

CHAPITRE 3

Le problème du voyageur de commerce

1. Historique :

Les premières approches mathématiques exposées pour le problème du voyageur de commerce ont été traitées aux 19 siècles par les mathématiciens Sir (William Rowan Hamilton et Thomas Penyngton Kirkman). Hamilton en a fait un jeu : " Hamiltons Icosian game " . Les joueurs devaient réaliser une tournée passant par 20 points en utilisant uniquement les connections prédéfinies. En 1930, ce problème est traité plus en profondeur par (Karl Menger) à Harvard. Il est ensuite développé à Princeton par les mathématiciens (Hassler Whitney et Merrill Flood), une attention particulière est portée sur les connections par (Menger et Whitney) ainsi que sur la croissance de ce problème. En 1954, des solutions de ce problème pour 49 villes ont été obtenues par (Dantzig et al, 1954) par une méthode de coupe. En 1974, des Solutions pour 100 villes par (Camerini et al). En 1987, le problème à été résolu pour 532 et 2392 villes par (Padberg et Rinaldi). En 1998, des solutions ont été trouvées pour 13 509 villes des Etats-Unis. En 2001, (Applegate et al) des universités de Rice et Princeton ont résolu le problème pour les 15 112 villes d'Allemagne. En mai 2004, le problème du voyageur de commerce qui consiste à visiter chacune des 24.978 villes en Suède a été résolu : Une excursion de longueur approximativement de 72.500 kilomètres a été trouvée et on a montrée l'inexistence d'une tournée plus courte. En mars 2005, un problème de taille égale 33.810 villes a été résolu (Cook et al. 2006).

2. Le problème du voyageur de commerce (PVC) :

Le problème du voyageur de commerce (" Traveling Salesman Problem ", TSP) naît d'une problématique vécue par des vendeurs ou commerciaux qui déplacent pour livrer ou rencontrer leurs clients. C'est l'un des problèmes le plus étudié dans l'optimisation combinatoires qui à la recherche d'un trajet minimal permettant à un voyageur de visiter n villes séparées par distances données en passant par chaque ville exactement une fois. Il commence par une ville quelconque et termine en retournant à la ville de départ. Quel chemin faut-il choisir afin de minimiser la distance parcourue ? Etant donné un graphe $G = (V, E)$ non orienté simple et sans boucles, le

problème du voyageur de commerce consiste à déterminer un cycle Hamiltonien (qu'on appellera tournée) de longueur minimale.

2.1 Formulation mathématique du problème :

Le PVC peut se modéliser comme (BIP). En effet, étant donnée n villes, notons $C = (C_{ij})$ la matrice des coûts et x_{ij} les variables de décision définies par

$$x_{ij} = \begin{cases} 1 & \text{Si le voyageur va immédiatement de la ville } i \text{ vers la ville } j, \\ 0 & \text{sinon ;} \end{cases}$$

La formulation mathématique est $\text{Min } \sum_{i=1}^n \sum_{j=1}^n C_{ij} x_{ij}$ (2.2)

$$\sum_{j=1}^n x_{ij} = 1 \text{ pour } i = 1, \dots, n \quad (2.3)$$

$$\sum_{i=1}^n x_{ij} = 1 \text{ pour } j = 1, \dots, n \quad (2.4)$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \forall S, \quad 2 \leq |S| \leq n - 1 \quad (2.5)$$

$$x_{ij} \in \{0, 1\} \text{ pour } i, j = 1, \dots, n \quad i \neq j \quad (2.6)$$

Dans cette formulation, la relation (2.2) décrit la fonction objectif. Les contraintes (2.3) et (2.4) s'appellent les contraintes de degré qui assurent qu'une ville visitée qu'une seule fois : on y arrive une et une seule fois (2.3), on en part une et une seule fois (2.4), ces contraintes ne sont pas suffisantes pour décrire les tours, d'où la nécessité d'introduire les contraintes (2.5) appelées contraintes d'élimination des sous-tours, avec S un sous ensemble de V et S son complémentaire dans V , $|S|$ est le cardinal de S . Enfin, les contraintes (2.6) sont les contraintes d'intégrité de variables. Une autre formulation des contraintes (2.5) peut être donnée par la relation suivante :

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1, \forall S, \quad 2 \leq |S| \leq n - 1 \quad (2.7)$$

Cette formulation de PVC contient $n(n-1)$ variables binaires, $2n$ contraintes de degré et $2^n - 2n - 2$ contraintes d'élimination de sous tours.

2.2 Définition du PVC :

Un voyageur de commerce (PVC) doit visiter n villes données en Assante par chaque ville exactement une fois. Il commence par une ville quelconque et termine en retournant à la ville

de départ. Les distances entre les villes sont connues. IL faut trouver le chemin qui minimise la distance parcourue. La notion de distance peut-être remplacée par d'autres comme le temps qu'il met ou l'argent qu'il dépense [14].

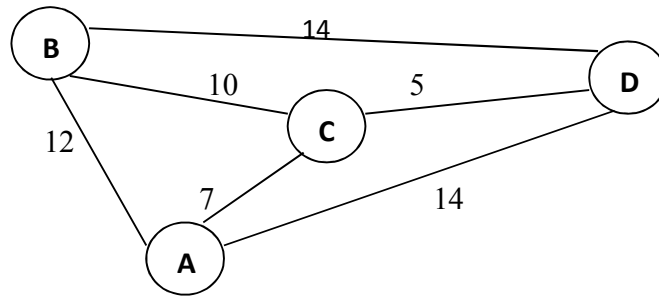


Figure.3.1 Le problème de voyageur de commerce.

2.3 Modélisation du PVC :

Le problème du voyageur de commerce peut être modélisé à l'aide d'un graphe constitué d'un ensemble de sommets et d'un ensemble d'arêtes. Chaque sommet représente une ville, une arête symbolise le passage d'une ville à une autre, et on lui associe un poids pouvant représenter une distance, un temps de parcours ou encore un coût. La figure 3.1 montre un exemple de graphe à 4 sommets. Résoudre le problème du voyageur de commerce revient à trouver dans ce graphe un cycle passant par tous les sommets une et une seule fois (un tel cycle est dit « Hamiltonien ») et qui soit de longueur minimale. Pour le graphe ci-contre, une solution à ce problème serait le cycle : 1, 2, 3, 4 et 1, correspondant à une distance totale de 23. Cette solution est optimale, il n'en existe pas de meilleure [15].

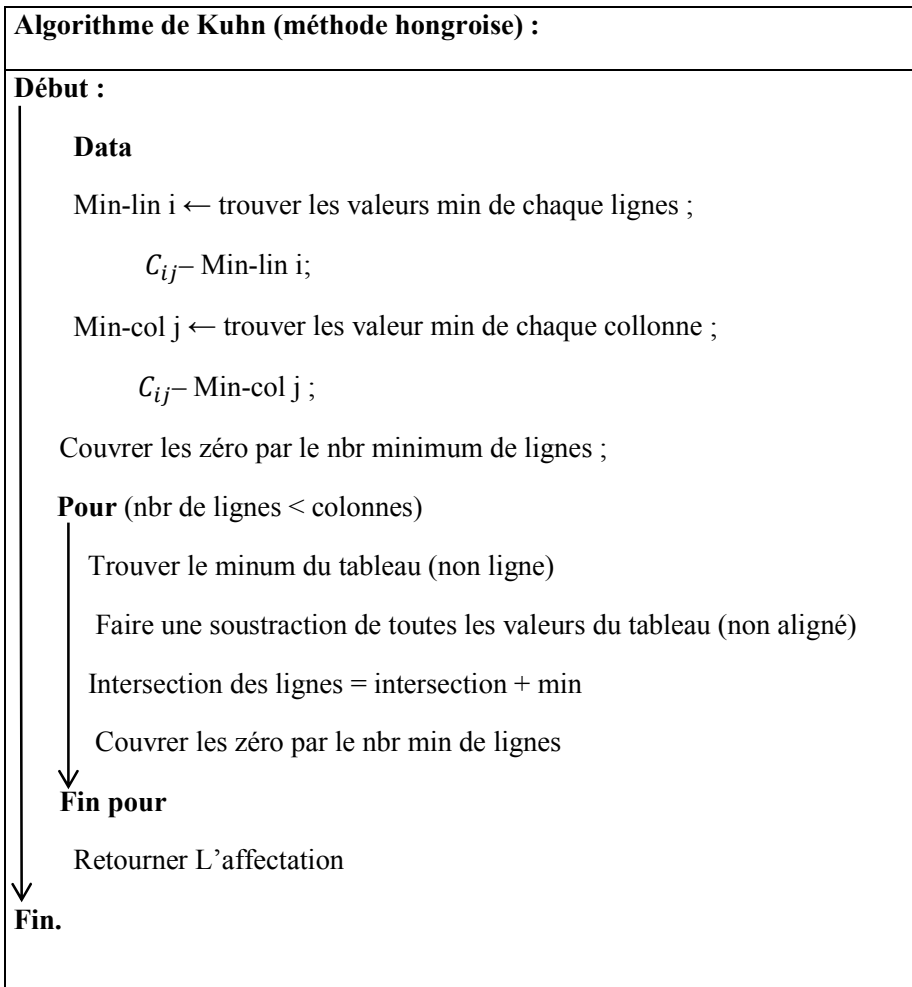
3. L'évaluation par défaut :

Soit un problème de voyageur de commerce (PVC) à n villes. On considère le problème d'affectation de n agents et n tâches. Bien sûr une solution optimale de ce problème d'affectation qui donne dans les méthodes cas un seul cycle élémentaire ((En générale plusieurs cycle élémentaire)) dont la somme des coûts des arcs est minimum. Il est clair que si on relâche la contrainte de l'élémentarité du cycle cherché du problème voyageur de commerce. On obtient un problème d'affectation, par conséquent la solution optimale de ce dernier constitue une d'évaluation par défaut sur la solution optimale du PVC. Ci-bas on rappelle une méthode dite hongroise servant à déterminer une solution optimale pour le problème d'affectation.

Ceci sera utilisé dans une méthode de séparation et évaluation (B&B) pour déterminer une solution optimale pour le PVC.

3.1 La méthode Hongroise (kuhn) :

La "méthode Hongroise", proposée par *Kuhn* en 1955, est un algorithme dual qui s'appuie sur une modélisation du problème d'affectation sous forme d'un programme linéaire, mais qui peut être vu comme une variante de l'algorithme de (Busaker et Gowen), spécialisée pour la structure biparti du graphe. Du fait de sa grande efficacité sur ce type de problème, c'est l'algorithme de référence en Recherche Opérationnelle pour résoudre le problème d'affectation. Son principe est basé sur le fait que les couplages de poids minimal dans le graphe du problème primal sont exactement les couplages de cardinalité maximale dans le graphe du problème dual (voir par exemple ou pour une présentation détaillée de la méthode). [2]



Algorithme 3. 1. Pseudo-code de méthode Hongroise.

Principe :

Cet algorithme, fondé sur le théorème de Konig, a pour but de résoudre le problème d'affectation. Tout problème de ce genre peut, en général, se résumer sous la forme d'un tableau de couts, dans lequel il s'agit de choisir un élément et un seul par ligne et par colonne, de manière à obtenir la somme minimale. La technique hongroise se divise en cinq phases.

➤ Phase1 :- obtention des zéros.

A tous les éléments d'une même colonne on enlève le plus petit élément de la colonne ; on fait de même pour les lignes .

➤ Phase2 :- Recherche d'une solution optimale.

Avec le tableau obtenu, on cherche à former une solution pour laquelle le cout total ait une valeur nulle : elle ne doit contenir que des zéros.

Si cela est possible, on a trouvé une solution optimale sinon on passe à la Phase 3.

D_{ij}

-	10	8	9	7
10	-	10	5	6
8	10	-	8	9
9	5	8	-	6
7	6	9	6	-

Telle que $(1 \leq i, j \leq 5)$

Matrice des distances

-	3	1	2	0
5	-	5	0	1
0	2	-	0	1
4	0	3	-	1
1	0	3	1	-

On cherche d'abord la ligne ou une des lignes comptant le moins de zéros ;

On encadre un des zéros de cette ligne, puis on barre les zéros qui se trouvent sur la même ligne ou dans la même colonne que le zéro encadré.

Parmi les lignes restantes, on cherche alors celle ou l'une de celles qui contient le moins de zéros et on répète le même processus. on procède ainsi jusqu'à ce qu'on ne puisse plus encadrer de zéros

-	3	⓪	2	⊘	
5	-	4	⓪	1	X
⓪	2	-	⊘	1	X
4	⓪	2	-	1	X
1	⊘	2	⊘	-	
	X		X		

➤ Phase 3 :- Recherche des lignes et colonnes en nombre minimal contenant tous les zéros.

On opère pas à pas comme suit :

- a) on marque d'une croix (x) toutes les lignes qui ne contiennent aucun zéro encadré ;
- b) on marque toute colonne qui a un zéro barré sur une ou plusieurs lignes marquées ;
- c) on marque toute ligne qui a un zéro encadré dans une colonne marquée ;
- d) on répète b) et c) jusqu'à ce qu'il n'y ait plus de colonne ou de ligne à marquer.

On trace alors un trait sur toute ligne non marquée et un trait sur toute colonne marquée.

On obtient ainsi les lignes et colonnes en nombre minimal qui contiennent tous les zéros encadrés ou barrés.

-	3	0	2	0	
5	-	4	0	1	X
0	2	-	X	1	X
4	0	2	-	1	X
1	X	2	X	-	
	X		X		

➤ Phase 4 :- Déplacement de certains zéros.

On prend le plus petit nombre du tableau partiel dont les éléments ne sont traversés par aucun trait. On enlève ce nombre aux éléments des colonnes non traversées par un trait et on l'ajoute à ceux des lignes traversées par un trait.

-	4	0	3	0
4	-	3	0	0
0	3	-	1	1
3	0	1	-	0
0	0	1	0	-

➤ Phase 5 :- Obtention de la solution optimale ou départ pour un nouveau cycle.

Sur le nouveau tableau obtenu en phase 4, on recherche une solution optimale selon la méthode de la phase 2. Si on n'aboutit pas à une solution optimale, on continue les opérations 3 et 4 et ainsi de suite.

-	4	⓪	3	∅
4	-	3	⓪	∅
⓪	3	-	1	1
3	∅	1	-	⓪
∅	⓪	1	∅	-

La solution : $X_{13}=X_{24}=X_{31}=X_{45}=X_{52}=1$

$Z = 8+5+8+6+6=33= LB$

La solution optimale peut évidemment ne pas être unique.

➤ Remarque 1 :

Si on a à chercher une affectation de somme maximale, il suffira de considérer le tableau :

$$[d'_{ij}]=[D]-[d_{ij}],$$

D étant un majorant de tous les d_{ij} et de chercher l'affectation de somme minimale précédemment décrite pour le tableau $[d'_{ij}]$ ainsi formé. [13].

➤ Remarque 2 :

On utilisant la méthode Hongroise pour déterminer lower bound (LB) c'est le dernier utilisera pour déterminer une évaluation par défaut pour la fonction objectif du problème du voyageur de commerce pour lequel on l'utilise pour construire une méthode de séparation et évaluation (branch and bound) pour déterminer une solution exacte .

4. Méthode de solution exacte (Branch and Bound) :

La méthode par séparation et évaluation (Branch and Bound), Les méthodes par séparation et évaluation sont le moyen générique le plus utilisé pour la résolution exacte des problèmes d'optimisation combinatoire et en particulier pour la résolution des (ILP) qui pratiquent une énumération intelligente de l'espace des solutions.

1. Algorithm B&B :
<p>▪ Dada</p> <p>Boucles $\leftarrow M$</p> <p>Lb $\leftarrow kohn$</p> <p>Ub $\leftarrow mindis$</p> <p>Branch (put M on possible bugs)</p> <p>Kohn</p> <p>Goto (if solution \leq ub)</p> <p>Return Tour ;</p> <p>Roturn upper bound ;</p>

Algorithme 3. 2. Pseudo-code de B&B.

4.1 L'évaluation par défaut (LB):

On Applique la méthode Hongroise pour déterminer une solution lower bound (LB), sera utiliser pour construire la méthode séparation et évaluation (B&B) sur le problème du voyageur de commerce pour déterminer une solution exacte, de 5 villes dont la matrice des distances entre villes est donnée ci-dessous :

Le problème d'affectation est un problème du voyageur de commerce relâché alors :

La solution : $X_{13}=X_{24}=X_{31}=X_{45}=X_{52}=1$

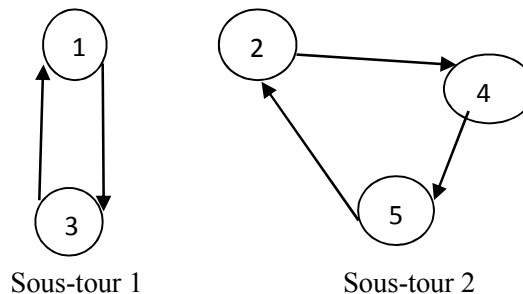
$Z = 8+5+8+6+6=33=LB$ qui correspond an nœud racine de l'Arber de la méthode de B&B

Enfin nous avons deux sous tours :

4.2 Séparation :

Sous-tour 1 : $X_{13}=X_{31}=1$

Sous-tour 2 : $X_{24}=X_{45}=X_{52}=1$



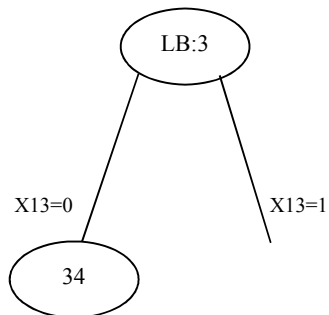
Chapitre 3 - Le problème du voyageur de commerce

La minimum valeur de matrice de sous tour :

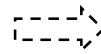
	1-3-1	2-4-5-2
1-3-1	-	0
2-4-5-2	0	-

Etape 1:


On pose big M sur l'arc X13 :



-	10	M	9	7
10	-	10	5	6
8	10	-	8	9
9	5	8	-	6
7	6	9	6	-



-	3	M	2	0
5	-	5	0	1
0	2	-	0	1
4	0	3	-	1
1	0	3	0	-



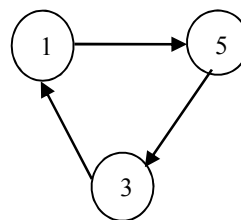
-	3	M	2	⓪
5	-	2	⓪	1
⓪	2	-	8	1
4	⓪	10	-	1
1	10	⓪	6	-

La solution est : $X_{15}=X_{24}=X_{13}=X_{42}=X_{53}$

$$Z=7+5+8+5+9=34$$

$$X_{15}=X_{53}=X_{31}=1$$

$$X_{24}=X_{42}=1$$



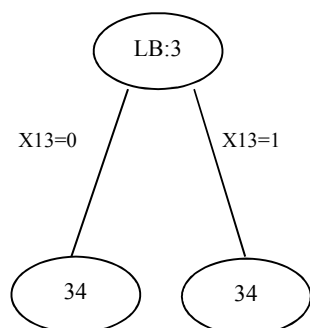
Sous-tour 1



Sous-tour 2

Etape 2 :

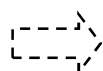
On pose big M sur l'arc X 31 :



-	10	8	9	7
10	-	10	5	6
M	10	-	8	9
9	5	8	-	6
7	6	9	6	-



-	3	1	2	0
5	-	5	0	1
M	2	-	0	1
4	0	3	-	1
1	0	3	0	-



-	3	0	2	X
4	-	4	0	1
M	2	-	X	1
3	0	2	-	1
0	X	2	X	-



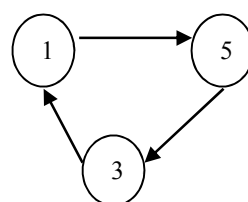
-	4	0	3	X
3	-	3	0	X
M	2	-	X	0
2	0	1	-	X
0	1	2	1	-

La solution est : $X_{13}=X_{24}=X_{35}=X_{42}=X_{51}$.

$$Z=8+5+9+5+7=34$$

Sous-tour 1 : $X_{13}=X_{35}=X_{51}=1$

Sous-tour 2 : $X_{24}=X_{42}=1$



Sous-tour 1

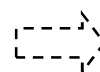


Sous-tour 2

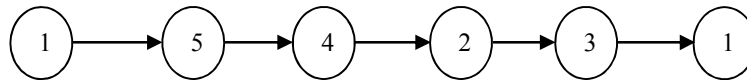
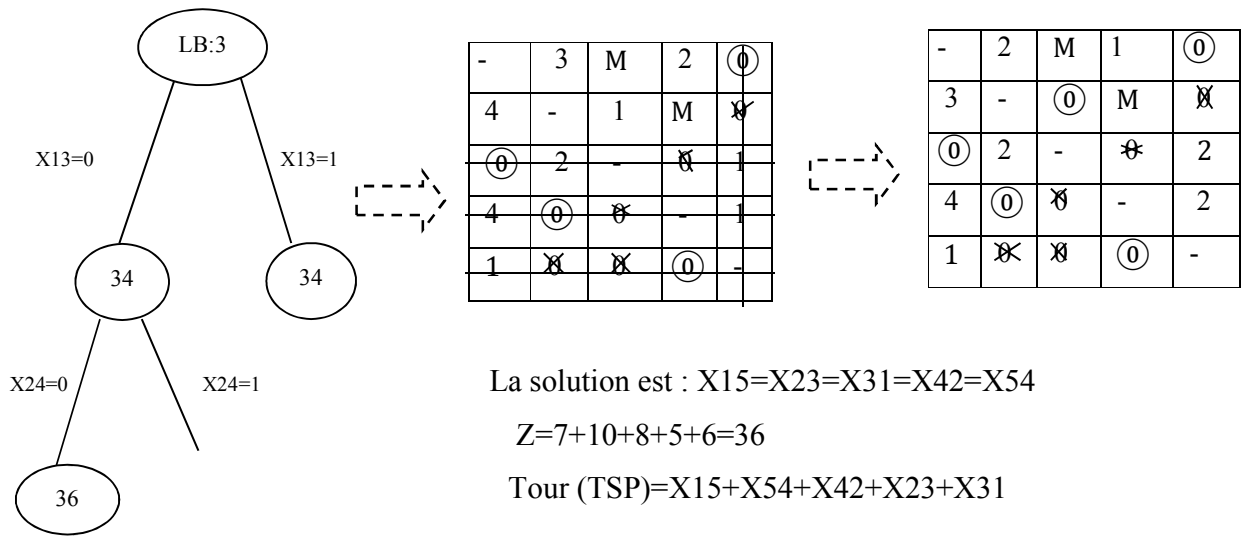
Etape 3 :

On pose big M sur l'arc X 24 de plus X13:

-	10	M	9	7
10	-	10	M	6
8	10	-	8	9
9	5	8	-	6
7	6	9	6	-

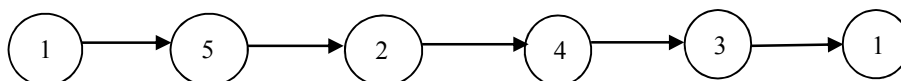
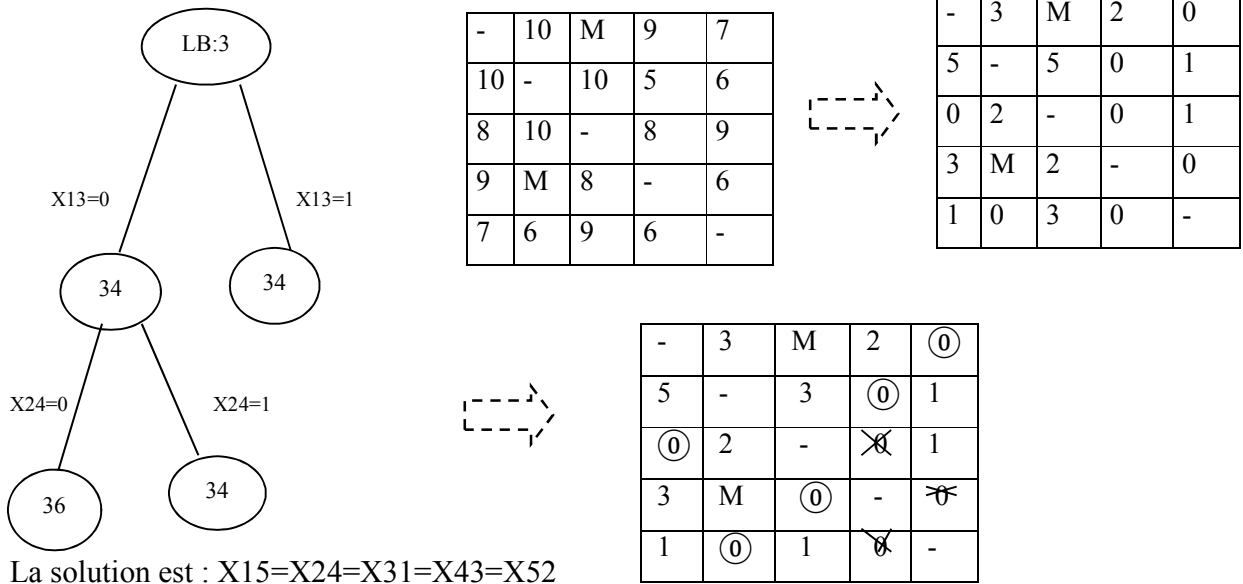


-	3	M	2	0
4	-	4	M	0
0	2	-	0	1
4	0	3	-	1
1	0	3	0	-



Etape 4 :

On pose big M sur l'arc X_{42} de plus X_{13}



5. Méthodes approchées (Métaheuristique):

Ici on considère uniquement les métaheuristiques à base de solution unique :

5.1 RS: permet d'accepter une solution de piètre qualité que la solution courante afin de diversifier la recherche et échapper au piège de l'optimum local. Le fait d'accepter des solutions de mauvaises qualités peut mener la recherche vers la meilleure solution (l'optimum global) car cette dernière peut faire partie du voisinage d'une mauvaise solution et non pas d'une bonne solution (i.e. la solution courante qui est de meilleure qualité que sa voisine) qui peut représenter un optimum local. Cependant, l'acceptation de solutions de mauvaises qualités peut causer une perte de la meilleure solution rencontrée au cours de la recherche et entraîne une convergence vers une solution de mauvaise qualité qu'une autre déjà trouvée. Problème peut être facilement résolu en ajoutant une variable permettant la mémorisation de la meilleure solution trouvée.

Algorithme : méthode de Recuit Simulé

Initialisation

- Choisir T_0, T_f, N_{max}
- Générer aléatoirement une solution admissible initiale s_0
- Calculer $f_0 = f(s_0)$, $s^* \leftarrow s_0$, $f^* \leftarrow f_0$
- $i \leftarrow 1$

Etape 1

- Choisir $s_1 \in \text{Rand}(V(s_0))$
- Calculer $\Delta f = f(s_1) - f(s_0)$
- Si $\Delta f \leq 0$ aller en Etape 3
- Sinon aller en Etape 2

Etape 2

- Choisir une valeur aléatoire $\theta \in [0,1]$ et calculer $p(\Delta f) = e^{-\Delta f/T}$
- Si $\theta < p(\Delta f)$ aller en Etape 3
- Sinon rejeter cette solution et aller en Etape 4

Etape 3

- Accepter la solution
- Poser $s_0 \leftarrow s_1$, $f_0 \leftarrow f_1$
- Aller en Etape 4

Etape 4

- Si $i > N_{max}$ stop
- Sinon diminuer la température en utilisant $T_{i+1} = \gamma * T_i$
- Poser $i \leftarrow i + 1$ aller en Etape 1

Algorithme 3. 3. Pseudo-code de Recuit Simulé

Remarquons que le choix de la fonction de température est déterminant.

Le recuit simulé est un algorithme basé sur la recherche à voisinage (recherche locale). C'est un algorithme simple, facile à implémenter et à adapter à un grand nombre de problèmes : traitement d'image, sac à dos, voyageur de commerce, ordonnancement, etc. Comparé avec la recherche locale simple, le RS permet de se sauver du piège de l'optimum local et d'offrir des solutions de bonne qualité comme il présente une souplesse d'intégration des contraintes liées au problème traité. En revanche, cet algorithme dispose d'un nombre important de paramètres (température initiale, paramètres liés à la fonction d'ajustement de la température... etc.) à ajuster. En outre, le RS est un algorithme lent surtout avec les problèmes de grande taille.

5.2 RT : est une métaheuristique à base d'une solution unique. Elle a été proposée en 1986 par Glover. RT est une méthode de recherche locale avancée, elle fait appel à un ensemble de règles et de mécanismes généraux pour guider la recherche de manière intelligente. L'optimisation de la solution avec la recherche Tabou se base sur deux astuces: l'utilisation de la notion du voisinage et l'utilisation d'une mémoire permettant le guidage intelligent du processus de la recherche. En parcourant le voisinage de la solution courante s , la recherche Tabou ne s'arrête pas au premier optimum local rencontré. Elle examine un échantillonnage de solution du voisinage de s et retient toujours la meilleure solution voisine s' , même si celle-ci est de piètre qualité que la solution courante s , afin d'échapper de la vallée de l'optimum local et donner au processus de la recherche d'autres possibilités d'exploration de l'espace de recherche afin de rencontrer l'optimum global. En fait, les solutions de mauvaise qualité peuvent avoir de bons voisinages et donc guider la recherche vers de meilleures solutions. Cependant, cette stratégie peut créer un phénomène de cyclage (i.e. on peut revisiter des solutions déjà parcourues plusieurs fois). Afin de pallier à ce problème, la recherche Tabou propose l'utilisation d'une mémoire permettant le stockage des dernières solutions rencontrées pour ne pas les visiter dans les prochaines itérations et tomber dans le problème du cyclage répétitif. Cette mémoire est appelée « la liste Tabou », d'où le nom de la métaheuristique Tabou. La taille de la liste Tabou est limitée, ce qui empêche l'enregistrement de toutes les solutions rencontrées. C'est la raison pour laquelle la liste Tabou procède comme une pile FIFO, où la plus ancienne solution sera écartée pour laisser place à la dernière solution rencontrée. Le but de faciliter la gestion de la liste tabou en termes de temps de calcul ou d'espace mémoire nécessaires à pousser les chercheurs à proposer de ne conserver dans la liste que des attributs (i.e. caractéristiques) des solutions au lieu des solutions complètes.

Particulièrement, dans le cas où le nombre de variables est très élevé. Toutefois, l'utilisation de la liste Tabou peut mener à un blocage dans certains cas. En fait, les nouvelles solutions qui possèdent des attributs des solutions tabou, seront considérées tabou aussi même si elles ne sont pas encore été visitées. Pour remédier à ce problème, on a défini une technique appelée « critère d'aspiration » permettant de sortir du blocage causé par la ressemblance des attributs des solutions tabou. Le critère d'aspiration permet d'omettre le statut tabou sur une solution lorsque certaines circonstances sont respectées. Un critère d'aspiration très classique consiste à autoriser une solution Tabou si celle-ci est l'une des meilleures solutions rencontrées depuis le démarrage de la recherche. La recherche tabou a été largement utilisée pour la résolution de problèmes d'optimisation difficiles comme: le problème du voyageur de commerce, les problèmes du sac à dos, les problèmes de routage, d'ordonnancement, de coloration de graphes, d'exploitation géologique, etc. C'est une méthode facile à mettre en œuvre, rapide, donne souvent de bons résultats et permet de se sauver du premier optimum local rencontré contrairement à la méthode de la recherche locale simple. En revanche, la liste Tabou demande de ressources importantes si elle est de grande taille. En fait, elle demande une gestion soigneuse de sa taille et de ce qu'elle doit contenir comme informations sur les solutions trouvées, de manière à ne pas être très exigeante en temps de parcours et d'espace mémoire requis et aussi pour éviter les confusions causées par la ressemblance des attributs des solutions qui bloquent la Recherche.

Algorithme : méthode Tabou
1 : Initialisation s_0 une solution initiale $s \leftarrow s_0, s^* \leftarrow s_0, f^* \leftarrow f(s_0)$ $T = \emptyset$ 2 : Générer un sous-ensemble de solution au voisinage de s $s' \in V(s)$ tel que $\forall x \in V(s), f(x) \geq f(s')$ et $s' \notin T$ Si $f(s') < f^*$ alors $s^* \leftarrow s'$ et $f^* \leftarrow f(s')$ Mise-à-jour de T 3 : Si la condition d'arrêt n'est pas satisfaite retour à l'étape

Algorithme 3.4. Pseudo-code de Recherche Tabou.

6 .Conclusion :

On a considéré dans ce chapitre l'un des fameux problèmes d'optimisation combinatoire c'est le problème de voyageur de commerce. On a montré comment on utilise une méthode exacte de problème d'affectation pour déterminer une évaluation par défaut puis on a incorporé dans la méthode de séparation et évaluation (B&B) pour déterminer une solution optimale pour le problème considéré. Et puisque les méthodes exactes comme celle de (B&B) échouent pour les problèmes des grandes tailles on a rappelé les métaheuristique à base solution unique pour déterminer une solution approchée.