Chapitre 1

L'ordonnancement de projet

Présentation

Un problème d'ordonnancement de projet (PSP : Projet Scheduling Problem) consiste à gérer et contrôler la mise en œuvre d'un projet comportant de nombreuses tâches dont il faut planifier l'exécution en respectant diverses contraintes. Ces projets concernent généralement la réalisation d'un produit unitaire complexe : construction d'un bâtiment, d'une usine ; élaboration d'un ouvrage d'art ; mise en place d'un chantier industriel ; fabrication d'une machine spécifique sur commande ; organisation d'un grand événement ; etc.

D'un point de vue historique, ce domaine est un des premiers de la Recherche Opérationnelle. A la fin des années 50, la nécessité de développer des méthodes pour résoudre des problèmes PSP est apparu simultanément aux États Unis et en Europe :

- en 1957, des sociétés américaines mettent au point la *méthode du chemin critique CPM* ("Critical Path Method") ou encore *méthode PERT* ("Programming Evaluation and Review Technics") – nom initialement utilisé pour la situation de durée aléatoire des tâches –, notamment à l'occasion de lancement de fusées "Polaris" et des premiers satellites,
- en 1958, la société française "Sema–Metra" introduit la *méthode MPM* (Méthode des Potentiels Metra) en vue de résoudre les problèmes posés par la construction du paquebot "France" et celle de centrales électriques de la société EDF.

  Grâce à leur simplicité et leur efficacité, ces méthodes ont été ensuite appliquées dans de nombreux domaines industriels (chimie, pétrole, construction, etc.).

  L'objectif des méthodes CPM – PERT et MPM est de minimiser la durée d'exécution d'un projet tout en satisfaisant des contraintes temporelles entre les tâches. Ces méthodes fournissent également les éléments nécessaires pour assurer le suivi et le contrôle de l'exécution du projet ou pour réagir à des événements impondérables durant celle-ci.

Par la suite, diverses extensions ont été proposées :

- *L'analyse des ressources*

  Souvent la réalisation des tâches nécessite l'utilisation de ressources et il convient de respecter des contraintes de limitation des ressources donnant lieu aux problèmes RCPSP (Resource Constrained Project Scheduling Problem).

– *L'analyse des coûts*
  La durée d'exécution des tâches peut être diminuée moyennant un coût supplémentaire et l'on peut donc étudier l'interaction de la durée d'exécution du projet avec le coût de celui-ci.

Ce chapitre comprendra trois paragraphes.
Le § 1.1 traitera du problème PSP de base relatif à la minimisation de la durée d'exécution du projet sous des contraintes temporelles entre les tâches. Nous y décrirons, dans le cas de durée déterministe, la méthode CPM (§ 1.1.1) qui utilise une représentation "graphe-événements" et la méthode MPM (§ 1.1.2) qui utilise une représentation "graphe-tâches". Ces deux méthodes seront comparées au § 1.1.3, mettant en évidence leurs avantages et inconvénients. Le cas de durée aléatoire des tâches sera analysé au §1.1.4 avec la méthode PERT.
L'analyse des ressources sera l'objet du § 1.2. La modélisation générale sous forme de problème linéaire en variables mixtes (MILP) sera formulée au § 1.2.1. Ensuite, seront décrits des algorithmes heuristiques, tant pour la situation de *ressources renouvelables* ("renewal") du type machines, main d'œuvre, etc., c'est-à-dire des moyens qui restent disponibles après chaque utilisation (§ 1.2.2), que pour la situation de *ressources consommables* ou non-renouvelables ("non-renewal") du type matières premières, moyens financiers, etc., c'est-à-dire des moyens diminuant au fur et à mesure de leur utilisation (§ 1.2.3).
Le § 1.3 s'intéresse à l'analyse des coûts. Nous traiterons principalement
– de la définition et de la formulation du problème général (§ 1.3.1)
– de la diminution du coût total d'un ordonnancement sans variation de sa durée (§ 1.3.2)
– de la diminution de la durée d'un ordonnancement au moindre coût (§ 1.3.3)
Quelques exercices et applications clôtureront cette partie au § 1.4.

De nombreux ouvrages spécialisés traitent plus en détail de l'ordonnancement de projet : en particulier des ouvrages précurseurs tels [8], [12] et des ouvrages plus récents [13], [14], [2], [10] ou un ouvrage orienté vers les exercices [1].

1.1 Minimisation du délai d'exécution d'un projet

Dans ce paragraphe, nous considérons un projet qui n'est soumis qu'à des contraintes temporelles ("temporal project scheduling").
L'objectif est d'ordonnancer – c'est-à-dire de planifier dans le temps – les différentes tâches du projet de façon à réaliser le projet en le moins de temps possible, tout en respectant bien sûr les contraintes temporelles entre les tâches.
L'application concrète d'une méthode d'ordonnancement nécessite un travail préliminaire d'information et de recueil des données du cas étudié. Il convient
– de décomposer le projet en ses différentes tâches ;
– de définir les contraintes temporelles liant ces tâches ;
– d'estimer le plus précisément ces contraintes, ainsi que les durées des tâches.
C'est donc un travail minutieux dont il ne faut pas sous estimer la difficulté... et que nous supposerons accompli pour les différents exemples envisagés.

1.1.1 La méthode CPM (ou $PERT$)

Soit un projet dont les tâches sont soumises uniquement à des contraintes de postériorité stricte ("precedence constraints") du type "la tâche j ne peut débuter que si la tâche i est terminée".
Si x_i représente l'instant de début d'une tâche i et d_i sa durée, une contrainte de postériorité stricte s'écrit

$$x_j \geq x_i + d_i$$

Dans la méthode du chemin critique (CMP) les données sont représentées sous forme d'un "graphe-événements" (dénommé en anglais AOA (Activity-on-arc) network) ;
- les *sommets* représentent des *étapes* (ou événements ou instants), notés $m, n, \ldots$.
  On introduit un sommet initial (DEB ou O) et un sommet final (FIN ou N).
- les *arcs* représentent les *tâches* et les valeurs portées sur les arcs indiquent les durées des tâches.
  Une tâche i est donc représentée par la figure 1.1
  et une contrainte de postériorité stricte par la figure 1.2

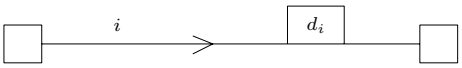


FIGURE 1.1 – Tâche dans un graphe-événements

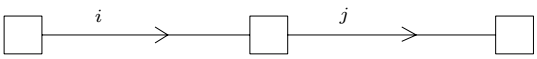


FIGURE 1.2 – Contrainte de postériorité stricte dans un graphe-événements

Avec les notations classiques de la théorie des graphes (voir partie VI du tome 1)

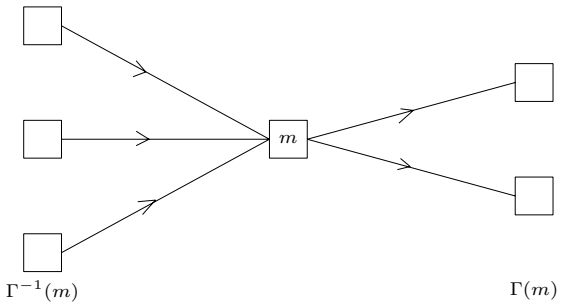


FIGURE 1.3 – Ensemble des prédécesseurs et successeurs directs d'un sommet

- $\Gamma^{-1}(m)$ désigne l'ensemble des sommets précédents directs du sommet m
- $\Gamma(m)$ désigne l'ensemble des sommets suivants directs du sommet m

Nous imposerons que chaque couple de sommets soit relié par au plus un arc, de sorte que l'on peut parler de la tâche (m, n), qui relie le sommet m au sommet n, de durée $d_{(mn)}$.

Remarque
Pour satisfaire cette convention, il peut être nécessaire d'introduire des tâches fictives de durée nulle.
C'est le cas si plusieurs tâches sont simultanées comme indiqué à la figure 1.4.a) bien qu'il soit aussi possible de ne retenir que la tâche de durée la plus longue, comme indiqué à la figure 1.4.b).

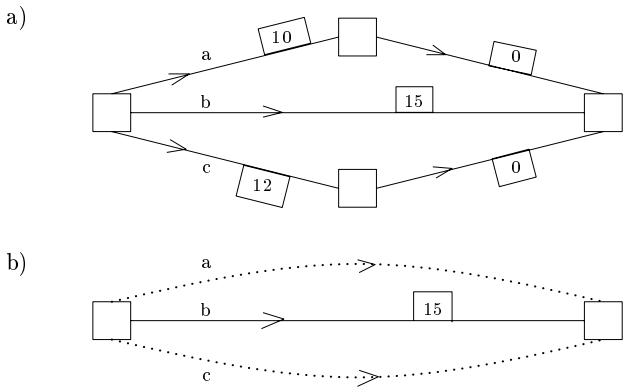


FIGURE 1.4 – Tâches a,b,c simultanées

Toutefois l'introduction de tâches fictives de durée nulle est parfois une nécessité. Ce sera par exemple le cas (voir figure 1.5) pour représenter la situation où
– la tâche c ne peut débuter que si les tâches a et b sont terminées,
– la tâche d ne peut débuter que si la tâche a est terminée.

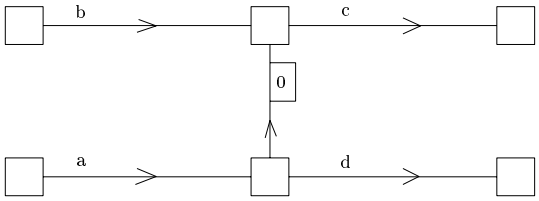


FIGURE 1.5 – Introduction d'une tâche fictive de durée nulle

Si le problème est bien posé, à l'évidence ce graphe-événements ne comporte pas de circuit.
Il convient donc d'être attentif à bien exprimer les différents contraintes de précédence entre les tâches dans l'élaboration du graphe-événements.
Étant donné cette absence de circuit, il est possible de *numéroter les sommets du graphe-événements selon l'ordre croissant des rangs.* (voir définition 19.11 du tome 1).
Rappelons
– qu'un sommet sans prédécesseurs (ici DEB) est de rang 0 ;
– qu'un sommet est de rang k si tous ses prédécesseurs sont de rang inférieur à k.
Les sommets de même rang sont numérotés dans un ordre quelconque et la procédure est appliquée jusqu'à numéroter le sommet FIN (soit sommet N).

Remarque
Bien évidemment pour un projet comprenant de très nombreuses tâches, il est difficile de représenter le graphe-événements. Comme nous l'indiquerons pour l'illustration 1.1, il est toutefois possible, une fois obtenue la numérotation des événements, de représenter les données sous forme d'une matrice D de dimension $(N+1 \times N+1)$ dont l'élémént (m,n) avec $m < n$, vaut la durée $d_{(mn)} = d_i$ de la tâche $i = (m,n)$.

A. Dates des étapes

Définition 1.1 Date au plus tôt de l'étape m : t_m
C'est la date représentant le premier instant auquel on peut s'attendre à la fin de l'exécution de toutes les tâches qui précèdent cette étape. Cette date au plus tôt de l'étape m est notée t_m et correspond à la longueur du chemin le plus long pour aller du sommet initial DEB au sommet m.

Pour calculer ces dates au plus tôt, nous pouvons donc appliquer l'algorithme de Dijkstra utilisant de manière élémentaire la programmation dynamique pour calculer la longueur des chemins les plus longs du sommet DEB aux différents sommets m (voir § 20.1.1 du tome 1). En posant initialement $t_0 = 0$, les dates t_m sont donc déterminées par la formule récurrente de $m = 0$ à $m = N$.

$$\begin{aligned} t_{DEB}(= t_0) &= 0 \\ t_m &= \max_{l \in \Gamma^{-1}(m)} (t_l + d_{(lm)}) \end{aligned} \tag{1.1}$$

Définition 1.2 Durée minimale d'exécution du projet
La date au plus tôt t_N du sommet FIN correspond à la durée minimale d'exécution du projet.

Définition 1.3 Date au plus tard de l'étape m : $t^\star_m$
C'est le dernier instant auquel doit se terminer toutes les tâches qui précèdent cette étape sous peine d'allonger la durée d'exécution minimale t_N du projet.
Cette date au plus tard de l'étape m est notée $t^\star_m$ et correspond à la différence entre la durée minimale du projet $(= t_N)$ et la longueur du chemin le plus long pour aller du sommet m au sommet FIN.

Afin de respecter la durée minimale du projet, on pose d'abord

$$t^\star_N = t_N$$

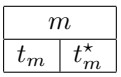
Ensuite, l'algorithme de Dijkstra est à nouveau appliqué pour calculer la longueur des chemins les plus longs des différents sommets m au sommet FIN et ces valeurs sont soustraites de $t^\star_N$. Les dates $t^\star_m$ sont donc déterminées par la formule récurrente de $m = N$ à $m = 0$.

$$\begin{aligned} t^\star_N &= t_N \\ t^\star_m &= \min_{p \in \Gamma(m)} (t^\star_p - d_{(mp)}) \end{aligned} \tag{1.2}$$

Ces deux dates t_m et $t^\star_m$ de l'étape m peuvent également être calculées sur la matrice D associée au graphe : pour les formules (1.1), les sommets $l \in \Gamma^{-1}(m)$ sont ceux situés

dans la colonne m avec une valeur positive $d_{(lm)}$ tandis que pour les formules (1.2), les sommets $p \in \Gamma(m)$ sont ceux situés dans la ligne m avec une valeur positive $d_{(mp)}$ (voir table 1.2 de l'illustration 1.1).

Remarque
La numérotation m d'un sommet , et ses dates au plus tôt t_m et au plus tard $t_m^\star$, sont indiquées de la façon suivante sur le graphe



B. Dates des tâches

Soit une tâche i allant du sommet m au sommet n telle que représentée à la figure 1.6

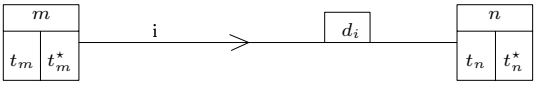


FIGURE 1.6 – Tâche i allant de l'événement m à l'événement n

A chaque tâche i, on associe quatre dates :

Définition 1.4 Début au plus tôt de la tâche i : $ES(i)$
C'est la date la plus proche, notée $ES(i)$ ("Earliest Starting date") à laquelle il est possible de commencer la tâche i :

$$ES(i) = t_m \tag{1.3}$$

Définition 1.5 Fin au plus tôt de la tâche i : $EF(i)$
C'est la date la plus proche, notée $EF(i)$ ("Earliest Final date") à laquelle il est possible de terminer la tâche i :

$$EF(i) = ES(i) + d_i \tag{1.4}$$

Définition 1.6 Fin au plus tard de la tâche i : $LF(i)$
C'est la date la plus éloignée, notée $LF(i)$ ("Latest Final date") à laquelle la tâche i doit se terminer sous peine d'allonger la durée d'exécution minimale du projet :

$$LF(i) = t_n^\star \tag{1.5}$$

Définition 1.7 Début au plus tard de la tâche i : $LS(i)$
C'est la date la plus éloignée, notée $LS(i)$ ("Latest Starting date") à laquelle la tâche i doit commencer sous peine d'allonger la durée d'exécution minimale du projet :

$$LS(i) = LF(i) - d_i = t_n^\star - d_{(mn)} \tag{1.6}$$

Ces quatre dates fournissent pour chaque tâche i

$$\begin{cases} - \text{ un intervalle de début} & : & [ES(i), LS(i)] \\ - \text{ un intervalle de fin} & : & [EF(i), LF(i)] \end{cases}$$

Définition 1.8 Ordonnancement au plus tôt et au plus tard
On parle d'ordonnancement au plus tôt lorsque toutes les tâches sont planifiées le plus tôt possible, (c'est-à-dire débutent en les $ES(i)$ et se terminent en les $EF(i)$).

On parle d'ordonnancement au plus tard lorsque toutes les tâches sont planifiées le plus tard possible, (c'est-à-dire débutent en les $LS(i)$ et se terminent en les $LF(i)$), sous entendu de façon à respecter la durée d'exécution du projet.

L'ordonnancement au plus tard est donc relatif à une date de fin prévue du projet, qui sans autre précision, est la date minimale de fin du projet.

C. Marges totales ; tâches critiques ; marges libres

Définition 1.9 Marge totale d'une tâche
La marge totale ("total float") d'une tâche i – notée $MT(i)$ – représente le retard que peut subir cette tâche par rapport à son exécution au plus tôt, sans allonger la durée totale du projet.

C'est donc la largeur commune des intervalles de début et de fin de tâche :

$$MT(i) = LF(i) - EF(i) = LS(i) - ES(i) = t_n^\star - (t_m + d_{(mn)}) \tag{1.7}$$

Si une tâche subit un retard inférieur ou égal à sa marge totale, il est possible qu'il soit nécessaire de replanifier des tâches suivantes mais cela peut se faire sans remettre en cause la date prévue pour la fin du projet.

Définition 1.10 Tâche critique et chemin critique
Une tâche critique est une tâche pour laquelle $MT(i) = 0$:
tout retard apporté à l'exécution d'une telle tâche retarde l'échéance finale pour le projet.
Un chemin critique est un chemin allant du sommet DÉBUT au sommet FIN et composé de tâches critiques.

Il existe toujours au moins un chemin critique mais il n'est pas nécessairement unique.
La durée d'exécution minimale t_N du projet est égale à la longueur commune des chemins critiques qui sont les chemins les plus longs allant du sommet DÉBUT au sommet FIN.

Définition 1.11 Marge libre d'une tâche
La marge libre ("free float") d'une tâche i – notée $ML(i)$ – représente le retard que peut subir cette tâche par rapport à son exécution au plus tôt, sans retarder l'exécution d'une tâche suivante.
Elle est donc déterminée par la relation

$$ML(i) = t_n - (t_m + d_{(mn)}) \tag{1.8}$$

Si une tâche subit un retard inférieur ou égal à sa marge libre, il n'est donc pas nécessaire de remanier l'ordonnancement prévu.

Propriété 1.1 Relation entre les marges totale et libre

$$0 \leq ML(i) \leq MT(i)$$

Par conséquent, la marge libre d'une tâche critique est nulle.

Propriété 1.2 Détermination de la marge libre à l'aide des marges totales

$$\left[\begin{array}{l} ML(i) = MT(i) - \min\limits_{j\in\Gamma(i)} MT(j) \\ \textit{où } \Gamma(i) \textit{ représente l'ensemble des tâches qui suivent directement la tâche } i \\ \textit{(voir figure 1.7).} \end{array}\right.$$

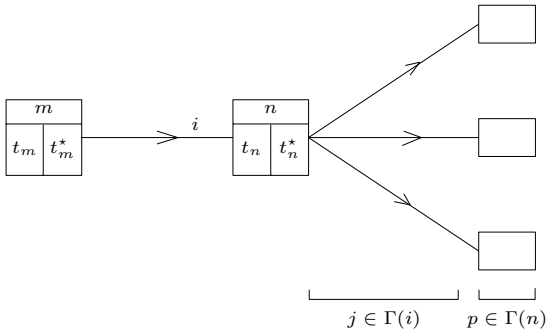


FIGURE 1.7 – Ensemble $\Gamma(i)$ des tâches qui suivent directement la tâche i

Démonstration

$$\begin{array}{rcl} ML(i) & = & t_n - (t_m + d_{(mn)}) \\ & = & t^\star_n - (t_m + d_{(mn)}) - (t^\star_n - t_n) \\ & = & MT(i) - \Big(\min\limits_{p\in\Gamma(n)} (t^\star_p - d_{(np)}) - t_n\Big) \\ & = & MT(i) - \min\limits_{p\in\Gamma(n)} (t^\star_p - (t_n + d_{(np)})) \end{array}$$

Par conséquent $ML(i) = MT(i) - \min\limits_{j\in\Gamma(i)} MT(j)$ ◇

Cette propriété fournit donc un autre moyen de calcul – indépendant des dates des étapes – de la marge libre qui peut s'avérer plus aisé.

Illustration 1.1
Considérons le problème d'ordonnancement de projet défini par les trois premières colonnes de la table 1.1 :

- la première colonne indique le nom des 21 tâches i, appelées de a à u,
- seules des contraintes de postériorité stricte sont considérées et la deuxième colonne indique l'ensemble des tâches, noté $\Gamma^{-1}(i)$, qui doivent être terminées avant que la tâche i puisse débuter,
- la troisième colonne indique la durée $d(i)$ de la tâche i.