الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Nº Réf :……

Centre Universitaire
Abdelhafid Boussouf Mila

Institut des sciences et de la technologie
Département des Mathématiques et Informatique

Mémoire Préparé En Vue de L'obtention Du Diplôme De Licence
En : - Filière Mathématiques

Planification D'un Problème D'ordonnancement

Préparer par : -LEBCIR Souad
-BELFAR Leila
-BOULESNANE Saida

ENCADREUR : ZAIDI. Ali

Année universitaire : 2014/2015

REMERCIEMENTS

ON REMERCIER TOUT D'ABORD NOTRE DIEU
QUI NOUS A DONNE LA FORCE POUR TERMINER
CE MODESTE TRAVAIL.

Nous tenons à remercier, en premier lieu, notre promoteur ZAIDI ALI pour ses conseils et ses encouragements pendant tout le suivi de ce travail.

Nos chaleureux remerciements vont plus particulièrement à nos enseignants du Département de science et technologie (M I), grâce au savoir qu'ils nous ont inculqué durant tout le long de notre cursus universitaire.

Et à tous ceux qu'on aura oubliés nous leur

adressons également un grand merci….

…..que Dieu vous bénisse tous.

……..MERCI……

Dedicaces

Je dédie ce modeste travail à :

❖ *A ma très* ***chère mère*** *NAANAA qui a beaucoup sacrifié à mon bonheur, qui a partagé mes malheurs et qui n'a jamais cessé de prier pour moi.*

❖

❖ *A* ***mon père*** *MAMAR qui m'a beaucoup conseillé de continuer mes études.*

A ***mes sœurs*** *SAMIA, SOUHILA , SALIHA, DALAL, FATIHA,AMIRA et* ***mes frères*** *HASSANE, HOUCIN, MOHAMADE .*

A tous ***mes amis*** *: SOUAD, SAIDA , ROKIA.B , ROKIA , IMANE, ASMA, SOUMA.*

BELFAR LEILA

Dedicaces

Je dédie ce modeste travail à :

❖ *A ma très* ***chère mère*** *MANOUBA qui a beaucoup sacrifié à mon bonheur, qui a partagé mes malheurs et qui n'a jamais cessé de prier pour moi.*

❖

❖ *A* ***mon père*** *MAHMOUDE qui m'a beaucoup conseillé de continuer mes études.*

A ***mes sœurs*** *ANISSA, ZOBIDA , ATIKA ,*

et ***mes frères*** *MONIR, BADER ,REDEWHANE et sa femme SOUAD et son fils WAILE.*

A tous ***mes amis*** *: SOUAD, LEILA , ROKIA , HAFIDA , KARIMA, ASMA, KHADIDJA, SARA , ROFIA.*

SAIDA BOULESENANE

Dedicaces

Je dédie ce modeste travail à :

❖ *A ma très* ***chère mère*** *HADRIA qui a beaucoup sacrifié à mon bonheur, qui a partagé mes malheurs et qui n'a jamais cessé de prier pour moi.*

❖

❖ *A* ***mon père*** *MOHAMED qui m'a beaucoup conseillé de continuer mes études.*

A ***mes sœurs*** *WASSILA, LEILA , NAWEL , RAHMA, CHAIMA et* ***mon frère*** *SOUFIANE* ***et sa femme*** *CHAHRA.*

A tous ***mes amis*** *: LEILA, SAIDA , ROKIA.B , ROKIA , IMANE, ISMAHANE, SARA.*

LEBCIR SOUAD

Table des matières

Introduction

Le problème d'ordonnancement fait partie des problèmes d'affectation généraux qui représentent une classe importante des problèmes d'optimisation. L'optimisation est un domaine qui mélange les connaissances des mathématiques appliquées de la recherche opérationnelle.

Le problème d'ordonnancement consiste à organiser dans le temps l'exécution d'opérations interdépendantes à l'aide des ressources disponibles en quantités limitées pour réaliser un plan de production.

En se basant sur les concepts de tâche, ressource, contrainte et objectif, l'ordonnancement peut être défini : « Ordonnancer un ensemble de tâches, c'est programmer leur exécution en leur allouant les ressources requises et en fixant leur date de début ».

Ce mémoire est composé de trois chapitres dont nous présentons une brève description dans les paragraphes suivants :

Dans ce premier chapitre, nous introduisons les notions liées aux problème d'optimisation mono objectif, et les formulations d'un problème d'optimisation mono objectif.

Dans le deuxième chapitre, nous introduisons les généralités de l'ordonnancement qui consiste sur les tâches, contraintes, ressources. nous donnons les objectifs d'ordonnancement qui distingues à trois objectifs, les objectif liées au temps, aux ressources, au côut.nous parelons aussi sur les problèmes d'atelier multi-machine. A la fin de ce chapitre nous representons les méthodes de résolution des problèmes d'ordonnancement qui contient deux méthodes: les méthode exactes, approchés.

Dans le dernière chapitre nous introduisons la méthode PERT, son définition, le principe de cette méthode, notion de base. A la fin de ce chapitre nous étudions un exemple.

Chapitre 1

Optimisation mono-objectif

1.1 Définition de l'optimisation

Un problème d'optimisation se définit comme la recherche du minimum ou du maximum (de l'optimum) d'une fonction donnée . On peut aussi trouver des problèmes d'optimisation pour lequelles les variables de la fonction à optimiser sont contraintes d'évoluer dans une certaine partie de l'espace de recherche. Dans ce cas, on a une forme particulière de ce que l'on appelle un problème d'optimisation sous contraintes.

1.2 Problème d'optimisation mono-objectif

Lorsqu'un seul objectif (critère) est donné, le problème d'optimisation est mono-objectif. Dans ce cas la solution optimale est clairement définie, c'est celle qui a le coût optimal (minimal, maximal). De manière formelle, à chaque instance d'un tel problème est associé un ensemble Ω des solutions potentielles respectant certaines contraintes et une fonction d'objectif $f : \Omega \longrightarrow \Psi$qui associe à chaque solution admissible $s \in \Omega$ une valeur $f(s)$. Résoudre l'instance $(\Omega\,.f)$ du problème d'optimisation consiste à trouver la solution optimale $s^* \in \Omega$ qui optimise (minimise ou maximise) la valeur de la fonction objectif f. Pour le cas de la minimisation : le but est de trouver $s^* \in \Omega$ tel que $f(s^*) \leq f(s)$ pour tout élément $s \in \Omega$. Un problème de maximisation peut être défini de manière similaire.

1.2.1 Variables de décision

Les variables de décision sont des quantités numériques pour les quelles des valeurs sont à choisir. Cet ensemble de $n$variables est appelé vecteur de décision : $(x_1, x_2, \ldots, x_n)$. Les différentes valeurs possibles prises par les variables de décision x_i constituent l'ensemble des solutions potentielles.

1.2.2 Espace décisionnel et espace objectif

Deux espaces Euclidiens sont considérés en optimisation :

· L'espace décisionnel, de dimension n, n étant le nombre de variables de décision. Cet espace est constitué par l'ensemble des valeurs pouvant être prise par le vecteur de décision.

· L'espace objectif : l'ensemble de définition de la fonction objectif, généralement défini dans $\mathbb{R}$. La valeur dans l'espace objectif d'une solution est appelée coût, ou fitness.

1.2.3 Contraintes

Dans la plupart des problèmes d'optimisation, des restrictions sont imposées par les caractéristiques du problème. Ces restrictions doivent être satisfaites afin de considérer une solution acceptable. Cet ensemble de restrictions, appelées **contraintes**, décrit les dépendances entre les variables de décision et les paramètres du problème. On formule usuellement ces contraintes c_j par un ensemble d'inégalités, ou d'égalités de la forme : $c_j(x_1, x_2, \ldots, x_n) \geq 0$.

1.2.4 Formulation d'un problème d'optimisation mono-objectif

Un problème d'optimisation mono-objectif est présenté sous la forme suivante :

Minimiser $f(\overrightarrow{x})$ (fonction à optimiser)

Sujet de :

$\overrightarrow{g}(\overrightarrow{x}) \leq 0$ (m contraintes d'inégalité)

$\overrightarrow{h}(\overrightarrow{x}) = 0$ (p contraintes d'égalité)

Avec $x \in \mathbb{R}^n$, $\overrightarrow{g}(\overrightarrow{x}) \in \mathbb{R}^m$, $\overrightarrow{h}(\overrightarrow{x}) \in \mathbb{R}^p$

Les vecteurs $\overrightarrow{g}(\overrightarrow{x})$ et $\overrightarrow{h}(\overrightarrow{x})$ représentent respectivement m contraintes d'inégalité et p contraintes d'égalité.

1.2.5 Minimum global

On a la fonction $f : \Omega \subseteq \mathbb{R}^n \longrightarrow \mathbb{R}$, tel que $\Omega \neq \varnothing$. Pour $x^* \in \Omega$, on dit que x^* est un optimum global si et seulement si:

$\forall x \in \Omega : f(x^*) \leq f(x)$

Tel que:

- x^* : l'optimum global;
- F : la fonction objective;
- Ω : La région faisable ($\Omega \in s$);
- s : l'espace de recherche global.

1.2.6 Minimum local

Un point x^* est un minimum local de la fonction f si et seulement si:

$f(x^*) \prec f(x)$,$\forall x \in V(x^*)$ et $x^* \neq x$

D'ou $V(x^*)$ définit un voisinage de x^*. [1]

Chapitre 2

Problème d'ordonnancement

2.1 Généralité sur l'ordonnancement

2.1.1 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'enchainements, ...) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises

2.1.2 Tâches

Une tâche est une entité élémentaire organisée dans le temps, par une date de début et/ou de fin, et dont la réalisation nécessite une durée préalablement définie.

On distingue deux types de tâches:

• les tâches morcelables (préemptives) qui peuvent être exécutées en plusieurs fois, facilitant ainsi la résolution de certains problèmes

• les tâches non morcelables (indivisibles) qui doivent être exécutées en une seule fois et ne sont interrompues qu'une fois terminées

2.1.3 Contraintes d'ordonnancement

Une contrainte exprime des restrictions sur les valeurs que peuvent prendre simultanément les variables représentant les relations reliant les tâches et les ressources. On distingue deux types de contraintes, les contraintes temporelles et les contraintes de ressource.

contraintes temporelles

Notation: t_j: date de début de la tâche j, p_j sa durée

Forme générale: $t_j - t_i \geq a_{ij}$

•Localisation temporelle: j ne peut débuter avant une certaine date (livraison de matière premiére, conditions climatiques,...)

•Contrainte de délai: j doit étre terminée avant une certaine date

•Contrainte de succession

- ∘succession simplet
- ∘succession avec attente
- ∘succession avec chevauchement
- ∘succesion immédiate

les contraintes de ressources

Ces contraintes concernent la limitation de la quantité de ressources de chaque type. Dans ce cadre, deux types de contraintes de ressources sont distinguées:

Contraintes disjonctives: Deux tâches i et j sont en disjonction si elles ne peuvent être exécutées simultanément

$\Longrightarrow$ Les intervalles d'exécution des taches disjonctives sont disjoints :

$]t_i, t_i + p_i[\cap]t_j, t_j + p_j[= \varnothing$, Disjonction d'inégalités de potentiels

$t_j - t_i \geq p_i$ ou $t_i - t_j \geq p_j$

Contraintes cumulatives: C'est une généralisation des contraintes disjonctives.

Exemple:

On a deux grues et 5 tâches nécessitant une grue sont candidates au même moment.

Soit

•$w_k(t)$ la quantité de moyen k disponible à t.

•$w_{ik}(t)$la quantité de moyen k nécessaire pour exécuter i à t.

Si

$\sum_{ik} w(t)\ w_k(t)$

alors les tâches de s peuvent être exécutées simultanément à t sinon les tâches de s sont en disjonction.

2.1.4 Ressources

Une ressource est un moyen technique ou humain utilisé pour réaliser une tâche. On trouve plusieurs types de ressources :

•Les ressources renouvelables, qui, après avoir été allouées à une tâche, redeviennent disponibles (machines, personnel, ... etc).

•Les ressources consommables, qui lorsqu'après sa libération, elle n'est pas disponible en même quantité (argent, matières premières, ... etc).

2.2 Objectifs d'ordonnancement

Les objectifs des entreprises se sont diversifiés et le processus d'ordonnancement est devenu de plus en plus multicritère. Les critères que doit satisfaire un ordonnancement sont variés. D'une manière générale, on distingue plusieurs classes d'objectifs concernant un ordonnancement.

2.2.1 Objectifs liés au temps

On trouve par exemple la minimisation du temps total d'exécution, du temps moyen d'achèvement, des durées totales de réglage ou des retards par rapport aux dates de livraison.

2.2.2 Objectifs liés aux ressources

Maximiser la charge d'une ressource ou minimiser le nombre de ressources nécessaires pour réaliser un ensemble de tâches sont des objectifs de ce type.

2.2.3 Objectifs liés au côut

Ces objectifs sont généralement de minimiser les coûts de lancement, de production, de stockage, de transport, etc.

La satisfaction de tous les critères à la fois est souvent délicate, car elle conduit souvent à des situations contradictoires et à la recherche de solutions à des problèmes complexes d'optimisation. problèmes d'atelier multi-machines.

Les problèmes d'ordonnancement peuvent être classifiés selon le nombre de machines et leur ordre d'utilisation pour fabriquer un produit (gamme de fabrication) qui dépend de la nature de l'atelier. Un atelier se définit par le nombre de machines qu'il contient et par son type.

Les différents types possibles sont les suivants :

D'une façon générale ...

On doit exécuter n tàches ou n travaux (jobs).

Le champ α est décomposé en deux sous-champs α_1 et α_2.

Selon les valeurs prises par α_1, on distingue :

- Les problèmes à une machine
- Les problèmes à machines parallèles
- Les problèmes d'ateliers
    - Ateliers à cheminement unique (flowshop)
    - Ateliers à cheminements multiples (jobshop)
    - Ateliers à cheminements libres (openshop)

- L'ordonnancement de projet sous contraintes de ressources

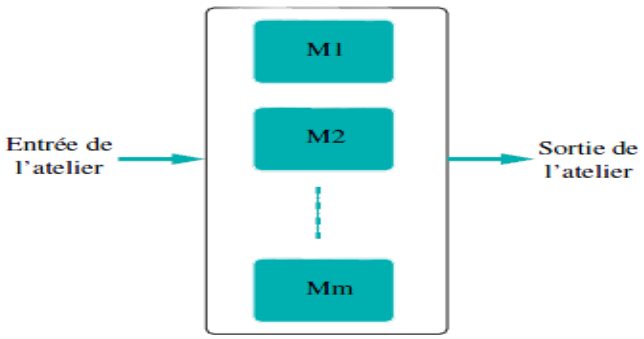
2.2.4 Problèmes à une machine

Toute tâche j_j, $j \in \{1, ..., n\}$ de durée p_j (processing time) s'exécute sur une machine qui ne peut traiter plus qu'une tâche à la fois.

Le champ α_1 est absent et $\alpha_2 = 1$.

2.2.5 Problèmes à machines parallèles

Toute tâche j_j , $j \in \{1, ..., n\}$ peut être exécutée indifféremment sur une des m machines mises en parallèle.



$p_{i,j}$ est la durée d'exécution de J_j sur la machine

$M_{i,i} = 1, ..., m$.

∘si $\alpha_1 = p$ alors machines identiques $\Longrightarrow \forall i, p_{i,j} = p_j$

∘si $\alpha_1 = Q$ alors machines uniformes $\Longrightarrow \forall i \ \ , p_{i,j} = p_j\prime / s_i$ ou s_i est la vitesse de traitement M_i

∘si $\alpha_1 = R$ alors machines ind´ependantes $\Longrightarrow \forall i, p_{i,j} = p_j \ / s_{i,j}$ ou $s_{i,j}$ est la vitesse de traitement de la tache J_j par la machine M_I

∘si α_2 est un entier positif, le nombre de machines est suppose constant. Si α_2 est absent alors ce nombre est suppose arbitraire.

2.2.6 Problèmes d'ateliers

•m machines différentes m_i, $i \in 1, ..., m$

•n travaux (jobs) j_j, $j \in \{1, ..., n\}$.

•Chaque job j_j est d´écrit par n_j tâches ou operations $O_{i,j}$, $i \in \{1, ...n_j\}$

•La durée d'une opération $O_{i,j}$ est $p_{i,j}$

•La machine qui exécute l'opération $O_{i,j}$ du job est notée $M\left(O_{i,j}\right)$ ou $M_{i,j}$.

Les opérations d'un même job ne peuvent pas être exécutées simultanément

• α_2 (voir machines parallèles)

Ateliers à cheminement unique : Flowshop

•Chaque Job est constitué de m opérations et l'ordre de passage sur les différentes machines est le même pour tous les jobs.

$j_j : O_{1,j} \longrightarrow O_{2,j} \longrightarrow, ..., O_{m,j}$ et $M_{i;j} = M_i$

$\alpha_1 = F$

Ateliers à cheminements quelconques : Jobshop

•Le nombre d'opérations n'est pas forcément le même pour tous les jobs

•Chaque job a son propre ordre de passage sur les machines

•$\alpha_1 = j$

Exemple:

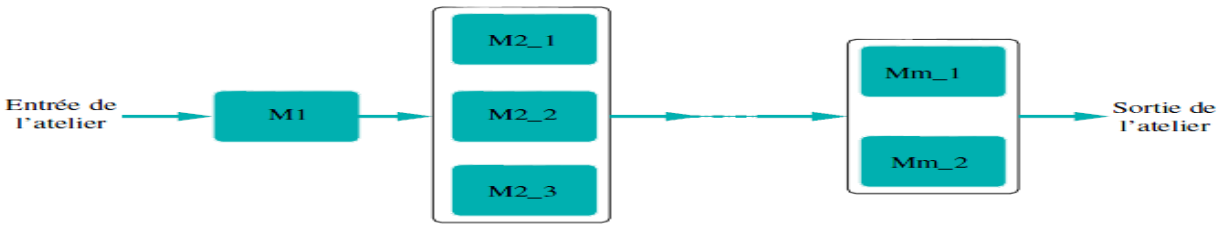
J_j	J_1			J_2			J_3	
$O_{i,j}$	$O_{1,1}$	$O_{2,1}$	$O_{3,1}$	$O_{1,2}$	$O_{2,2}$	$O_{3,2}$	$O_{1,3}$	$O_{2,3}$
$M_{i,j}$	M_1	M_2	M_3	M_2	M_1	M_3	M_3	M_2
$p_{i,j}$	3	2	5	4	2	2	2	3

Ateliers à cheminements libres : Openshop

•Le nombre d'opérations n'est pas forcément le même pour tous les jobs

•L'ordre de passage sur les machines est totalement libre

- $\alpha_1 = O$. [2]



2.3 Méthodes de résolution

2.3.1 Méthodes exactes

Nous présentons d'abord quelques méthodes de la classe des algorithmes complets ou exacts, ces méthodes donnent une garantie de trouver la solution optimale pour une instance de taille finie dans un temps limité et de prouver son optimalité .

La méthode de coupes planes (Cutting-Plane)

La méthode de coupes planes est destinée à résoudre des problèmes d'optimisation combinatoire (POC) qui se formulent sous la forme d'un programme linéaire (PL) : $\min \left\{ c^\top x : Ax \geq b, x \in R^n \right\}$

Dans le cas, où (POC) est de grande taille pour le représenter explicitement en mémoire ou pour qu'il tient dans un solveur de programmation linéaire, on utilise une technique qui consiste à enlever une partie de ces contraintes et de résoudre le problème relaxé (POCR). La solution optimale de (PL) est contenue dans l'ensemble de solutions réalisables de cette relaxation. Pour un problème de minimisation la solution optimale du problème (POCR) est inférieure ou égale à la solution optimale donnée par (POC). Cette méthode consiste à résoudre un problème relaxé, et à ajouter itérativement des contraintes du problème initial. On définit une contrainte pour le problème de minimisation par le couple (s, s_0) où $s \in \mathbb{R}^n$ et $s_0 \in \mathbb{R}$, cette contrainte est dite violée par la solution courante $\overline{x}$ si pour tout $y \in \{x : Ax \geq b\}$ on a

$s^{\intercal}\overline{x} < s_0$ et $s^{\top}y \geq s_0$, on appelle alors ces contraintes des coupes planes. On arrête l'algorithme lorsqu'il n'y a plus de contraintes violées par la solution courante, on obtient ainsi une solution optimale pour le problème initial.

La méthode des coupes planes est peu performante mais sa performance est améliorée lorsqu'elle est combinée avec la méthode "Branch and Bound".

La méthode (Branch and Cut)

La méthode des coupes planes n'est pas toujours efficace face aux problèmes difficiles. De même, bien que l'algorithme du "Branch and Bound" puisse être très performant pour une certaine classe de problèmes, pour cela on utilise la méthode "Branch and Cut" qui combine entre l'algorithme du "Branch and Bound" et de la méthode des coupes planes. Pour une résolution d'un programme linéaire en nombres entiers, la méthode "Branch and Cut" commence d'abord par relaxer le problème puis appliquer la méthode des coupes planes sur la solution trouvée. Si on n'obtient pas une solution entière alors le problème sera divisé en plusieurs sous-problèmes qui seront résolus de la même manière.

On veut résoudre le problème d'optimisation $(min\ c^{\top}x : A\ x \geq b\ ;\ x \in \mathbb{R}^n)$ avec $A \in \mathbb{R}^{mxn}$et $b \in \mathbb{R}^m$

Algorithme Branch and Cut

Liste des problèmes = $\varnothing$;

Initialiser le programme linéaire par le sous problème de contraintes

(A_1, b_1)avec $A_1 \in \mathbb{R}^{m_1 x_n}$et $b_1 \in \mathbb{R}^{m_1}$) avec $m_1 < m$;

Etapes d'évaluation d'un sous problème

Calculer la solution optimale $\overline{x}$ du programme linéaire

$c^{\top}\overline{x} = min(c^{\top}x : A_1 x \geq b1, x \in \mathbb{R}^n)$;

Solution courante = Appliquer la méthode des coupes polyédrales ();

Fin étapes d'évaluation

Si Solution courante est réalisable alors

$x^* = \overline{x}$ est la solution optimale de $min(c^{\top}x : Ax \geq b, x \in \mathbb{R}^n)$;

Sinon

Ajouter le problème dans Liste des sous problèmes;

Fin Si

Tant que Liste des sous problèmes $\neq \varnothing$ faire

Sélectionner un sous problème;

Brancher le problème; Appliquer les étapes d'évaluation ;

Fin Tant que

2.3.2 Méthodes approchés

Heuristiques

une heuristique est un algorithme approché qui permet d'identifier en temps polynomial au moins une solution réalisable rapide, pas obligatoirement optimale. L'usage d'une heuristique est efficace pour calculer une solution approchée d'un problème et ainsi accélérer le processus de résolution exacte. Généralement une heuristique est conçue pour un problème particulier, en s'appuyant sur sa structure propre sans offrir aucune garantit quant à la qualité de la solution calculée.

Les heuristiques peuvent être classées en deux catégories:

– Méthodes constructives qui génèrent des solutions à partir d'une solution initiale en essayant d'en ajouter petit à petit des éléments jusqu'à ce qu'une solution complète soit obtenue.

– Méthodes de fouilles locales qui démarrent avec une solution initialement complète (probablement moins intéressante), et de manière répétitive essaie d'améliorer cette solution en explorant son voisinage.

Méthodes de trajectoire Les méthodes de recherche locale passent d'une solution à une autre dans l'espace des solutions candidates (l'espace de recherche) qu'on note S, jusqu'à ce qu'une solution considérée comme optimale soit trouvée ou que le temps imparti soit dépassé. La méthode de recherche locale la plus élémentaire est la méthode de descente.

Méthode de descente Pour un problème de minimisation d'une fonction f, la méthode descente peut être décrite comme suit:

Algorithme La méthode de descente

Solution initiale s ;

Repeter :

Choisir $s^{'}$ dans un voisinage $N(s)$ de s ;

Si $f(s^{'}) < f(s)$ alors $s = s^{'}$;

jusqu'à ce que $f(s^{'}) \geq f(s), \forall s^{'} \in N(s)$.

Métaheuristiques

Les métaheuristiques ont fait leur apparition. Ces algorithmes sont plus complets et complexes qu'une simple heuristique, et permettent généralement d'obtenir une solution de très bonne qualité pour des problèmes issus des domaines de la recherche opérationnelle ou de l'ingénierie dont on ne connait pas de méthodes efficaces pour les traiter ou bien quand la résolution du problème nécessite un temps élevé ou une grande mémoire de stockage.

Les métaheuristiques peuvent être classées de nombreuses façons. On peut distinguer celles qui travaillent avec une population de solutions de celles qui ne manipulent qu'une seule solution à la fois. Les méthodes qui tentent itérativement d'améliorer une solution sont appelées méthodes de recherche locale ou méthodes de trajectoire. Ces méthodes construisent une trajectoire dans l'espace des solutions en tentant de se diriger vers des solutions optimales. Les exemples les plus connus de ces méthodes sont: La recherche Tabou et le Recuit Simulé. Les algorithmes génétiques, l'optimisation par essaim de particules et Les algorithmes de colonies de fourmis présentent les exemples les plus connus des méthodes qui travaillent avec une population.

Recuit simulé (simulated annealing) Le recuit simulé (SA) est inspiré par l'étude de la stabilité thermique d'un système physique. La méthode part d'une solution initiale admissible et continue l'exploration de l'espace d'états en effectuant des perturbations mineures sur la solution courante. Si la nouvelle solution obtenue est améliorée alors elle est retenue. Si elle est détériorée par rapport au critère d'optimisation

alors elle est retenue avec une probabilité inversement proportionnelle au nombre d'itérations. Le recuit simulé a l'avantage de couvrir un espace de recherche plus grand et d'éviter la convergence prématurée vers un optimum local. Plusieurs problèmes de job shop ont été traités par la méthode de recuit simulé.

Recherche Tabou (Tabu Search) La recherche tabou (TS) est basée sur deux principes. Le premier est d'améliorer à chaque itération la valeur de la fonction objectif en croissant à chaque fois la meilleure solution voisine. Si celle-ci n'existe pas, le choix se fait sur le moins mauvais des voisins. Le deuxième principe consiste à garder en mémoire les dernières solutions choisies et de ne plus les prendre en considération. L'inconvénient de cette méthode est que si la solution atteinte est tout proche de l'optimum global il y a la possibilité de ne pouvoir pas échapper à l'optimum local atteint. La méthode Tabou a été utilisée dans l'ordonnancement des lignes flow shop ou job shop.[4]

Chapitre 3

Méthode PERT

3.1 Méthode PERT

3.1.1 Définition

Program Evaluation and Review Technique (Technique d'organisation et de contrôle des projets).

La méthode PERT est une méthode d'ordonnancement de projets importants à long terme, permettant la coordination optimale des tâches constituant ce projet.

3.1.2 Principe de la méthode

Réduire la durée totale d'un projet par une analyse détaillée des tâches ou activités élémentaires et de leur enchaînement. On étudie les délais sans prendre en compte les charges.

3.1.3 Notion de base

La méthodes s'appuie en grande partie sur une représentation graphique qui permet de bâtir un « réseau PERT ».

Un réseau PERT est constitué par des tâches et des étapes.

Etape

un'étape indique le début et / ou la fin d une tâche.on numérote les étapes afin de clarifier le schéma .

l'étape a des propriétés d'ordre temporel: date au plus tôt et au plus tard, exprimées en minutes, heures, etc.

Exprimée via la méthode PERT, une étape est représentée par un rond, découpé en 3 zones, précisé par son numéro, ainsi que ses dates au plus tôt et au plus tard.

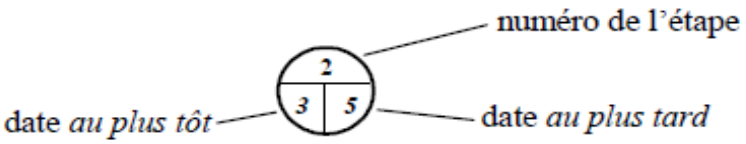


*Figure*1 : *réprésentation d'une étape*

la date au plus tot est le délai minumin, depuis le début du projet, nécissaire pour attaindre l'étape considérée.

la date au plus tard est la date maximun, depuis le début du projet, a laquelle doit étre attainte l'étape considérée pour que le délai de l'ensemble du projet ne soit pas modifié.

Tâche

une tâche est le déroulement dans le temps d'une opération. Contrairement à l'étape, la tâche est pénalisante car elle demande toujours une certaine durée, des moyens (ou ressources) et coûte de l'argent. Elle est symbolisée par un vecteur (ou arc orienté, ou liaison orientée) sur lequel seront indiqués l'action à effectuer et le temps estimé de réalisation de cette tâche.

Exprimée via la méthode PERT, une tâche est représentée par une flèche, précisée par son nom et sa durée.

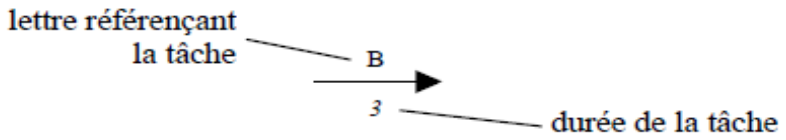


*Figure*2 : *réprésentation d'une tâche*

On distingue trois types des tâches:

•**tâche successive**

•**tâche simultanée:** deux tâches qui commencent en même temps et exécutent en même temps.

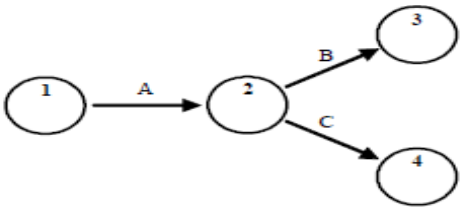


*Figure*3 : *réprésentation d'une tâche simultanée*

les tâches B et C sont similtanées et suivent la tâche A : B et C ne pourront débuter que lorsque A sera complètement achevée.

•**tâche convergente:** deux tâches qui s'exécutent en même temps et s'achèvent en même temps.

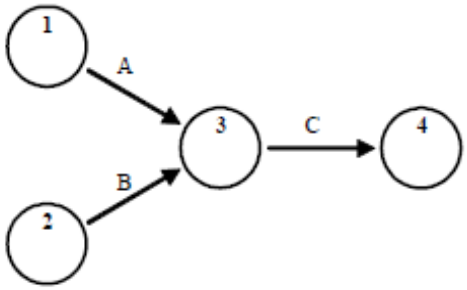


*Figure*4 : *réprésentation d'une tâche convergente*

les tâches A et B sont convergentes et précédent la tâche C : C ne pourra débuter que lorsque A et B seront complètement achevées.

Réseau

Un réseau est l'ensemble des tâches et des étapes forment l'intégralité de la planification du projet (on parle aussi de daigramme PERT).

deux tâches qui se succédent immédaitement dant le temps sont représentées par deux fléches qui se suivent, séparées par une étape.

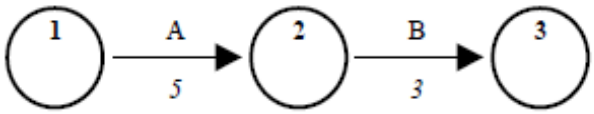


*Figure*5 : *réprésentation d'une réseau*

3.2 Quelques définitions à retenir

- Début au plus tôt d'exécution d'une tâche : C'est le maximum des fins au plus tôt des tâches qui la déclenche (Il peut exceptionnellement y avoir un retard ou

chevauchement si le cahier des charges du projet le précise et que la faisabilité est vérifiée).

- Début au plus tard d'une tâche : C'est la date de fin au plus tard de la tâche moins la durée de la tâche.

- Fin au plus tôt : C'est la date de début au plus tôt plus la durée de la tâche.

- Fin au plus tard : C'est le minimum des dates de début au plus tard des tâches qu'elle enclenche.

- Marge totale : C'est le retard admissible du début d'une tâche qui n'entraîne aucun recul de la date de fin du projet, mais qui consomme les marges libres des opérations suivantes. C'est la date de début au plus tard moins la date de début au plus tôt.

- Marge libre : C'est le retard admissible sur une tâche qui n'entraîne pas de modification des calendriers des tâches suivantes. C'est la date de début au plus tôt de la tâches suivante moins la durée de la tâche moins la date de début au plus tôt de la tâche.

- Chemin critique : C'est l'ensemble des tâches dont la marge totale et la marge libre est nulle. C'est le chemin dont la succession des tâches donne la durée d'exécution la plus longue du projet et fournit le délai d'achèvement le plus court. Si l'on prend du retard sur la réalisation de ces tâches, la durée globale du projet est allongée.

3.3 Etude d'un exemple

3.3.1 Construction d'un réseau PERT

La construction d'un réseau PERT, et son exploitation,supposent d'effectuer les opérations suivantes:

- établir une liste précise des tâches.
- déterminer les tâches antérieures (ainsi que les tâches postérieures éventuellement).
- construire les graphes partiels.
- regrouper les graphes partiels.

•construire le réseau.

Établir une liste précise des tâches

On considère la liste des 12 tâches suivantes, numérotées de A à L. la durée estimée de chaque tâches est indiquée.

tâche	durée
A	3
B	1
C	5
D	6
E	4
F	2
G	9
H	5
I	8
J	2
K	3
L	7

Déterminer les tâches antérieures

L'analyse du projet et de l'ensemble des tâches le constituant nous amène à définir les relations chronologiques d'antériotités des tâches;ces résultats sont regroupés dans le tableau suivant,dans la colonne tâche(s) antérieure(s):

tâche	durée	tâche(s) antérieure(s)
A	3	aucune
B	1	A
C	5	A
D	6	B
E	4	B
F	2	C, I, D
G	9	E, F
H	5	aucune
I	8	H
J	2	H
K	3	I
L	7	J, K

par simple déduction, on en déduit les valeures de la colonne tâche(s) postérieure(s).

tâche(s) antérieure(s)	tâche	tâche(s) postérieure(s)	durée
aucune	A	B, C	3
A	B	D, E	1
A	C	F	5
B	D	F	6
B	E	G	4
C, I, D	F	G	2
E,F	G	aucune	9
aucune	H	I, J	5
H	I	K	8
H	J	L	2
I	K	L	3
J, K	L	aucune	7

Construire les graphes partiels

Les graphes partiels de niveau tâche(s) antérieure(s)/tâche courante sont les suivants:

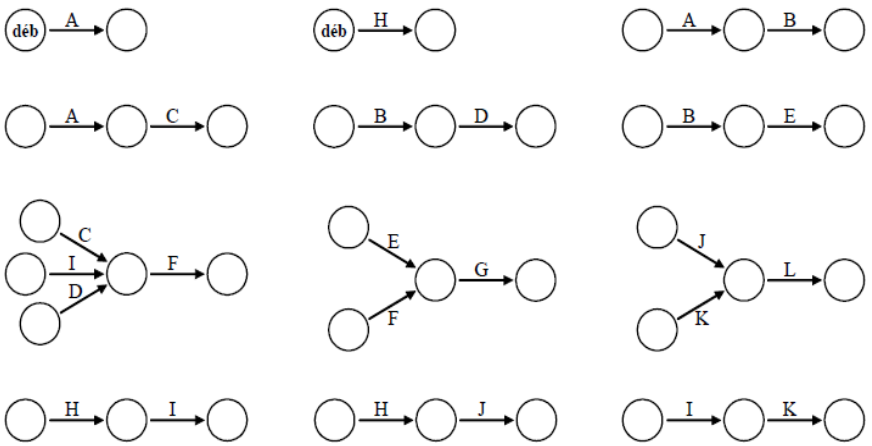


*Figure*6 : *ensemble des graphes partiels de niveau tâche*(s) *antérieure*(s)*/tâche courante*

Les graphes partiels de niveau tâche(s) antérieure(s)/tâche courante/tâche(s) postérieure(s) sont les suivantes:

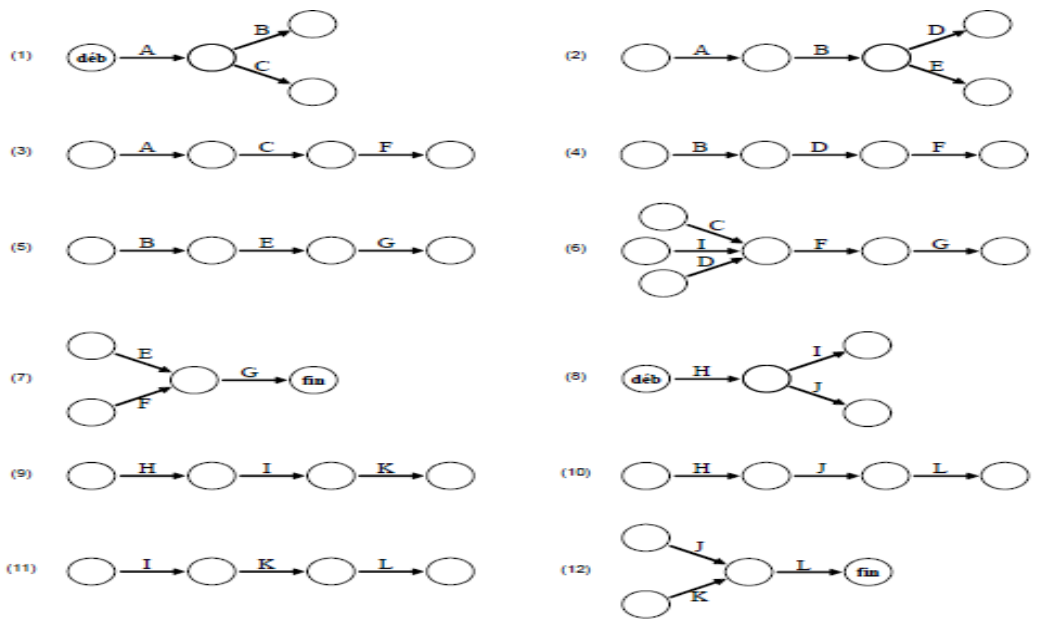


Figure7 : ensemble des graphes partiels de niveau tâche(s) antérieure(s)/tâche courante/tâche(s) postérieure(s)

Regrouper les graphes partiels

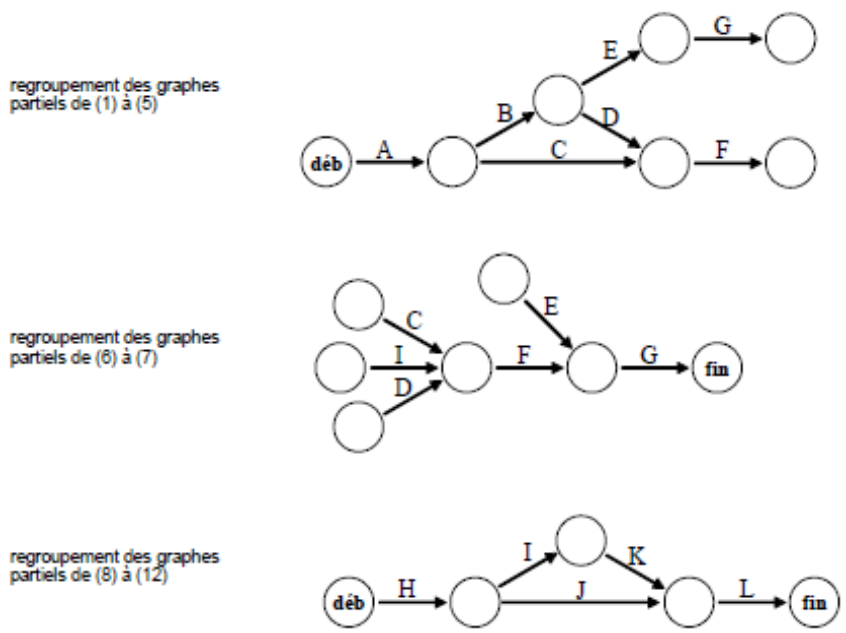


*Figure*8 : *regroupement des graphes partiels*(1)

Construire le réseau

Le regroupement de tous les graphes partiels nous permet d'obtenir le réseau PERT

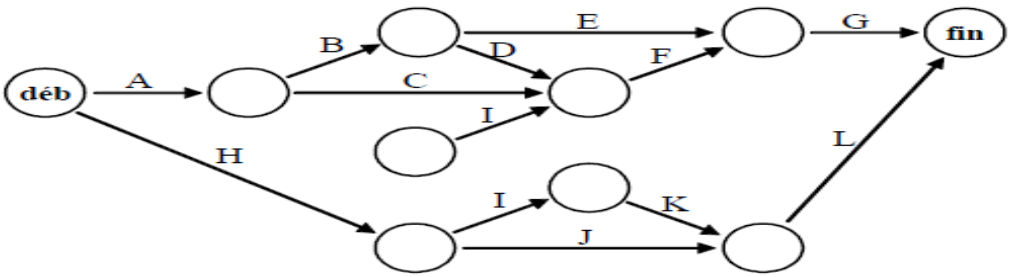


*Figure*9 : *regroupement des graphes partiels*(2)

On peut voir que l'on se retrouve ici confronté à un problème concernant la tâche I. En effet,celle-ci précède les tâches F et K. En revanche la tache K est précédée

uniquement de la tâche I (et pas de C et D) alors que la tâche Fest précédée des tâches I. mais aussi D et C.

Si l'on faisait abstraction de ce problème, on aurait alors une étape, à laquelle C, D et I amènent et de laquelle F et K en repartent. Or ceci est incorrect car cela signifierait alors que K précède aussi C et D en plus de I, pour résoudre ce problème on va donc utiliser une tâche fictive .

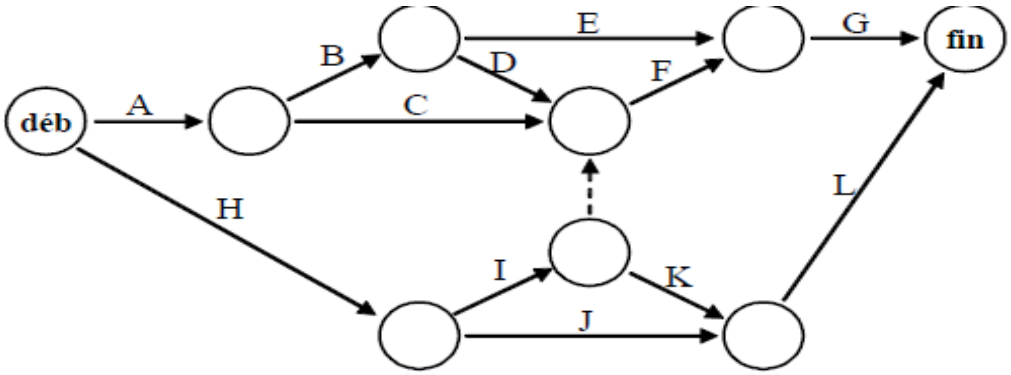


*Figure*10 : *rajout d'une tâche fictive*

la tâche fictive permet de spécifier que F précéde I en plus de Cet D, et par contre que K précéde I uniquement on obtient alors le réseau complet suivant:

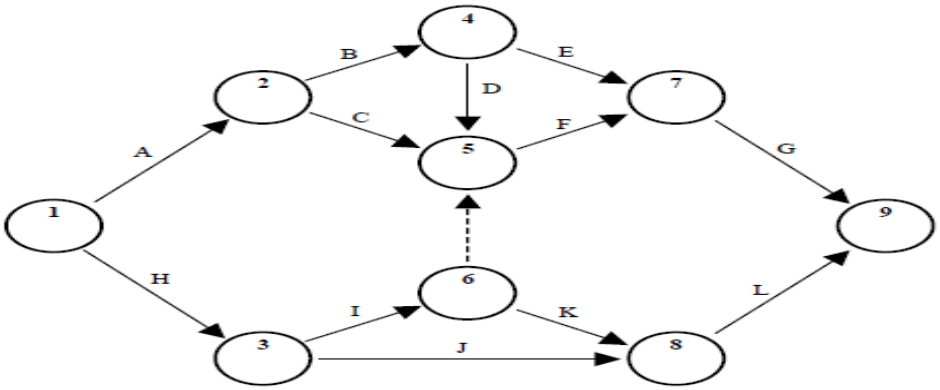


*Figure*11 : *réseau PERT*

3.3.2 Exploitation du réseau PERT

Calcul des dates"au plus tôt"

Deux méthodes de calcule existent alors selon que l'étape considérée est attainte par 1ou par plusieurs tâche:

•1 tâche: il n'ya qu'un seul chemin possible pour atteindre l'étape.

la date au plus tôt vaut la date au plus tôt antérieure à laquelle on rajoute la durée de la tâche liant les 2 étapes:

$to0 = to1 + durée1$

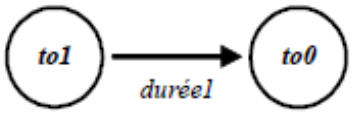


*Figure*12 : *calcul de la date"au plus tôt "dans le cas d'une seule tâche*

•Plusieurs tâches: il y a plusieurs chemins possibles pour atteindre l'étape.

On applique le procéde décrit ci-dessus (pour 1 tâche) pour chacune des tâches antérieures; la date au plus tôt vaut alors le maximum parmi ces résultats:

$to0 = \max(to1 + durée1); (to2 + durée2)$

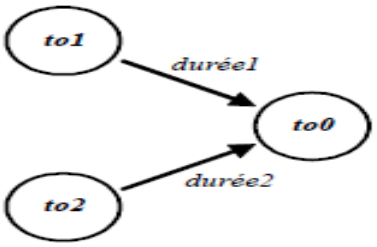


*Figure*13 : *calcul de la date"au plus tôt "dans le cas de plusieurs tâches*

Appliquons ce procéde à notre exemple:

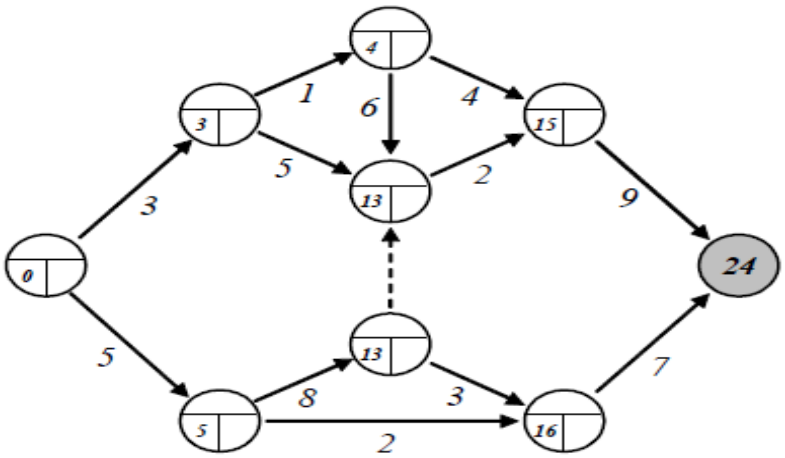


*Figure*14 : *calcul des dates*"*au plus tôt* "

On détermine donc que le projet pourra au mieux être finalisé en l'éspace de 24 jours ouvrés.

Calcul des dates"au plus tard"

Il existe deux méthodes de calcul selon que 1 ou plusieurs tâches partent de l'étape considérée:

•1tâche: il n'ya qu'un seul chemin possible pour partir l'étape.

la date au plus tard vaut la date au plus tard "précédente " (la postérieure dans l'agencement des tâches) à laquelle on retranche la durée de la tâche liant les 2 étapes:

$ta0 = ta1 - durée1$

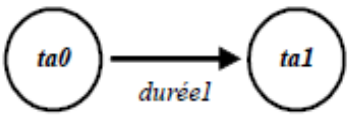


*Figure*15 : *calcul de la date*"*au plus tard* "*dans le cas d'une seule tâche*

•Plusieurs tâches: il y a plusieurs chemins possibles pour partent l'étape.

On applique le procéde décrit ci-dessus (pour 1 tâche) pour chacune des tâches précédentes; la date au plus tard vaut alors le minimum parmi ces résultats:

$to0 = \min(to1 - durée1);\ (to2 - durée2)$

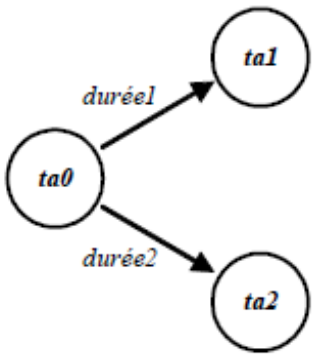


*Figure*16 : *calcul de la date"au plus tard "dans le cas de plusieurs tâches*

Appliquons ce procéde à notre exemple:

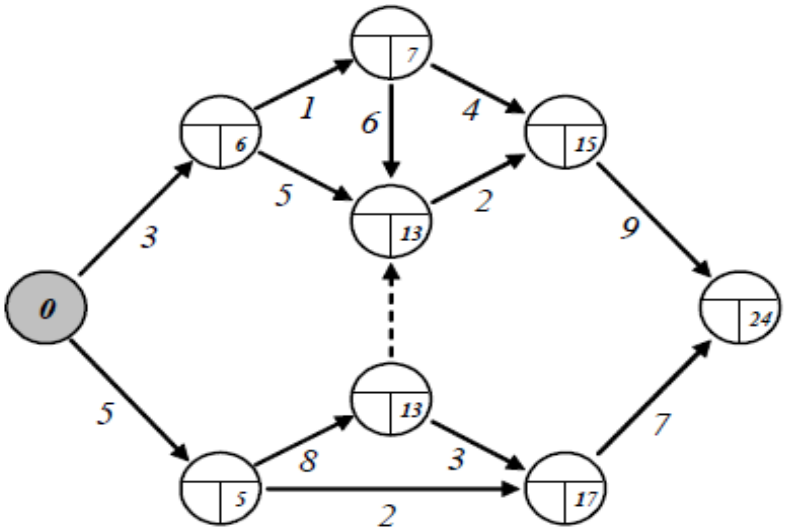


*Figure*17 : *calcul des dates*"*au plus tard* "

On regrouper les informations de date au plus tôt et au plus tard en un seul schema; le réseau PERT obtenu est donc:

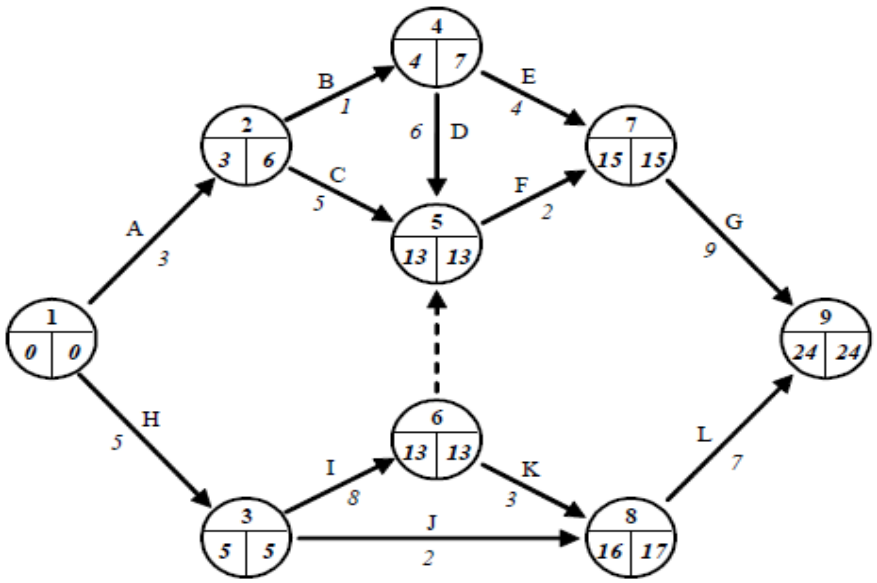


*Figure*18 : *réseau PERT avec des dates "au plus tôt "et "au plus tard "*

Calcul des marges

la marge relative a une tâche se détermine en considérant la valeure des dates au plus tôt et au plus tard des étapes entourant la tâche. la marge vaut alors la date au plus tard de l'étape postérieure a laquelle on retranche la date au plus tôt de l'étape antérieure ainsi que la durée de la tâche elle-même:

$$marge = (ta2 - to1) - durée$$

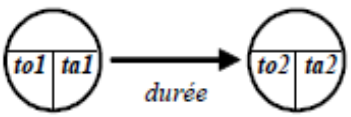


*Figure*19 : *calcul de la marge relative à une tâche*

dans le cadre de notre exemple, on aregroupé les marges calculées dans le tableau suivant:

tâche	marge
A	6-0-3=3
B	7-3-1=3
C	13-3-5=5
D	13-4-6=3
E	15-4-4=7
F	15-13-2=0
G	24-15-9=0
H	5-0-5=0
I	13-5-8=0
J	17-5-2=10
K	17-13-3=1
L	24-16-7=1

les tâches ayant une marge nulle ne bénéficient donc d'oucune latence dans leur exécution leur permettant de ne pas retarder le projet.

l'ensemble de ces tâches permet de déterminer le chemin critique.

Détermination da chemin critique

Pour savoir quel est le chemin critique et donc aussi quelles tâches observer, il suffit de répertoire toutes les tâches ayant une marge nulle. La mise en avant de ces tâches détermine détermine d'elle-même le chemin critique.

les tâches ayant une marge nulle sont : F, G, H et I

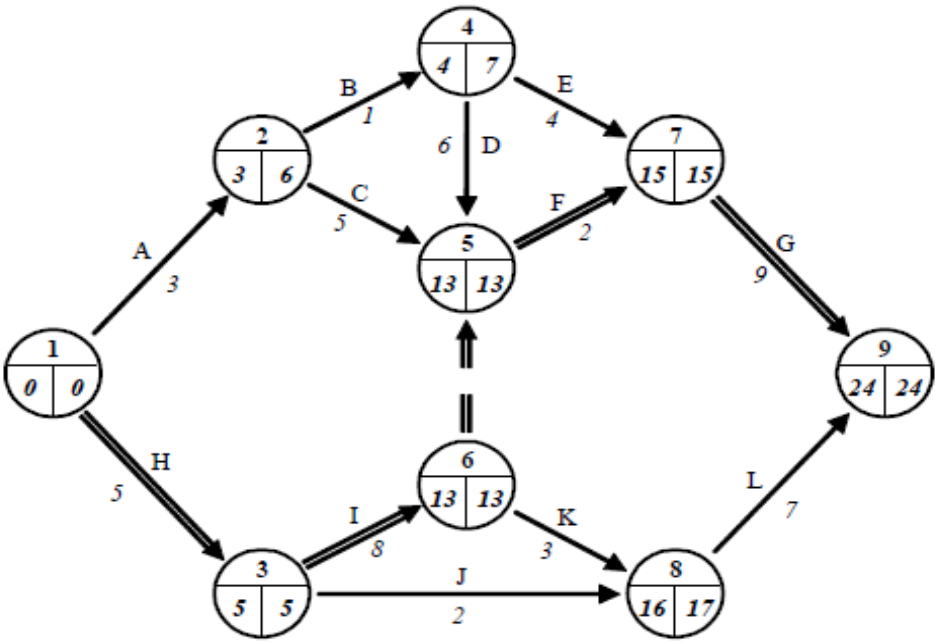


*Figure*20 : *détermination du chemin critique*

Le chemin critique passe donc successivement par les tâches H, I, F et G soit donc les étapes $1, 3, 6, 5, 7$ et 9. [3]

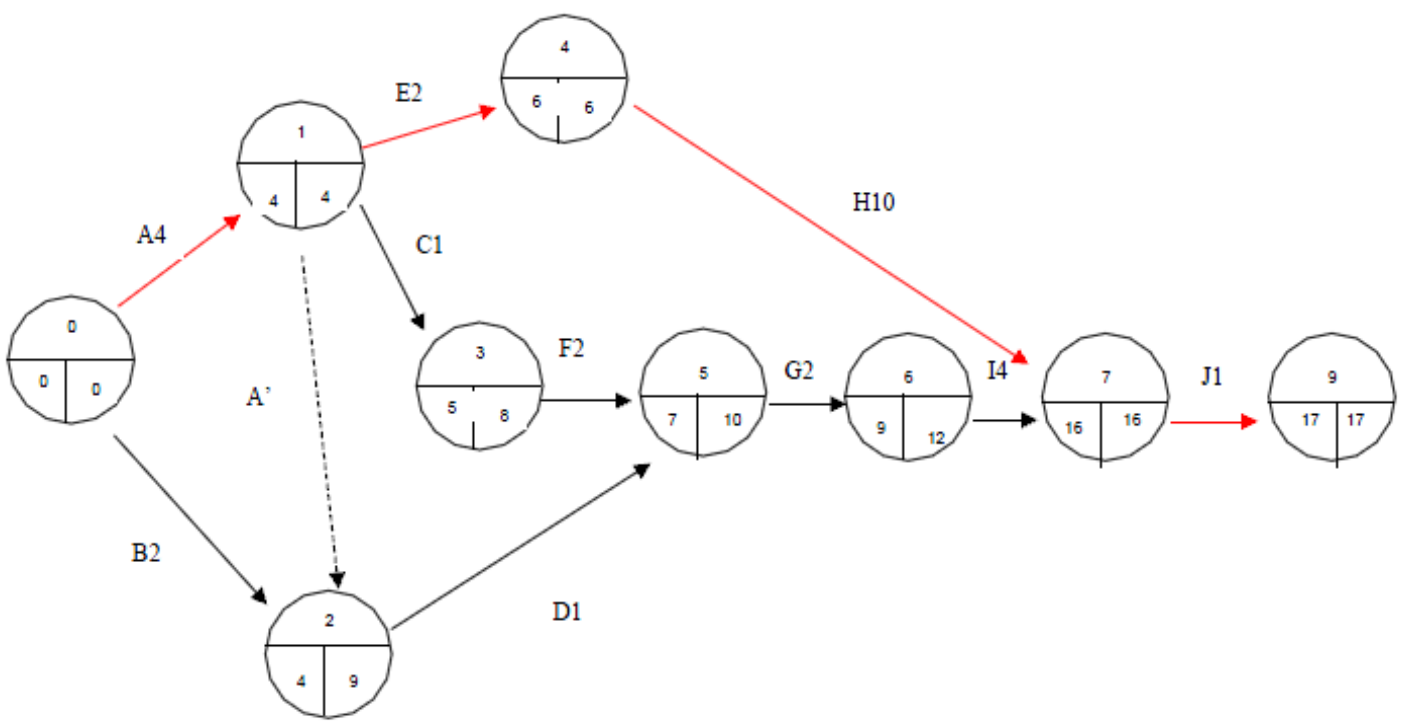
Exemple:

La construction d'un entrepôt est découpée en dix tâches dont les caractéristiques sont données dans le tableau suivant :

tâches	nature	Travaux antérieurs	durée en jours
A	Acceptation des plans par le propriétaire		4
B	Préparation du terrain		2
C	Commande des matériaux	A	1
D	Creusage des fondations	A, B	1
E	Commande des portes et fenêtres	A	2
F	Livraison des matériaux	C	2
I	Coulage des fondations	D, F	2
J	Livraison des portes et fenêtres	E	10
K	Pose des murs, de la charpente et du toit	G	4
L	Mise en place des portes et fenêtres	H, I	1

Solution:

On obtient (le chemin critique est en rouge): $A\ E\ H\ J$.



.

Bibliographie

[1] M.BOUDOUR, Optimisation Multiobjectif de la Synthèse des FACTS par les Particules en Essaim pour le Contrôle de la Stabilité de Tension des Réseaux Electriques, Mémoire de Magister en Electrotechnique, Université Amar Telidji, Laghouat 2007.

[2] SARI Zaki, Planification et Ordonnancement en temps réel d'un Job shop en utilisant l'Intelligence Artificielle, Mémoire de Magister en Automatique, Université de Tlemcen, 2012.

[3] Info.arqendra.net/dowload.php?Filename=FILES%2F_PERT_cours.pdf.

[4] Www.fsr.um5.ac.ma/cours/maths/bernoussi/cours%20c2SI.pdf.