

CHAPITRE I

**FONDEMENTS THEORIQUE SUR L'ORDONNANCEMENT
ET LES PROBLEME D'ORDONNANCEMENT****1. Introduction :**

L'ordonnancement apparaît dans tous les domaines de l'économie, l'informatique, la construction, l'industrie et l'administration. Il consiste à organiser dans le temps la réalisation des tâches, compte tenu des contraintes temporelles (délais, contraintes d'enchaînement,) et des contraintes portant sur l'utilisation et la disponibilité des ressources requises.

2. Définition d'ordonnancement :

Un ordonnancement constitue une solution au problème d'ordonnancement. Il est la programmation dans le temps l'exécution d'une série de tâches (ou activités, opérations) sur un ensemble de ressources physiques (humaines et techniques), en cherchant à optimiser certains critères, financiers ou technologiques, et en respectant le plus possible les contraintes que nous allons préciser. De manière plus précise, on parle d'ordonnancement lorsqu'on fixe les dates de début ou de fin de chacune des tâches,

Les différentes données d'un ordonnancement sont les tâches, les ressources et les contraintes.

3. Les tâches

Une tâche i , appartenant à un ensemble T fini de tâches à ordonnancer, est une entité élémentaire de travail caractérisée par une date de début au plus tôt r_i , une date de fin au plus tard d_i et une durée opératoire p_i . Sa localisation dans le temps est définie par une date de début $s_i \geq r_i$ et/ou une date de fin $c_i \leq d_i$. Dans le cas où la durée opératoire $p_i = c_i - s_i$ est connue, la valeur de la variable c_i peut être déduite de la valeur de s_i (et vice-versa).

La réalisation de la tâche i nécessite l'utilisation d'une ou plusieurs ressources $k \in R$, où R est l'ensemble de ressources. Les ressources concernées par la réalisation de la tâche sont généralement supposées connues a priori. Parfois, les ressources sont à choisir dans des ensembles de ressources (ou pools de ressources), chaque ensemble correspondant à une compétence particulière. L'intensité de ressource k consommée pour la réalisation de i est

notée q_{ik} (supposée constante lors de l'exécution de la tâche). Si les tâches peuvent être exécutées par morceaux, alors le problème est dit *préemptif*. Dans ce cas, la durée p_i d'une tâche ne peut être déterminée qu'une fois l'ordonnancement construit [LA05].

Dans le cas de l'ordonnancement d'atelier, notons que le terme opération remplace parfois celui de tâche. Dans le domaine de la gestion de projet, c'est le terme activité qui est préféré.

Une tâche est une entité élémentaire de travail (opération ou ensemble d'opérations).

4. Les caractéristiques d'une tâche

On peut regrouper les caractéristiques d'une tâche j en trois types:

- **Caractéristiques d'époques**

Une tâche doit avoir des limites chronologiques bien définies.

- s_i : date de début d'exécution de la tâche i (*starting time*).
- c_i : date de fin d'exécution de la tâche i (*completion time*).
- r_i : date au plus tôt de la tâche i (après laquelle i peut commencer) (*release date*).
- d_i : date au plus tard de la tâche i (avant laquelle i doit être achevée) (*deadline*).

- **Caractéristique de durée**

Chaque tâche possède une durée d'exécution notée p_i (processing time) telle que :

- $p_i = c_i - s_i$

- **Caractéristique d'urgence d'exécution**

- W_i : poids (*weight*).

- **Caractéristique de moyens**

Il s'agit de divers moyens (matériels, personnel, fournitures, monnaie,...) qui sont nécessaires à la réalisation des tâches.

Pour simplifier, on supposera que l'intensité est constante durant l'exécution d'une tâche.

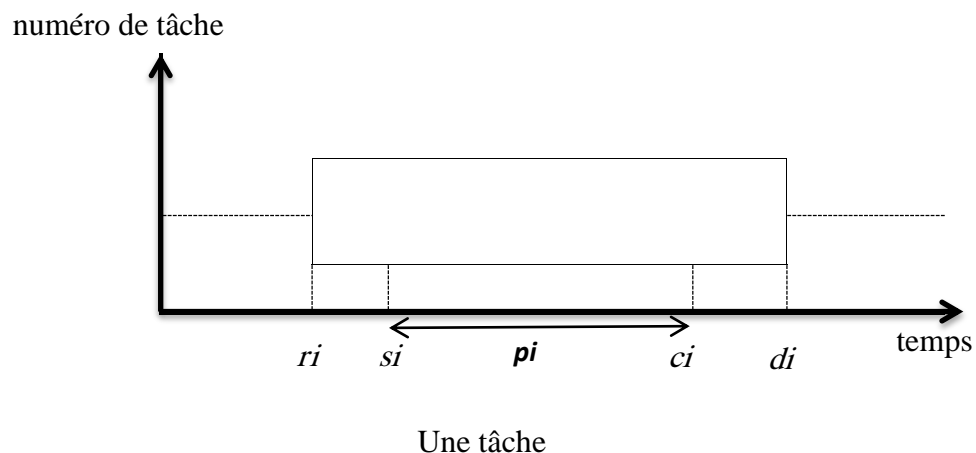


Figure 1 : Représentation une tâche

5. Type de tâches

Les tâches, peuvent être:

- **Indépendantes**

Lorsqu'elles ne sont pas liées entre elles par des contraintes de cohérence technologique (par exemple contraintes de précédences liées aux procédés de réalisation).

- **Successives**

Lorsqu'elles se déroulent les une après les autres. La tâche suivante ne peut démarrer que si la tâche précédente est terminée.

- **Simultanées**

Lorsqu'elles commencent en même temps.

- **Convergentes**

Lorsqu'elles précèdent la même tâche

6. Les ressources

Une ressource k est un moyen technique ou humain, disponible en quantité limitée, destiné à être utilisé pour la réalisation de plusieurs tâches. La disponibilité est généralement exprimée par une capacité propre à chaque ressource k , notée Q_k ($Q_k \geq 1$). Une ressource est dite renouvelable si, après avoir été utilisée par une ou plusieurs tâches, elle est à nouveau disponible en même quantité (les hommes, les machines ...). Dans le cas contraire, elle est dite consommable (matières premières, budget ...). Dans certains problèmes d'ordonnancement, la notion d'état de ressource est également utilisée afin de prendre en compte des contraintes technologiques liées à son utilisation. Dans ce cas, la capacité de la ressource dépend de l'état considéré. Un changement d'état de la ressource peut être produit par l'occurrence d'événements externes incontrôlables (panne, maintenance, . . .) ou par le début ou la fin de certaines tâches du problème[LA05].

Remarquons que dans le cas de l'ordonnancement d'atelier, le terme machine est généralement utilisé à la place de ressource.

La disponibilité d'une ressource peut à priori varier au cours du temps, sa courbe de disponibilité est en général connue à l'avance, sauf dans les cas où elle dépend du placement de certaines tâches génératrices.

On distingue deux types de ressources :

- **Les ressources consommables**

Une ressource est *consommable* si, après avoir été allouée à une tâche, elle n'est plus disponible pour les tâches suivantes. (Matières premières, l'énergie budget,...).

- **Les ressources renouvelables**

Une ressource est renouvelable si, après avoir été utilisée par une ou plusieurs tâches, elle est à nouveau disponible en même quantité, après la fin de cette ou ces tâches, pour les tâches suivantes, (les hommes, les machines, les processeurs, les fichiers,...).

On distingue principalement deux types de ressources renouvelables :

- **Les ressources disjonctives** (non partageables) qui ne peuvent exécuter qu'une tâche à la fois (machine, outil, robot manipulateur, etc.....).

- **Les ressources cumulatives** (partageables) qui peuvent être utilisées par plusieurs tâches simultanément (équipe d'ouvriers, post de travail,...).

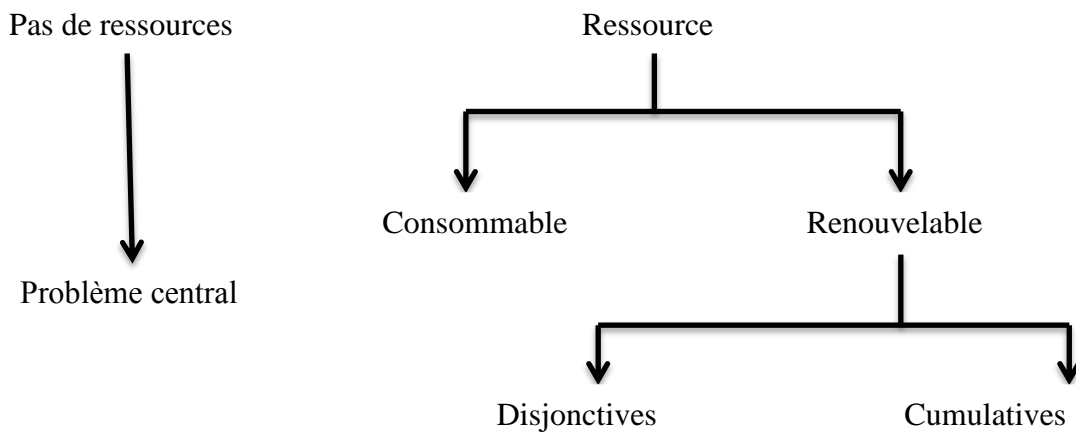


Figure 2 : Les types des ressources

7. Les Contraintes

La notion générique de contrainte est relative à un ensemble de variables de décision. Elle exprime une restriction sur les domaines de valeur de ces variables. Il y a deux sortes de variables en ordonnancement. Certaines variables concernent les décisions de localisation des tâches dans le temps (variables d'ordonnancement si et ci), d'autres concernent les décisions d'affectation des tâches sur les ressources (variables d'affectation) [LA05].

On a plusieurs types de contraintes auxquelles sont soumises les diverses tâches. On distingue:

7.1. Les contraintes potentielles

On a deux sortes

- **Contraintes de précédence** selon laquelle une tâche ne peut commencer avant qu'une autre tâche ne soit terminée (exemple : la pose de tuiles est postérieure à celle de la charpente).

- **Contraintes de localisation temporelle** impliquant qu'une tâche donnée ne peut commencer avant une date imposée, ou quelle doit être terminée avant une date imposée.

7.2. Les contraintes de ressources

- Contraintes d'utilisation de ressources qui expriment la nature et la quantité des moyens utilisés par les tâches ainsi que les caractéristiques d'utilisation de ces moyens.
- Contraintes de disponibilité des ressources qui précisent la nature et la quantité des moyens disponibles au cours du temps.

Lorsque dans un problème, certaines tâches nécessitent l'intervention simultanée de plusieurs types de ressources, on parle de contraintes multi ressources.

7.3. Les contraintes disjonctives

Deux tâches liées sont soumises à des contraintes disjonctives, si elles doivent être exécutées sur des intervalles de temps disjoints ,c.a.d qu'elles ne peuvent s'exécuter simultanément sur la même machine.

7.4. Les contraintes cumulatives

Interdisent la réalisation simultanée d'un nombre trop important de tâche compte tenu de la disponibilité maximale de la ressource à chaque instant et des quantités requises individuellement par les tâches. Ceci apparaît lorsque par exemple quatre machines sont disponibles pour la réalisation de cinq tâches simultanément, donc il faut savoir à l'avance laquelle des tâches sera retardée

Exemple

La réalisation d'un projet nécessite souvent une succession de tâches auxquelles s'attachent certaines contraintes de:

- **Temps** : délais à respecter pour l'exécution des tâches ;
- **Précédence**: certaines tâches doivent s'exécuter avant d'autres ;
- **Production** : temps d'occupation du matériel ou des hommes qui l'utilisent.

8. Une séquence (d'exécution des tâches sur une machine): est l'ordre selon laquelle les tâches sont exécutées sur cette machine. Elle ne contient aucune information explicite concernant les instants de début ou de fin d'exécution des différentes opérations.

9. Un ordonnancement (d'exécution des tâches sur une machine): est une séquence accompagnée des instants de début ou de fin d'exécution des différentes opérations

10. Dresser un calendrier d'exécution (Time tabling) : c'est établir un ordonnancement à partir d'une séquence.

11. Caractéristiques générales d'un ordonnancement

- **Ordonnancement admissible**: est un ordonnancement qui respecte toutes les contraintes du problème (une solution possible).

- **Ordonnancement actif:** est un ordonnancement dans lequel aucune tâche ne peut être commencée plus tôt sans reporter le début d'une autre.
- **Ordonnancement semi_actif:** est un ordonnancement dans lequel aucune opération ne pouvait commencer plus tôt sans altérer la séquence.
- **Ordonnancement sans retard :** dans un ordonnancement sans retard, on ne doit pas retarder l'exécution d'une tâche si celle-ci est en attente et si la ressource est disponible.

12. Critère

L'approche par optimisation suppose que les solutions candidates à un problème puissent être ordonnées de manière rationnelle selon un ou plusieurs critères d'évaluation numériques permettant d'apprécier la qualité des solutions.

- **Critère d'optimisation :** est la fonction qui attribue une valeur (coût) à un ordonnancement admissible.
- **Valeur optimal d'ordonnancement :** est le minimum (ou dans certains cas le maximum) des valeurs de tous les ordonnancements admissibles.
- **Un ordonnancement optimal :** est un ordonnancement admissible de valeur égale à la valeur optimale d'ordonnancement.
- **Problème d'ordonnancement**

Beaucoup de problème d'ordonnancement ont la forme suivante: étant donnée une collection de tâches à ordonnancer sur un système

particulier de traitement et sous des contraintes variées, trouver un ordonnancement optimal.

Parmi les critères, on a ceux qui sont liées :

➤ Au temps

- . Le temps total d'exécution (makespn) : $C_{max} = \text{Max} (c_i)$;
- . Le temps moyen d'achèvement d'un ensemble de tâches (flowtime) : $\sum C_i$.
- . Le temps moyen d'achèvement pondéré d'un ensemble de tâches (weighted flowtime) : $\sum W_i C_i$.
- . Les retards

.Plus grand retard (maximum tardiness) $\text{Max } T_i$ où

$$T_i = \begin{cases} 0 & \text{si } C_i \leq d_i \\ C_i - d_i & \text{sinon} \end{cases}$$

.Somme pondéré pondérée des retards : $\sum W_i T_i$;

.Nombre de tâches en retard : $\sum U_i$ où

$$U_i = \begin{cases} 0 & \text{si } C_i \leq d_i \\ 1 & \text{sinon} \end{cases}$$

.Nombre pondéré de tâches en retard : $\sum W_i U_i$

➤ **Aux ressources**

.La qualité - maximale, moyenne ou pondérée - de ressources nécessaires pour réaliser un ensemble de tâches.

.La charge de chaque ressource.

.Maximiser la productivité.

➤ **Aux coûts** de lancement, de production, de transport, de stockage, etc..., mais aussi aux revenus, aux retours d'investissement.

➤ **à une énergie** ou un débit...

13. Critère d'optimisation régulier

Habituellement, les critères d'optimisation sont exprimés en fonction de l'ensemble des dates de fin d'un ordonnancement de telle sorte que leur forme générale est toujours :

$Z(\sigma) = f(C_{\sigma(1)}, \dots, C_{\sigma(n)})$, où σ est un ordonnancement, $\sigma(i)$ est la tâche qui est affectée à la i ème position dans σ et C_i est la date de fin de la tâche i dans σ .

Soit σ' un ordonnancement, on désigne par $C_{i'}$ la date de fin de la tâche i' dans σ' .

Le critère Z est régulier si la fonction f est croissante au sens large c.a.d si et seulement si l'implication suivante est vraie, quels que soient les ordonnancements σ et σ' :

$$[C_{\sigma(1)} \leq C'_{\sigma(1)}, \dots, C_{\sigma(i)} \leq C'_{\sigma(i)}, \dots, \text{et } C_{\sigma(n)} \leq C'_{\sigma(n)}] \Rightarrow [Z_{(\sigma)} \leq Z_{(\sigma')}]$$

Exemple

Le critère $\sum W_i C_i$ est un critère d'optimisation régulier.

Dans les systèmes de production, terminer une tâche (fabrication d'un équipement) avant sa date de fin (date de livraison) induit un stockage de l'équipement. Intérêt de la production juste à temps.

Le coût **d'avance-retard** n'est pas régulier.

14. Satisfaction de contraintes

Lorsqu'en dehors des contraintes fortes qui définissent l'admissibilité des solutions, il est difficile de traduire l'ensemble des objectifs de résolution par un ou plusieurs critères numériques, on peut avoir recours à une approche par satisfaction de contraintes. L'ensemble des contraintes regroupe alors à la fois des contraintes intrinsèques du problème (cohérence technologique par exemple) et des objectifs de type seuil à ne pas dépasser (durée maximale, stocks plafond/plancher, etc.). On pourra dans ce cas se contenter d'une solution quelconque pourvu qu'elle soit admissible. La stratégie d'exploration de l'espace des solutions admissible doit exploiter intelligemment les contraintes pour élaguer cet espace de recherche. C'est le principe de la propagation de contraintes, qui désigne un ensemble de techniques de filtrage (retrait de valeurs inconsistantes).

15. Définition Problème d'ordonnancement :

Plusieurs définitions d'un problème d'ordonnancement sont données dans la littérature, Nous indiquons ici celle proposée dans (Esquirol & Lopez) [E&L99] :

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

Le mot « ordonnancement » désigne soit une solution au problème d'ordonnancement, soit par abus de langage, le processus ayant conduit à la détermination de cette solution.

Une solution d'ordonnancement décrit les dates prévues pour l'exécution des tâches et l'allocation des ressources au cours du temps. La méthode utilisée pour l'élaboration d'un ordonnancement vise souvent à satisfaire un ou plusieurs objectifs (coûts, délais, qualité), exprimés au sein d'un ou plusieurs critères de performance.

Pour une description plus détaillée de la problématique de l'ordonnancement, le lecteur est prié de se référer aux bibliographiques [COF76], [BLA96] ou [PIN95].

16. Quelques types de problèmes d'ordonnancement :

- Lorsqu' on connaît à l'avance l'ensemble des tâches à exécuter, leur caractéristiques et les ressources nécessaires à la résolution, on dit qu'on est devant un problème *statique*.

Et lorsque l'ensemble des tâches varie au cours du temps de façon indéterministe on dit qu'on a un problème *dynamique*.

- Lorsque Chaque tâche ne doit s'exécuter qu'une seule fois, on a un problème d'ordonnancement *non répétitif* (classique).

Le problème est appelé *répétitif* dans le cas contraire.

- Lorsque l'exécution de n'importe quelle tâche peut souvent être interrompue arbitrairement et reprise à ce même instant par, une machine différente ou ultérieurement sur n'importe quelle autre machine, on a un problème d'ordonnancement dit *préemptif* et il est dit *non préemptif* dans le cas contraire.

17. Représentation des problèmes d'ordonnancement sur machine

Pour représenter en abrégé les problèmes d'ordonnancement, on utilise la notation à trois champs ($\alpha / \beta / \gamma$)

- Les caractéristiques des ressources

Le premier champ $\alpha = \alpha_1 \alpha_2$ décrit les caractéristiques des ressources

- Le paramètre $\alpha_1 \in \{\emptyset, P, Q, R, F, J, O\}$ caractérise le type des machines utilisées.

- si $\alpha_l \in \{\emptyset, P, Q, R\}$ chaque tâche j est constitué d'une seule opération qui peut être exécutée sur n'importe quelle machine M_i avec V_{ij} .

$\alpha 1 = \emptyset$: ordonnancement **sur machine unique**;

$\alpha 1 = P$: ordonnancement **sur machines parallèles identiques**. Une tâche peut s'exécuter indifféremment sur l'une ou l'autre des machines.

Toutes les vitesses- V_{ij} sont égales. .

Dans ce cas on suppose que $V_{ij} = 1$ quelque soient i et j .

$\alpha 1 = Q$: ordonnancement **sur machines parallèles uniformes**.

$V_{ij} = V_{ik}$ pour toute machine MJ et quelque soient j et k .

Dans ce cas chaque machine MJ exécute toutes les tâches à la même vitesse V_i .

$\alpha 1 = R$: **machines parallèles indépendantes**;

- si $\alpha 1 \in \{F, J, O\}$ chaque tâche j est constitué d'un ensemble d'opérations.

L'atelier contient m machines distinctes et chaque tâche est un ensemble de m opérations, chacune d'elles devant s'exécuter sur une machine différente.

On distingue trois types d'atelier selon la nature des contraintes de précédence entre les opérations d'une même tâche.

$\alpha 1 = F$: **FLOW shop**; les opérations de toutes les tâches suivent le même ordre ; les tâches traversent les machines de l'atelier selon le même ordre (cheminement unique). Mais certaines tâches peuvent dépasser d'autre sur une certaines machines.

$\alpha 1 = J$: **Job shop**

Les opérations de chaque tâche traversent les machines d'atelier selon le même ordre ; qui n'est pas nécessairement le même pour les opérations des autres tâches.

$\alpha 1 = O$: **OPEN shop**

Est un modèle d'atelier moins contraint que le flow shop et le job shop, car l'ordre d'exécution des opérations n'est pas fixé ; chaque tâche est constituée d'opérations indépendantes.

- $\alpha 2 \in \{\emptyset, K\}$ indique le nombre de machines.

$\alpha 2 = \emptyset$: le nombre de machines est supposé être variable.

$\alpha 2 = K$: le nombre de machines est égale à K (un entier positif).

- **Les caractéristiques des tâches**

Le second champ $\beta = \beta 1\beta 2\beta 3\beta 4\beta 5\beta 6\beta 7$ indique un nombre de caractéristique des tâches, qui peuvent être définie comme suit:

- Le paramètre $\beta 1 \in \{\mathbf{pmtn}, \emptyset\}$ indique s'il est possible ou non d'exécuter les tâches par morceaux (division de tâches ou encore préemption).

$\beta 1 = \mathbf{pmtn}$: la préemption est permise ;

$\beta 1 = \emptyset$: la préemption n'est pas permise;

- Le paramètre $\beta 2 \in \{\emptyset, \mathbf{prec}, \mathbf{chain}, \mathbf{tree}\}$ reflète les contraintes de précédence.

$\beta_2 = \emptyset$: les tâches sont indépendantes.

$\beta_1 = \text{prec}$: des relations de précédence arbitraires entre les tâches sont données.

$\beta_2 = \text{chain}$: les contraintes de précédences forment une chaîne.

$\beta_2 = \text{tree}$: le graphe de précédence associé ayant n sommets $1, 2, \dots, n$ prend la forme d'un arbre.

- Le paramètre $\beta_3 \in \{\emptyset, p_i = p, p \leq p_i \leq \bar{p}\}$ décrit la durée d'exécution des opérations.

$\beta_3 = \emptyset$: les opérations ont des durées d'exécution arbitraires (entier non négatifs).

$\beta_3 = p_i = p$: toutes les opérations ont des durées d'exécution égale à p unités.

$\beta_3 = p \leq p_i \leq \bar{p}$: toutes les opérations ont des durées d'exécutions comprises largement entre p et \bar{p} .

- Le paramètre $\beta_4 \in \{\emptyset, r_i, d_i, r_i \text{ et } d_i\}$ montre des contraintes temporelles.

$\beta_4 = \emptyset$: les tâches sont disponibles continuellement.

$\beta_4 = d_i$: les tâches ont des dates au plus tard.

$\beta_4 = r_i$: les tâches ont des dates au plus tôt.

$\beta_4 = r_i \text{ et } d_i$: les tâches ont des dates au plus tôt et les tâches ont des dates au plus tard.

- Le paramètre $\beta_5 \in \{\emptyset, p_i \leq p_j \Rightarrow w_i \geq w_j\}$ concerne le choix des poids w_i .

$\beta_5 = \emptyset$: les w_i sont des entiers positifs arbitraires.

$\beta_5 = p_i \leq p_j \Rightarrow w_i \geq w_j$: les w_i sont agréables.

- Le paramètre $\beta_6 \in \{\emptyset, r_i \leq r_j \Rightarrow d_i \geq d_j\}$ concerne le choix des dates au plus tard d_i .

$\beta_6 = \emptyset$: les d_i sont des entiers positifs arbitraires.

$\beta_6 = r_i \leq r_j \Rightarrow d_i \geq d_j$: les d_i sont agréables.

- Le paramètre $\beta_7 \in \{\emptyset, l_j\}$ concerne les durées de transport des tâches entre les machines.

$\beta_7 = \emptyset$: aucune durée de transport.

$\beta_7 = l_j$: les tâches ont des durées-de transport entre les machines.

- Les caractéristiques des critères**

Le troisième Champ $\gamma \in \{f_{max}, \sum f_j\}$ indique un critère d'optimisation.

Exemples:

$f_{max} = C_{max}$ (makespan).

$\sum f_j = \sum W_i C_i$

18.Représentation des solutions avec Le diagramme de Gantt

Gantt développa une représentation graphique de déroulement des projets qui a pris plus tard le nom de **diagramme de Gantt** du nom de son inventeur.

Pour faire le planning de l'exécution des projets et pour en contrôler le Déroulement, on utilise le diagramme à barres (Bar Chart) ou diagramme de Gantt. Chaque tâche est

symbolisée par un rectangle dans lequel sont inscrits le code et la durée de la tâche. En commençant par représenter les tâches avec précedence α . Une tâche est portée dans le diagramme lorsque toutes ses antécédentes y sont déjà portées. Ceci est toujours possible, car une tâche ne se succède pas à elle-même, les rectangles sont ainsi toujours placés à droite (c'est le même principe que celui des niveaux d'un graphe sans circuit). Le diagramme de Gantt est le plus répandu et plus simple pour visualiser graphiquement l'exécution des tâches et/ou l'occupation des ressources au cours du temps.

Exemple

Dix tâches dont les caractéristiques sont résumées dans le tableau, sont à réaliser pour mener à bien un projet.

Tâche	A	B	C	D	E	F	G	H	I	J
Pi	4	5	6	3	7	5	8	6	5	6
Contrainte De précédence (α)	-	-	A,B	A,B	A,B	C	D,E	D,E	F,G,H	F,G,H

Exemple de projet

La figure suivante, présente un diagramme de Gantt associé à une solution minimisant la durée ; le projet est achevé en 26 jours.