existe $4! = 24$ séquences possibles de couleurs que l'on a énumérées à la Figure 14.

Position 1	**Position 2**	**Position 3**	**Position 4**	**Coût**
Blanc	Jaune	Rouge	Bleu	**16**
		Bleu	Rouge	**16**
	Rouge	Jaune	Bleu	17
		Bleu	Jaune	21
	Bleu	Jaune	Rouge	19
		Rouge	Jaune	19
Jaune	Blanc	Rouge	Bleu	19
		Bleu	Rouge	19
	Rouge	Blanc	Bleu	19
		Bleu	Blanc	18
	Bleu	Blanc	Rouge	18
		Rouge	Blanc	19
Rouge	Blanc	Jaune	Bleu	18
		Bleu	Jaune	23
	Jaune	Blanc	Bleu	19
		Bleu	Blanc	18
	Bleu	Blanc	Jaune	19
		Jaune	Blanc	21
Bleu	Blanc	Rouge	Jaune	19
		Jaune	Rouge	**16**
	Jaune	Rouge	Blanc	20
		Blanc	Rouge	20
	Rouge	Blanc	Jaune	19
		Jaune	Blanc	18

Figure 14. Séquences possibles de couleur

Comme on peut le voir à la Figure 14, il existe plusieurs séquences qui minimisent le coût. La méthode qui consiste à dénombrer toutes les combinaisons possibles n'est évidemment pas la plus efficace. On peut songer d'abord à la programmation dynamique que nous avons examinée en présentant l'algorithme de Wagner-Whitin. Il existe toutefois un algorithme plus efficace pour ce type de problème présentant une analogie avec celui du voyageur de commerce qui doit visiter n villes différentes, en ne passant qu'une seule fois par chacune d'entre-elles, sachant qu'il existe un coût $c_{i,j}$ pour aller de la ville i à la ville j. Les villes à visiter s'apparentent donc aux couleurs que l'on doit séquencer, avec un coût de passage de l'une à l'autre. Il est possible de formuler le problème comme un programme linéaire en

nombre entier où la variable $x_{i,j}$ prend la valeur 1 si le voyageur se rend de la ville i à la ville j et 0 sinon. Dans notre exemple d'atelier de peinture, $x_{i,j}$ vaut 1 si l'on passe de la couleur i à la couleur j et zéro sinon. Le programme s'écrit

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n} c_{i,j} \cdot x_{i,j},$$

$$s.c. \begin{cases} \sum_{i=1}^{n} x_{i,j} = 1 \\ \sum_{j=1}^{n} x_{i,j} = 1 \end{cases}$$

La première contrainte exprime le fait que si le voyageur entre dans la ville i alors il en sort (c'est-à-dire qu'il existe une ville j où le voyageur se rend depuis la ville i). La deuxième contrainte implique que pour arriver à la ville j le voyageur venait nécessairement d'une ville i.

Le problème peut se résoudre par le Branch and Bound (encore appelé procédure par séparation - évaluation) qui s'applique aux programmes linéaires en nombres entiers et se fonde sur des relaxations du problème, résolus à l'aide du simplexe. La procédure garantit l'obtention d'une solution optimale.

3.5.2 Programmation linéaire en nombres entiers : l'algorithme de Branch-and-Bound

Lorsqu'on a affaire à un problème avec des contraintes d'intégralité (tout ou partie des variables du problèmes doivent être entières), la tentation est grande de résoudre le problème en ignorant ces contraintes d'intégralité, ce qui peut se faire à l'aide du simplexe (problème linéaire, où la fonction objectif et les contraintes sont linéaires). Puis, prendre les solutions obtenues par le simplexe et les arrondir. Mais cela ne marche pas. L'exemple suivant montre le "danger" d'une telle méthode. Considérons le problème :

$$\max z = x_1 + 5x_2$$
$$s.c.$$
$$x_1 + 10x_2 \le 20$$
$$x_1 \le 2$$
$$x_1, x_2 \in N$$

En ignorant les contraintes d'intégralité, la solution fournie par le simplexe est $x_1 = 2; x_2 = 1.8$ et donne $z = 11$. La première idée serait d'arrondir x_2 à l'entier immédiatement supérieur, soit $x_2 = 2$. Mais cette solution est infaisable car elle ne respecte pas la contrainte $x_1 + 10x_2 \le 20$. En arrondissant alors x_2 à l'entier immédiatement inférieur, soit $x_2 = 1$, les contraintes sont respectées mais $z = 7$ n'est pas l'optimum. En effet, la solution optimale de ce problème est $x_1 = 0; x_2 = 2$, ce qui donne $z = 10$.

Une autre solution à laquelle on peut penser consiste en l'énumération de toutes les solutions possibles et de choisir la meilleure. Cela peut marcher pour les problèmes de petites tailles, mais ce n'est pas pensable pour les problèmes de moyenne et grande taille. Par exemple, un problème contenant 20 variables binaires donne $2^{20} = 1048576$ solutions à énumérer.

L'explosion combinatoire est encore plus importante lorsqu'on considère des variables entières prenant leur valeur dans un intervalle plus large.

La méthode de Branch and Bound constitue la technique appropriée pour résoudre ce type de problème. Elle se fonde sur l'observation selon laquelle l'énumération de solutions entières prend la forme d'un arbre. Considérons par exemple l'énumération complète des solutions d'un problème comportant une variable entière "générale" $x_1 \leq 3$ et deux variables binaires $0 \leq x_2 \leq 1$ et $0 \leq x_3 \leq 1$. La Figure 15 montre l'énumération complète de toutes les solutions.

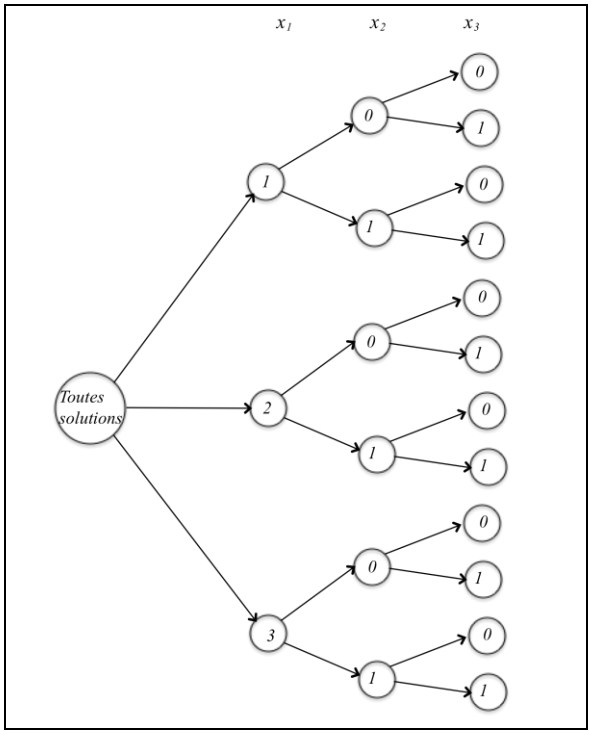


Figure 15. Arbre d'énumération

Le noeud racine représente toutes les solutions au problème. Le noeud $x_1 = 2$ représente toutes les solutions avec $x_1 = 2$. Ce noeud est le noeud parent du noeud enfant $x_2 = 0$ qui contient quant à lui toutes les solutions où on a la fois $x_1 = 2$ et $x_2 = 0$, etc. Les noeuds feuilles de l'arbre sont ceux pour lesquels x_1, x_2 et x_3 ont toutes une valeur entière. Il en existe 12 qui représentent toutes les combinaisons possibles de solutions entières au problème.

L'idée de base du Branch and Bound est d'éviter autant que possible que l'arbre grandisse, en explorant uniquement les noeuds qui fournissent les solutions les plus prometteuses. Le caractère prometteur de ces solutions est évalué en estimant une borne sur la valeur de la fonction objectif qui pourra être obtenue par l'exploration de noeuds à des itérations ultérieures. La séparation (branching) se produit lorsqu'un noeud est sélectionné pour une exploration des solutions issues de ce noeud. L'évaluation (bounding) se produit lorsque la meilleure valeur d'un noeud est estimée. On utilise la terminologie suivante.

- Noeud : toute solution, partielle ou complète
- Noeud-feuille, ou feuille : solution complète (les valeurs de toutes les variables sont connues)
- Noeud - bourgeon (bud node) : toute solution partielle, réalisable ou non.

- Méthode de détermination d'une borne : elle permet d'estimer la meilleure valeur de la fonction objectif qu'il est possible d'obtenir en laissant grandir un noeud. Cette méthode doit toujours produire une estimation optimiste.
- Méthode de séparation : processus par lequel on crée un noeud enfant à partir d'un noeud bourgeon.
- Incumbent : meilleure solution entière trouvée jusque là.

C'est la méthode de détermination d'une borne qui est le vecteur d'efficacité de l'algorithme de branch and bound. Reprenons par exemple le problème du voyageur de commerce. Dans ce cas, un noeud représente un tour partiel et la méthode de détermination d'une borne peut consister à estimer la longueur du tour le plus court qu'il est possible d'effectuer à partir du tour partiel. Considérons le tour partiel apparaissant en rouge à la Figure 16.

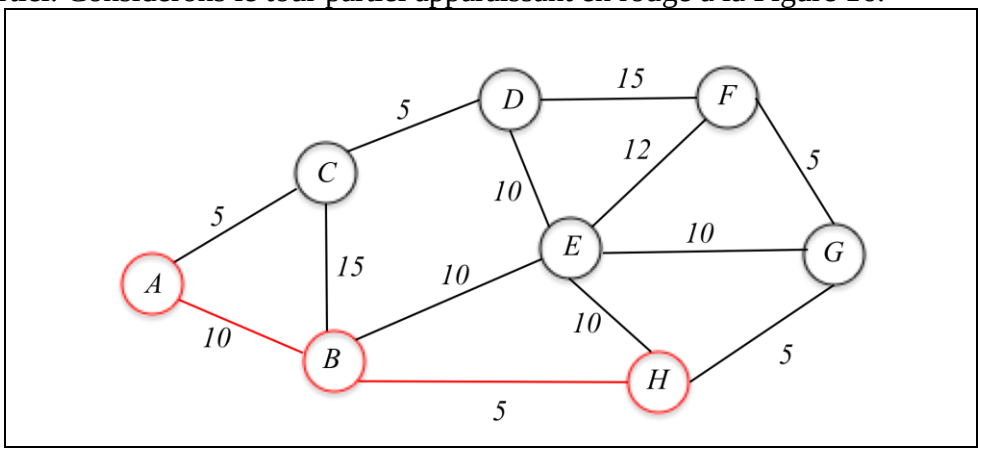


Figure 16. Tour partiel dans le problème du voyageur de commerce

Une méthode de détermination de la borne peut consister à rechercher un arbre de recouvrement minimum qui est un graphe joignant tous les noeuds et de longueur totale minimum. Dans notre exemple, l'arbre de recouvrement minimum apparaît en pointillés et en bleu sur la Figure 17.

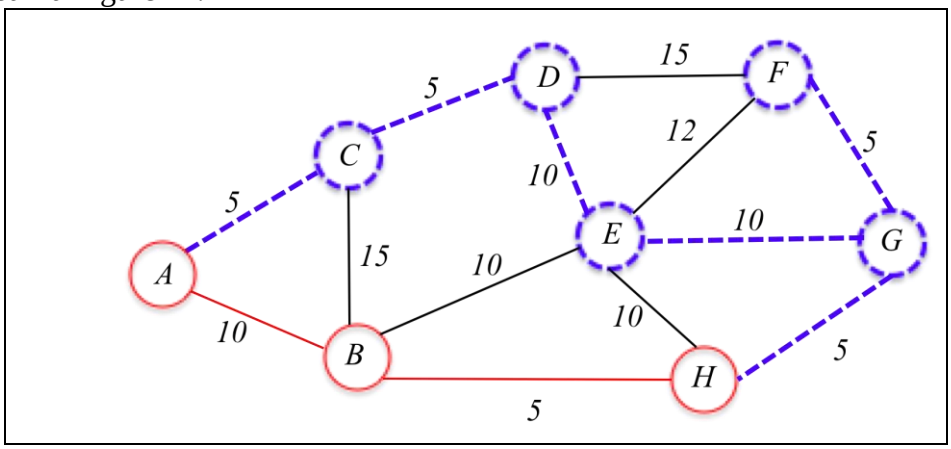


Figure 17. Arbre de recouvrement minimum

La borne sur la valeur de la fonction objectif est donnée par la somme de la longueur du tour partiel et de la longueur de l'arbre de recouvrement minimum, à savoir (10+5)+(5+5+10+10+5+5)=55. La solution ainsi générée n'est pas réalisable car certaines villes sont visitées plusieurs fois. Mais il s'agit d'une bonne sous-estimation de la longueur totale du tour.

Un autre algorithme plus efficace pour ce problème existe ; on le doit à Little, Murty, Sweeney et Karel. Sa présentation est faite dans l'ouvrage de Vincent Giard, pp. 390-401.