

CHAPITRE 1

ELEMENT FONDAMENTAUX

D'ORDONNANCEMENT

1 Introduction

Les problèmes d'optimisation combinatoire abordés dans la littérature sont la plupart du temps issus de situations concrètes de l'industrie ou de service. Ils représentent une vue simplifiée d'une situation, et leur résolution aide à la prise de décision.

Un problème d' optimisation combinatoire est un problème qui consiste à chercher une meilleure solution parmi un ensemble de solutions réalisables. Le plus souvent, un problème d'optimisation se présente comme un objectif à poursuivre (maximisation d'un profit, minimisation d'une distance, ou d'un coût, etc.). Cet objectif peut être monocritère ou multicritère. Dans ce qui suit on ne s'intéresse qu'aux problèmes monocritères.

Les solutions possibles à ces problèmes doivent respecter un ensemble de contraintes, liées à la situation étudiée. Parmi les techniques de formulation de ces problèmes on cite la programmation linéaire, dynamique,... ainsi que les problèmes d'optimisation combinatoire en théorie des graphes. Cette thèse se penche sur l'étude d'un des problèmes combinatoires, à savoir les problèmes d'ordonnancement. [20]

2 Recherche opérationnelle

La recherche opérationnelle (RO) est la discipline des mathématiques appliquées qui traite des questions d'utilisation optimale des ressources dans l'industrie et dans le secteur public. Depuis une dizaine d'années, le champ d'application de la RO s'est élargi à des domaines comme l'économie, la finance, le marketing et la planification d'entreprise. Plus récemment, la RO a été utilisée pour la gestion des systèmes de santé et d'éducation, pour la résolution de problèmes environnementaux et dans d'autres domaines d'intérêt public. [18]

3 Problèmes d'optimisation combinatoire

L'optimisation combinatoire est une étude qui combine différentes techniques de mathématiques discrètes et de l'informatique afin de résoudre des problèmes d'optimisation. Ce genre de problème apparaît dans des domaines divers, tels que l'industrie, le transport, les télécommunications,..., ect.

La résolution d'un problème d'optimisation combinatoire consiste à explorer un espace de recherche afin de maximiser (ou minimiser) une certaine fonction donnée sur un ensemble fini de solutions. Cet ensemble des solutions du problème, permet de représenter diverses structures combinatoires, comme par exemple des chemins, des cycles, des arbres, etc, dans les graphes.

Bien qu'il soit fini, l'ensemble de solutions d'un problème d'optimisation combinatoire peut avoir un nombre très grand de solutions. Les complexités (en taille ou en structure) relatives de l'espace de recherche et de la fonction (appelée fonction de coût, fonction d'adéquation ou fitness) à maximiser ou à minimiser (appelée optimum global) conduisent à utiliser des méthodes de résolution radicalement différentes. Une méthode énumérative consiste à l'énumération des solutions du problème ne peut donc être étudiée même si la puissance des calculateurs augmentait considérablement. Des outils plus performants ont été développés pour aborder ce type de problèmes, comme la programmation linéaire, les méthodes de la recherche opérationnelle. [9]

4 Complexité

4.1 Complexité d'un algorithme

On désigne par complexité d'un algorithme le nombre d'opérations nécessaires à celui-ci pour s'exécuter. Bien évidemment, ce nombre peut varier en fonction de ce que l'on appelle les données d'entrées, c'est-à dire les paramètres que l'on donne à l'algorithme. Par exemple, un algorithme de tri d'éléments dans un tableau ne s'exécutera pas avec le même nombre d'opérations s'il y a 10 éléments ou s'il y en a 100.

Ainsi, on cherchera à estimer la complexité d'un algorithme en fonction de la taille des données entrées. Par exemple, dans le cas du tableau, on exprimera la complexité en fonction de la taille du tableau. Pour une matrice, ce serait en fonction de sa largeur et de sa hauteur. De plus, la nature même des données pour une même taille peut ne pas aboutir au même nombre d'opérations à l'exécution de l'algorithme. En effet, si le tableau est déjà trié,

l'exécution de l'algorithme de tri risque d'être très rapide comparée au cas d'un tableau totalement en désordre. C'est pour cela que l'on estime le nombre d'opérations dans le pire des cas.[26]

4.2 Algorithme efficace

Une fois la complexité définie de manière succincte, on peut tenter de définir ce qu'est un algorithme efficace. Un algorithme sera dit efficace si sa complexité est bornée par un polynôme ayant la taille des données comme variable. Par exemple, un algorithme qui a en entrée un tableau de n éléments et qui a une complexité de n^2 est un algorithme efficace. On dit aussi que l'algorithme est polynomial. Cette définition est justifiée par le fait qu'on s'intéresse aux performances des algorithmes quand la taille des données en entrée devient très importante.

4.3 Problèmes faciles et problèmes difficiles

Tout d'abord, on fait une distinction entre les problèmes décidables et les problèmes indécidables. Les problèmes indécidables sont ceux pour lesquels aucun algorithme, quel qu'il soit, n'a été trouvé pour les résoudre. Ainsi, les problèmes décidables sont ceux pour lesquels il existe au moins un algorithme pour les résoudre.[26]

4.3.1. La classe P

La classe P, qui regroupe les problèmes les plus simples de la classe NP, contient les problèmes pour lesquels on connaît au moins un algorithme polynomial pour les résoudre. Pour le reste de la classe NP, on n'est pas sûr qu'il n'existe pas un algorithme polynomial pour résoudre chacun de ses problèmes. Ainsi, on sait que P est inclus dans NP mais on n'a pas pu prouver que P n'est pas NP. [26]

4.3.2. La classe NP

Parmi les problèmes décidables, les plus simples à résoudre sont regroupés dans la classe NP. Un problème appartient à la classe NP si quelqu'un ayant la solution au problème peut démontrer que c'est la solution en un temps polynomial. Les autres problèmes décidables sont considérés comme très difficiles. La classe NP est également décomposée en trois catégories

qui permettent d'identifier les problèmes les plus simples et les problèmes les plus compliqués de la classe.[26]

4.3.2.1.La classe NP-Complet

La classe NP-Complet regroupe les problèmes les plus difficiles de la classe NP. Elle contient les problèmes de la classe NP tels que n'importe quel problème de la classe NP leur est polynomialement réductible. Entre eux, les problèmes de la classe NP-Complet sont aussi difficiles. [26]

4.3.2.2. La classe NP-Difficile

La classe NP-Difficile regroupe les problèmes (pas forcément dans la classe NP) tels que n'importe quel problème de la classe NP leur est polynomialement réductible. [22]

Tableau récapitulatif :

Décidables	Classe NP	Classe P (plus court chemin, arbre de poids min, flot maximum, flot de coût minimum,...)	Classe NP-Difficile
		Classe NP-Complet (Voyageur de commerce, Sac à dos,...)	
Indécidables			

Table 1.1 : les classes de problème d'ordonnancement

5 l'ordonnancement

L'ordonnancement est une branche de la recherche opérationnelle et de la gestion de la production qui vise à améliorer l'efficacité d'une entreprise en termes de coûts de production et de délais de livraison.

On peut rencontrer les problèmes d'ordonnancement dans de très nombreux domaines : les systèmes industriels de production (activités des ateliers en gestion de production et

problèmes de logistique), les systèmes informatiques (les tâches sont les programmes et les ressources sont les processeurs, la mémoire...), les systèmes administratifs (gestion du personnel, emplois du temps,...), les systèmes de transport, la construction, ... etc.

Les différentes données d'un problème d'ordonnancement sont les tâches, les ressources, et les contraintes.[16]

5.1 Définition

Le problème d'ordonnancement consiste à organiser dans le temps la réalisation d'un ensemble de tâches, compte tenu de contraintes temporelles (délais, contraintes déchainements, ...) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises.[6]

5.2 Les tâches

Une tache est une entité élémentaire localisée dans le temps par une date de début et/ou de fin, dont la réalisation nécessite une durée, et qui consomme un moyen selon une certaine intensité. Selon les problèmes, les taches peuvent être exécutées par morceaux, ou doivent être exécutées sans interruption ; on parle alors respectivement préemptifs et non préemptifs. Lorsque les taches ne sont soumises à aucune contrainte de cohérence, elles sont dites indépendantes. [12]

5.3 Les ressources

La ressource est un moyen technique ou humain destiné à être utilisé pour la réalisation d'une taches et disponible en quantité limitée, sa capacité. Plusieurs types de ressources sont à distinguer. Une ressource est renouvelable si après avoir été allouée à une ou plusieurs taches, elle est à nouveau disponible en même quantité (les hommes, les machines, l'équipement en générale) ; la quantité de ressource utilisable à chaque instant est limitée. Dans le cas contraire, elle est consommable (matières premières, budget) ; la consommation globale (ou cumul) au cours du temps est limitée. Une ressource est doublement contrainte lorsque son utilisation instantanée et sa consommation globale sont toutes deux limitées (l'argent en est un bon exemple). [12]

5.4 Les contraintes

Les contraintes expriment des restrictions sur les valeurs que peuvent prendre certaines variables. On distingue deux types de contraintes : les contraintes temporelles et les contraintes de ressources.

5.4.1 Les contraintes temporelles

Les contraintes temporelles comprennent les contraintes de temps alloué, qui correspondent généralement aux impératifs liés aux tâches (délais de livraisons, disponibilité des approvisionnements) ou encore à la durée totale d'un ordonnancement. Elles comprennent les contraintes d'antériorité ou de précédence qui correspondent à des contraintes de cohérence technologique qui positionnent les tâches les unes par rapport aux autres. Elles comprennent aussi les contraintes de calendrier qui correspondent, par exemple, aux plages horaires de travail, etc.

5.4.2 Les contraintes de ressources :

Les contraintes de ressources, quant à elles, traduisent la disponibilité des ressources et le fait qu'elle soit en quantité limitée. Deux types de contraintes de ressources, liées à la nature cumulative ou disjonctive des ressources, peuvent alors être distingués.

Les ressources disjonctives ne peuvent être utilisées que par une tâche à la fois. Les ressources cumulatives, quant à elles, peuvent être utilisées par plusieurs tâches simultanément, comme dans le cas d'un ensemble de ressources. [25]

6 Caractéristiques générales des ordonnancements

6.1 Ordonnements statique et dynamique

Un problème d'ordonnement est statique si l'ensemble des informations nécessaires à sa résolution est fixé a priori et n'est pas remis en cause durant la résolution (ensemble des tâches, des ressources, et leurs caractéristiques). La solution est alors un plan prévisionnel dont l'exécution nécessite un contrôle d'exécution. Celui-ci peut inclure une fonction de décision en temps réel si le système est doté d'une certaine souplesse.[21]

6.2 Ordonnements actifs et semi-actifs

Pour Ordonnements semi-actif : Si aucun travail, ne peut avancé sur la ressource où il se trouve, compte tenu des contraintes de gamme et de précédence. La longueur de l'ordonnement correspond au chemin le plus long dans le graphe potentiel-tâche. Aucun glissement à gauche local n'est possible : on ne peut avancer une tâche sans modifier la séquence sur la ressource.

Pour Ordonnements actif : si aucune opération d'un travail i ne peut débiter son exécution plus tôt sans déplacer au minimum une autre opération. Aucun glissement à gauche, local ou global, n'est possible. Aucune tâche ne peut être commencée plus tôt sans reporte le début d'une autre.[3]

6.3 Ordonnements sans retard

Lorsqu'aucune machine n'est laissée inoccupée, ceci alors qu'une file d'attente contient au moins un travail susceptible de débiter son exécution sur cette machine. Lorsqu'une méthode parcourt l'ensemble des ordonnancements sans délai, elle n'est pas capable de trouver la solution optimale, on ne laisse pas une machine inoccupée alors qu'une file contient des tâches. On ne doit pas retarder l'exécution d'une tâche si celle-ci est en attente et si la ressource est disponible.[3]

6.5 Ordonnancement préemptif et non préemptif

- Un ordonnanceur préemptif peut interrompre une tâche au profit d'une tâche plus prioritaire
- Un ordonnanceur non préemptif n'arrête pas l'exécution de la tâche courante

7 Classification des problèmes d'ordonnancement

- Problème de gestion de projet à contraintes de ressources et sa représentation graphique, le réseau PERT.
- Problème d'ordonnancement d'atelier
 - Job-shop
 - Flow-shop
 - Open-shop
- Problèmes d'ordonnancement dans les systèmes d'exploitation.
- Problèmes d'ordonnancement de tâches informatiques. [32]

8 Méthodes de résolution des problèmes d'ordonnancement

Les méthodes d'optimisation peuvent être réparties en deux grandes classes de méthodes pour la résolution des problèmes :

- Les méthodes exactes.
- Les méthodes approchées.

8.1 Les méthodes exactes

Le principe des méthodes exactes consiste à rechercher, souvent de manière implicite, une solution, la meilleure solution ou l'ensemble des solutions d'un problème. L'optimisation exacte concerne toutes les méthodes permettant d'obtenir un résultat dont on sait qu'il est optimal à un problème précis. On peut classer les méthodes exactes en quatre grandes classes :

- La programmation dynamique,
- La programmation linéaire continue ou en nombres entiers,
- La programmation non linéaire avec ou sans contraintes,

- Les méthodes de recherche arborescente (Branch & Bound). [13]

8.2 Les méthodes approchées

Les méthodes approchées fournissent une solution approchée au problème traité. Elles sont en général conçues de manière à ce que la solution obtenue puisse être proche de la valeur optimale : de telles méthodes permettent d'obtenir des bornes inférieures ou supérieures de la valeur optimale telles que :

- Méthodes Heuristiques
- Méthodes Méta heuristiques

8.1.2.1 Les méthodes Heuristiques

Une heuristique est plutôt une méthode qui cherche (une stratégie) sans garantir le résultat. Destiné à un problème spécifique, le temps de calcul est raisonnable sans garantir la faisabilité ou l'optimalité. [2]

En recherche opérationnelle, les heuristiques sont des règles empiriques simples qui ne sont pas basées sur l'analyse scientifique (différents algorithmes). Elles sont basées sur l'expérience, les résultats déjà obtenus et sur l'analogie pour optimiser les recherches suivantes. Généralement, on n'obtient pas la solution optimale mais une solution approchée. Parmi les heuristiques, on peut citer l'algorithme glouton). [38]

8.1.2.2 Les Métaheuristiques

Une métaheuristique est constituée d'un ensemble de concepts fondamentaux (par exemple, la liste taboue et les mécanismes d'intensification et de diversification pour la métaheuristique tabou), qui permettent d'aider à la conception de méthodes heuristiques pour un problème d'optimisation. Ainsi les métaheuristiques sont adaptables et applicables à une large classe de problèmes. [19]

Les métaheuristiques se subdivisent en deux sous-classes :

- Les méthodes de voisinage.
- Les algorithmes évolutionnaire.

9 Conclusion

L'objectif essentiel de ce chapitre était de présenter les éléments fondamentaux d'ordonnancement par la représentation des problèmes d'optimisation combinatoire et les complexités de ces problèmes et les classements.

Dans un second temps, nous avons fait un survol de l'ordonnancement et les contraintes puis les caractéristiques générales de l'ordonnancement et les méthodes d'ordonnancement pour les résoudre.