République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene

Faculté de Mathématiques

Département de Recherche Opérationnelle

Rapport

Licence Recherche Opérationnelle

Thème

Problème de transversal minimum dans les graphes bipartis

Présenté par : LAIB omar
ACHACHI el hadj ali

Encadré par : Hissoum Anissa

Table des matières :

Introduction Générale

La **Recherche Opérationnelle** (**RO**) est une discipline exploitant ce qu'il y a de plus opérationnel dans les mathématiques, l'économie et l'informatique. Elle est en prise directe avec l'industrie et joue un rôle-clé dans la compétitivité.

La **RO** est une approche quantitative permettant de produire de meilleures décisions. Elle fournit des outils pour rationaliser, simuler et optimiser l'architecture et le fonctionnement des systèmes industriels et économiques. Elle propose des modèles pour analyser des situations complexes et permet aux décideurs de faire des choix efficaces et robustes. Ce domaine est lié par un grand nombre de disciplines comme (Programmation Linéaire, Programmation non Linéaire, Programmation Dynamique, La théorie des Graphes,…).

Une des objectives de la **RO**, c'est de trouver souvent la meilleure solution appelée La **Solution Optimale** pour le modèle représentant, cette recherche d'optimalité a été résumée par un terme dit "**Optimisation**". Si cette dernière est linéaire, il s'agit alors de la programmation linéaire.

La **Programmation linéaire** est l'une des plus importantes techniques d'optimisation utilisées en recherche opérationnelle. Il a été développé en fin des années 1940, et à partir de 1951, l'année ou G. Dantzig (1914-2005) Découvrit l'algorithme du simplexe, principale outil de résolution des programmes linéaires. L'objectif de ce domaine est de déterminer l'affectation optimale de ressources rares entre des activités ou produits concurrents, les situations économiques et les problèmes de transport par exemple.

Un autre outil qui permet de la modélisation est la **Théorie des Graphes**, ce domaine est devenu maintenant important de la recherche mathématique. Fondamentalement, elle constitue un outil puissant pour schématiser les modèles des liens et relations entre les objets.

On s'intéresse alors à la programmation linéaire dans la résolution des problèmes d'optimisation dans les graphes, notre sujet parlera sur le problème du transversal minium.

Ce mémoire contient trois chapitres organisés comme suit :

Dans le premier chapitre, nous rappelons quelque définition et notions de base de la programmation linéaire et l'algorithme du simplexe, nous expliquons d'abord les étapes pour formuler un programme linéaire, la Dualité et les formes d'un PL (variable de décision, les contraintes, forme canonique..), et on donne une explication de la méthode du simplex avec des exemples.

Dans le second chapitre nous appelons quelque définitions et notions de base de la théorie des graphes, d'abord nous expliquons le contenu d'un graphe (sommet, arcs,…) et en présentes quelques types des graphes (graphe connexe, graphe biparti, …, etc.). Ensuite on va définir et expliquer notre sujet : problème de transversal minimum dans les graphes biparti.

Dans le troisième chapitre on va faire un programme pour le sujet.

Chapitre1 : Programmation linéaire

Introduction :

La programmation linéaire est un domaine de la recherche opérationnelle qui a été développé à la fin des années 1940, et c'est une méthode permettant d'optimiser (maximiser ou minimiser) une fonction linéaire appeler la fonction objectif de plusieurs variable qui sont relies par des contraintes.

1. Notion fondamentales sur la programmation linéaire

1.1 Programme linéaire :

Un programme linéaire noté PL est défini par les éléments suivants :

1) Les variables de décision x_1, x_2,..., x_n qui sont des réelles.
2) Les contraintes qui traduire les conditions de problème donne avec les variables de décision et peuvent être de type :

- Inégalités " ≤ " :

$$\sum_{j=1}^{n} A_i^j \, x_j \leq b_i \quad , i = \overline{1.m}$$

- Égalités " = " :

$$\sum_{j=1}^{n} A_i^i \, x_j = b_i \quad , i = \overline{m+1,\ m+p}$$

- Inégalités " ≥ " :

$$\sum_{j=1}^{n} A_i^i \, x_j \geq b_i \quad , i = \overline{m+p+1,\ m+p+q}$$

3) la fonction objectif (Max ou Min) donnée par : $\sum_{j=1}^{n} c_j x_j = Z$ à minimiser ou à maximiser. Les nombres A_i^j, b_i , c_j qui sont supposés connus.

1.1.1 La forme standard d'un programme linéaire :

On dit qu'un programme linéaire est sous sa forme standard si toutes les variables sont positives et il n'y a pas de contraintes de type inégalité. En écriture matricielle, on a :

$$\begin{cases} \text{Max Z} = cx \\ x \geq 0 \\ Ax = b \end{cases}$$

Où $A = (A_i^j)$ $une\ m \times n\ matrice$, $b = (b_1, b_2, ..., b_m)^t$,
$c = (c_1, c_2, ..., c_n)$ $et\ x = (x_1, x_2, ..., x_n)^t$.

1.1.2 La forme canonique d'un programme linéaire :

Dans un problème de maximisation (respectivement minimisation), on dit qu'un programme linéaire est sous sa forme canonique si toutes les variables sont positives et il n'y a pas de contraintes de type égalité ni de type "≥" (respectivement "≤").

En écriture matricielle on a :

$$(\text{Pc}) \begin{cases} \text{Max } Z = cx \\ x \geq 0 \\ Ax \leq b \end{cases} \quad res\text{p.} \quad (\text{Pc}) \begin{cases} \text{Min } Z = cx \\ x \geq 0 \\ Ax \geq b \end{cases}$$

Où $A = (A_i^j)$ $une\ m \times n\ matrice$, $b = (b_1, b_2, ..., b_m)^t$, $c = (c_1, c_2, ..., c_n)$ $et\ x = (x_1, x_2, ..., x_n)^t$.

1.2 Notion de base et solution de base et dualité :

1.2.1 Base d'un programme linéaire :

On appelle une base de (PL) ou du système linéaire $Ax = b$ un ensemble $J \subset \{1, ..., n\}$ d'indices de colonnes de A, tel que Aj soit carré régulière .Où Aj est la matrice de base associé à J.

Remarque : Les indices j (resp. les colonnes, les variables xj) sont de base si $j \in J$ et hors base si $j \notin J$.

1.2.2 Solution de base :

La solution x du système $Ax=b$ obtenu en posant $x_j = 0$ est appelé solution de base associée à la base si elle vérifie les contraintes de PL donne et cette solution est donc $x_j = (A^j)^{-1}$ $et\ x_j = 0$.

1.2.3 Solution dégénérée :

Soit J une base de (P). La solution $x_j = (A^j)^{-1}$ et $x_j = 0$ associée à Jest dite dégénérée si la **solution** optimale possède des variables de base nulles, et c'est le cas quand il y a plus de variables de base qu'il y a de contraintes.

1.2.4 Base réalisable :

Une base J de (P) est dite réalisable si la solution de base associée est telle que $x_j \geq 0$ .Cette solution est dite réalisable.

1.2.5 Dualité :

Considérons un programme linéaire sous sa forme canonique

$$(P) \quad \begin{cases} \text{Max } Z = cx \\ x \geq 0 \\ Ax \leq b \end{cases} \qquad \text{(Primal)}$$

On appelle dual de (P), le programme linéaire suivant :

$$(D) \quad \begin{cases} \text{Min } W = yb \\ y \geq 0 \\ yA \geq c \end{cases} \qquad \text{(Dual)}$$

Principales règles d'écriture du dual d'un programme linéaire

Le Dual d'un PL sous la forme général suivre les règles suivant :

Primal	Dual
Min c : coût b : terme de droite	Max b : coût c : terme de droite
Contrainte i du type ≥ Contrainte i du type = Contrainte i du type ≤	Variable y_i du type ≥ 0 Variable y_i du type libre Variable y_i du type ≤ 0
Variable x_j du type ≥ 0 Variable x_j du type libre Variable x_j du type ≤ 0	Contrainte j du type ≤ 0 Contrainte j du type = Contrainte j du type ≥ 0

1.3 Méthode de résolution :

1.3.1 Algorithme du simplexe :

L'**algorithme du simplexe** est un algorithme de résolution des problèmes d'optimisation linéaire.

Il a été découvert en 1947 par George Bernard Dantzig est la plus célèbre méthode pour la résolution numérique des programmes linéaires.

Depuis les années 1985-90, il est concurrencé par les méthodes de points intérieurs, mais garde la 1 ère place.

Pour utiliser cet algorithme le programme linéaire doit être à la forme standard (on ajoute les variables d'écart), et elle consiste à trouver une base optimale en passant par des bases réalisables. La méthode de simplexe est composée de deux phases essentielles :

Phase 1 : Déterminer une première solution de base réalisable et si le système ne contient pas de solution réalisable de base il est impossible à résoudre.

Phase 2 : Calculer, à partir d'une solution de base réalisable (Phase 1) une autre solution de base réalisable meilleure que la précédente (en terme de valeur de la fonction objectif). Leurs Coordonnées correspondent à deux solutions de base réalisable dont l'ensemble des indices de base (idem pour indices hors base) ne diffèrent que d'un seul indice. Cette caractéristique permettra de déterminer la nouvelle solution de base réalisable à partir de l'ancienne.

Les étapes d'Algorithme du simplexe :

1) Initialisation :

On doit écrit le (PL) SFC% à une base réalisable j

$$(PL)\begin{cases} Ax = b \\ x \geq 0 \\ Cx = Max\ (Z) - \alpha^* \end{cases}$$

Application col : {1,……,m} → J donnée . Matrice des coefficients : M

$$\underbrace{X_{col(i)} = b_i\ (i=1,..,m) \text{ et } x_{\bar{J}} = 0}_{\text{Solution de base réalisable associée à J (initiale)}} \qquad \underbrace{Z(x) = \alpha^*}_{\text{évaluation initiale}}.$$

2) Choisir s tel que $c^s > 0$:

2.1) si n'existe pas. Terminer. J est une base optimale, On pose $x^* = x$

x^* solution de base optimale associée à J ; $z(x) = \alpha^*$ évaluation optimale.

2.2) sinon, aller à **3)**

3) Soit I = {i | $A_i^s > 0$} :

3.1) si I = Ø. terminer. (P) n'a pas de solution optimale.

3.2) sinon, aller à **4)**

4) Soit L = $\left\{ l \,\middle|\, \frac{b_l}{A_l^s} = \text{Min}_{(i \in I)} \left\{ \frac{b_i}{A_i^s} \right\} \right\}$; choisir r ∈ L

Effectuer le pivotage (m+1, n+1, r, s ; M) = M` ,

J' = (J∪ {s}) \ {col(r)} nouvelle base réalisable

$x'_s = \frac{b_l}{A_l^s}$; $x'_{col(i)} = b_i - A_l^s x'_s$; $x'_j = 0$; $\alpha = \alpha^* + c^s \frac{b_r}{A_r^s}$. Nouvelle évaluation

Poser J = J', col(r) = s ; x = x' ; $\alpha^* = \alpha$; M = M'. Aller à **2)**

Fin de l'algorithme.

Remarque : Pour un PL à fonction objectif à minimiser, remplacer dans l'algorithme l'instruction « Choisir s tel que $c^s > 0$ » Par « Choisir s tel que $c^s < 0$ ».

Conclusion

Dans ce chapitre, Nous avons mentionné des notions de base de la programmation linéaire et l'algorithme du simplexe qui est l'outil principal de résolution des programmes linéaires.

Chapitre2 : Problème du transversal minimum dans les graphes bipartis

Introduction

Nous présenterons dans ce chapitre les connaissances et les procédés actuellement employés pour le problème du **Transversale minimum** (ou **vertex cover** en anglais). Ensuite, sa modélisation par un passage vers la programmation linéaire. Par conséquent, nous citerons quelques domaines d'application. Puis nous terminerons par un exemple explicatif résolu par un solveur qui utilise l'algorithme du simplexe.

2. Rappel sur la théorie des graphes :

2.1. Graphe :

2.1.1 Définition d'un graphe :

Un graphe est une structure très simple qui est constituée d'un ensemble de sommets et d'une famille de liens entre certains couples de ces sommets.

Le graphe est un outil puissant de modélisation de nombreux problèmes combinatoires et de n'importe quel problème comportant des objets avec des relations entre ces objets (ordonnancement de tâches, carte géographique en coloriant les villes, un réseau de communication, ou encore des relations de domination non-réciproque entre personnes, etc).

2.1.2 Notions sur les graphes :

- **Graphe orienté :**

Un graphe **G = (X, U)** est déterminé par un ensemble **X** fini ($X \neq \emptyset$) d'éléments distincts appelés **sommets** représentés par des points $\{x_1, x_2, \ldots, x_n\}$, et un ensemble **U** de **couples ordonnés** des sommets appelés **arcs** notés $\{u_1, u_2, \ldots, u_n\}$ avec :

$$u = (x_i, x_j) \; ; \; i, j \in \{1, \ldots, n\}$$

Où tout arc u est représenté par une ligne dirigée du sommet x_i vers le sommet x_j à l'aide d'une flèche.

Un peu de vocabulaire

- L'**ordre** d'un graphe G est le nombre de ces sommets, noté par l'entier **n = |X|**.
- La **taille** d'un graphe G est le nombre de ces arcs, notée par l'entier **m = |U|**.

- **Concept non orienté :**

Les sommets de graphe non orienté **G = (X, E)** sont reliés à l'aide des **arêtes**, avec **E** est l'ensemble des arêtes.

Une arête e reliant deux sommets x et y dans un graphe non orienté est notée **e = xy**.

2.1.3 Quelque type des graphes :

Graphe simple : Un graphe est dit simple si deux sommets quelconques de X sont reliés par au plus une arête, et il est dit multi-graphe dans le cas contraire.

Graphe complet : Un graphe est dite complet si pour toute paire de sommets v_i et v_j de G il existe une arête u_k qui les relient.

Graphe biparti : Un graphe est biparti si l'ensemble de ses sommets X peut être partitionné en deux sous-ensembles A et B de sorte que toutes les arêtes du graphe relient un sommet dans A à un sommet dans B uniquement (deux sommets de la même classe ne soient jamais voisins).

- Un graphe biparti peut être noté $G=(X_1 \cup X_2, U)$, avec :

$$n(G) = |X_1| + |X_2| \quad \text{et} \quad m(G) = |X_1|.|X_2|.$$

2.1.4 Matrices associées à un graphe :

1. Matrice d'adjacence

Soit G = (X, U) un 1-graphe avec au plus une boucle par sommet e posons $X = \{x_1, x_2, .., x_n\}$. La **matrice d'adjacence *sommets-sommets*** $B = (b_{ij})_{i=1, n ; j=1, n}$ A coefficients 0 et 1 telle que :

$$B_{ij} = \begin{cases} 1 & \text{si} \quad u = (x_i, x_j) \in U \\ 0 & \text{sinon} \end{cases}$$

2. Matrice d'incidence sommets-arêtes

Soit G = (X, E) un graphe non orienté avec $X = \{x_1, x_2, \ldots, x_n\}$ et $E = \{e_1, e_2, \ldots, e_m\}$ **sans boucle**.

La **matrice d'incidence sommets-arêtes** de G est la matrice $A = (a_{ij})_{i=1,n ;\ j=1,m}$ à coefficients 0 et 1 telle que :

$$A_{ij} = \begin{cases} 1 & \text{l'arête } e_j \text{ incidente au sommet } x_i \\ 0 & \text{sinon} \end{cases}$$

2.2. Problème du transversal minimum :

2.2.1 Définition du problème du transversal minimum :

Soit G = (X, E) un graphe ; T ⊆ X.
L'ensemble T est dit un **transversal** si toute arête e dans G a une extrémité dans T i.e :

$$\forall\, e = ab \in E \quad a \in T \vee b \in T.$$

Le **nombre transversal** de G, noté $\boldsymbol{\tau(G)}$, est le cardinale du plus petit transversal (**transversal minimum**) de G.

Définition d'un couplage maximum :

Etant donné un graphe simple non orienté, on appelle couplage dans G un ensemble d'arêtes de G deux à deux non-adjacentes.
- Un couplage est maximum s'il contenant le plus grand nombre possible d'arêtes.

2.2.2 Domaines d'application :

Le problème de l'assemblage SNP : Nous définissons un graphe conflit où les sommets représentent les séquences dans l'échantillon et il y a une arête entre deux sommets si et seulement s'il y a un conflit entre les séquences correspondantes. Le but est de supprimer le moins de séquences possibles qui élimineront tous les conflits.

Sécurité du réseau informatique : L'idée est de trouver un transversal minimum de sommets dans le graphe, dont les sommets sont les serveurs de routage et dont les arêtes sont les connexions entre les serveurs de routage. C'est une solution optimale pour la propagation des vers et une solution optimale pour la conception de la stratégie de défense du réseau.

Ainsi, il peut être utilisé dans d'autre domaines comme : Le problème des horaires, coloration de cartes et réseaux de téléphonie mobile GSM …etc.

2.3. Résolution et modélisation du problème :

2.3.1 Résolution du problème à l'aide d'un graphe :

Pour ce problème (transversal minimum dans un graphe biparti ou problème de couverture minimal des sommets) il n'y a pas un algorithme exact pour le calculer, il y a seulement le théorème de **KONING** qui dit ≪ dans un graphe biparti le transversal minimum égale au couplage maximum (Thau(G) = v(G)) ≫ tel que le couplage maximum en a déjà vu aussi y'a pas d'algorithme le déterminer.

Alors on propose un algorithme qui donne le résultat direct mais une approximation qui est comme suit :

D'abord, on détermine le nombre de stabilité de G qui est toujours égale à deux car le graphe est biparti donc X(G) = 2 (partitionnement de deux stables S1 et S2).

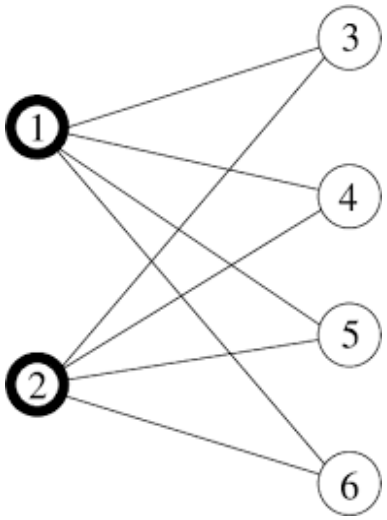
Ensuite, pour tous deux sommets de S1 et S2 qui ont les mêmes arrêts on supprime l'un des eux qui avait le plus petit degré, on répète cette étape de suppression jusqu'à ce que toutes les stables soient disjointes.

Remarque :

Y a des cas que ne doit pas supprimer les sommets qui ont le même degré, jusqu'à la fin qui reste justes ces sommet on observer est ce que la suppression l'un de ces sommet mène à quitter le graphe ou non ? Si oui, on supprime rien et sinon on supprime l'un des eux, mais si tous les sommets avaient le même degré en peut supprimer, pas de problème.

Ainsi, quand on a un sommet (sommet a) qu'est déjà supprimé et qu'est adjacent avec autre sommet (sommet b) on doit jamais supprimer ce sommet (b), on dit directement que b appartient à l'ensemble des sommets du transversale minimum.

Exemple : le graphe G=(X, E)



1°) le sommet (1) et (3) :
$dg(X_1) > dg(X_3)$. Alors on supprime le sommet (X_3).

2°) le sommet (1) et (4) :
$dg(X_1) > dg(X_4)$. Alors on supprime le sommet (X_4).

3°) le sommet (2) et (5) :
$dg(X_2) > dg(X_5)$. Alors on supprime le sommet (X_5).

4°) le sommet (2) et (6) :
$dg(X_2) > dg(X_6)$. Alors on supprime le sommet (X_6).

En fin il reste les deux sommets (X_1) et (X_2), alors le transversal minimum est définit par ces deux sommets donc Thau(G) = 2.

2.3.2 Modélisation en Programme Linéaire :

La formule générale de ce programme linéaire est définie par les éléments suivants :

- La variable de décision :

$$x_i = \begin{cases} 1 \text{ si le sommet } v_i \text{ est dans T} \\ 0 \quad \text{sinon} \quad i = 1,\ldots, n \end{cases}$$

- Les contraintes : $x_i + x_j \geq 1$ pour toute arête x_ix_j de E

Chaque inégalité de formule ≥ 1 cela signifie que chaque contrainte donne au moins un sommet dans le transversal.

- La fonction objective :

$$\text{Min } W = \sum_{i=1}^{n} x_i$$

La fonction objectif vise à minimiser le nombre de sommets dans le transversal, les variables x_i sont binaires, donc x_i prend la valeur 1 si le sommet v_i existe dans le transversale, sinon il prend la valeur 0.
Par conséquent, la somme des résultats obtenue donne le nombre minimum de sommets dans le transversal.

Le PL associé : (PT)
$$\begin{cases} \sum_{i=1}^{n} x_i = W \text{ (Min)} \\ x_i + x_j \geq 1 \text{ pour toute arête } x_ix_j \text{ de E} \\ x_i \in \{0,1\} \; i = 1,\ldots, n \end{cases}$$

Les sommets correspondant à la variable x_i qui valent 1 donne une solution optimale de (PT), formant le transversal minimum.

Le (PL) s'écrit sous forme matricielle :

(PT)
$$\begin{cases} \sum_{i=1}^{n} x_i = W \text{ (Min)} \\ A^t x \geq e \\ x_i \in \{0,1\} \; i = 1,\ldots, n \end{cases}$$

Comme A est totalement unimodulaire, on peut remplacer (PT) par le problème en variables continues :

(PTC)
$$\begin{cases} \sum_{i=1}^{n} x_i = W \text{ (Min)} \\ A^t x \geq e \\ x_i \in \{0,1\} \; i = 1,\ldots, n \end{cases}$$

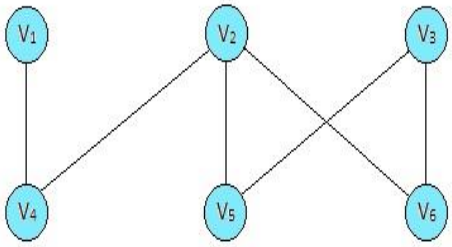
2.3.4 Application :

Le département de sécurité d'un campus veut installer des téléphones d'urgence. Caque rue doit servie par un téléphone, le graphe ci-dessous est la représentation graphique de problème, où chaque arête représente une rue et chaque sommet un carrefour.

Sachant que l'installation de ces téléphones se fait au niveau des carrefours, on veut trouver le nombre minimum de téléphones.

$e_1 = (V_1, V_4)$; $e_2 = (V_2, V_4)$
$e_3 = (V_2, V_5)$; $e_4 = (V_2, V_6)$
$e_5 = (V_3, V_5)$; $e_6 = (V_3, V_6)$

m = 6 ; n = 6



Graphe G

La transposé de la matrice d'incidence (arêtes-sommets) :

$$A^t(G) = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{array} \begin{array}{c} \begin{array}{cccccc} V_1 & V_2 & V_3 & V_4 & V_5 & V_6 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} \end{array}$$

Sa modélisation en programme linéaire est :

$$(PT) = \begin{cases} x_1 + x_4 \geq 1 \\ x_2 + x_4 \geq 1 \\ x_2 + x_5 \geq 1 \\ x_2 + x_5 + x_6 \geq 1 \\ x_3 \geq 1 \\ x_3 + x_4 + x_6 \geq 1 \\ \text{Min } W = \sum_{i=1}^{n} x_i \end{cases}$$

L’exemple est résolue à l’aide du solveur Excel qui un outil de modélisation et de résolution des programmes linéaires.

Solution :

La solution optimale est : $\boldsymbol{x^* = (1, 1, 1, 0, 0, 0)}$

Les sommets V_i correspond à la variable x_i qui valent 1 entre dans le transversal T :

$$\mathbf{T = \{V_1, V_2, V_3\}}$$

La valeur optimale : $\mathbf{W(x^*) = 3}$

Indique qu’il y a au plus 3 sommets dans le transversal.

Conclusion

L’objectif de ce chapitre était de faire une modélisation en programmation linéaire de l’un des problèmes d’optimisation dans les graphes (problème du transversal minimum) avec une résolution à l’aide d’un solveur qui utilise la méthode du simplexe.

Chapitre 3 : Implémentation

Introduction :

Dans ce chapitre on va présenter notre application qui permet de résoudre le problème du Transversal minimum dans les graphes bipartis, cette implémentation est basé sur le langage de programmation 'Matlab'. Tout en présentant notre interface graphique et en expliquant le déroulement de l'application.

3. La programmation :

3.1 Définitions :

3.1.1 La programmation :

La programmation ou le codage, est l'ensemble des activités liées à la création d'un logiciel et des programmes qui le composent, et aussi pour dire à une machine ce qu'elle doit faire et la guider. Cela inclut la spécification du logiciel, sa conception, puis son implémentation proprement dite au sens de l'écriture des programmes dans un langage de programmation bien défini. Dans ce mémoire l'implémentation a été faite via MATLAB.

3.1.2 Matlab :

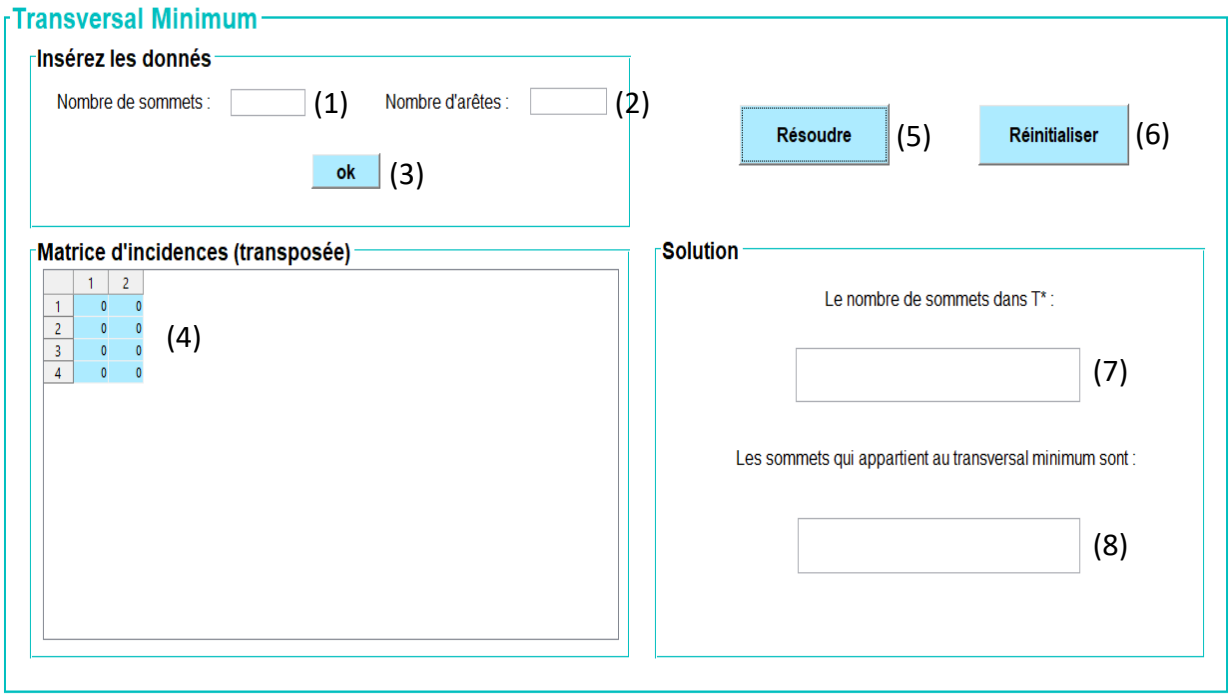
Matlab est en logiciel de calcul matriciel à syntaxe simple, avec ses fonctions
Spécifies Il peut être aussi considéré comme un langage de programmation adapté pour les problèmes scientifiques.

Il propose une interface graphique vers un éditeur de code en Matlab, et un outil de debugging pour exécuter des programmes en mode pas à pas, il est particulièrement performant pour le calcul matriciel, car sa structure de données interne est basée sur les matrices.

3.2 Présentation du logiciel :

Dans cette partie, nous allons faire expliquer et présenter notre application.

3.2.1 Présentation de l'interface graphique :



(1) Champ pour entrer le nombre de sommets.
(2) Champ pour entrer le nombre d'arêtes.
(3) Bouton pour régler les dimensions de la matrice suivant le nombre des sommets de (1) et d'arêtes de (2).
(4) Champ pour l'insertion de la matrice d'incidence (transposée).
(5) Bouton pour résoudre le problème.
(6) Bouton de réinitialiser les données.
(7) Champ pour afficher le nombre des sommets dans le transversal optimal.
(8) Champ pour afficher les sommets qui appartient au transversal minimum.

3.2.2 Insertion des données :

Dans cette partie, l'utilisateur doit saisir le nombre de sommets n et d'arêtes m. sachant que les sommets et les arêtes représentent respectivement les colonnes et les lignes de la transposée de la matrice d'incidence. Ensuite, les données sont traitées pour voir si elles respectent les conditions du graphe.

Si les données sont correctes, cette dernière affichera une matrice avec m Ligne et n Colonnes, et vous pourrez remplir les données de la matrice. Sinon, elle affichera un message d'erreur.

3.2.3 Traitement et Affichage des résultats :

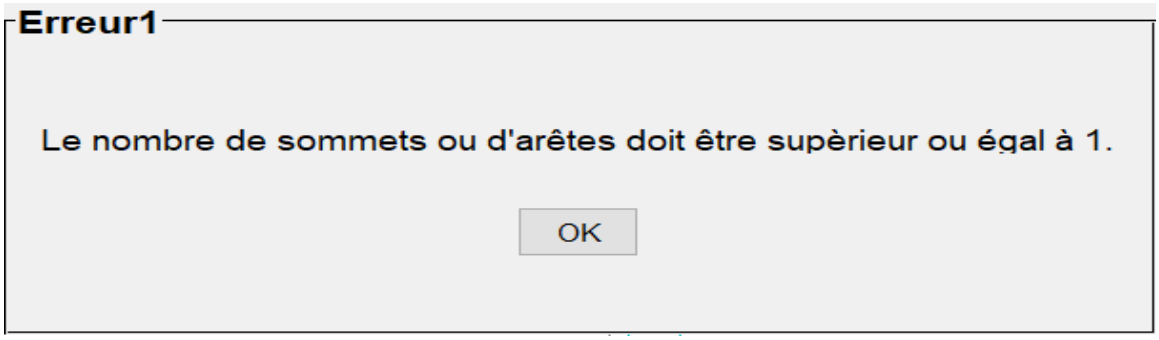
Dans cette étape, l'application doit faire deux traitements, le premier traitement est une Etude sur la matrice, après que l'utilisateur ait cliqué sur le bouton « Résoudre ». Si la matrice respecte les conditions, elle passe vers le deuxième traitement où notre résultat du transversal s'affichera. Sinon, une erreur sera affichée.

Remarque :

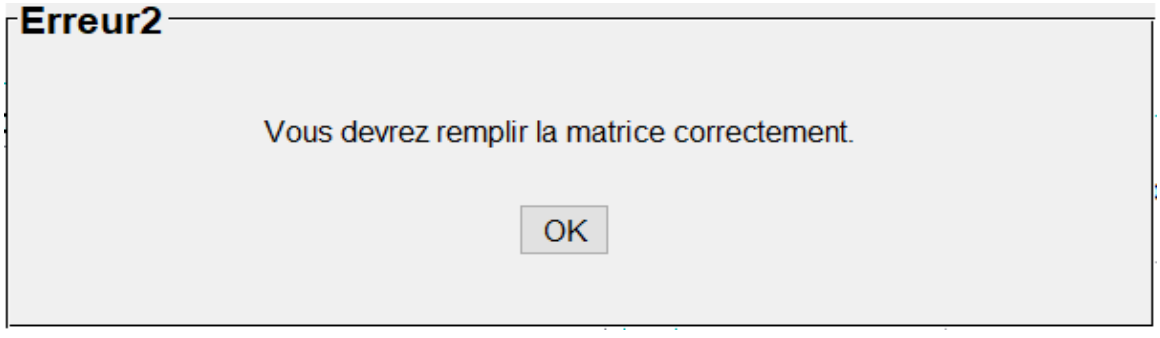
Le nombre de sommets correspond au nombre des colonnes de la matrice, et le nombre d'arêtes correspond au nombre des lignes. Car, dans le problème de transversal minimum on s'intéresse à le transposé de la matrice d'incidence.

Messages d'erreurs :

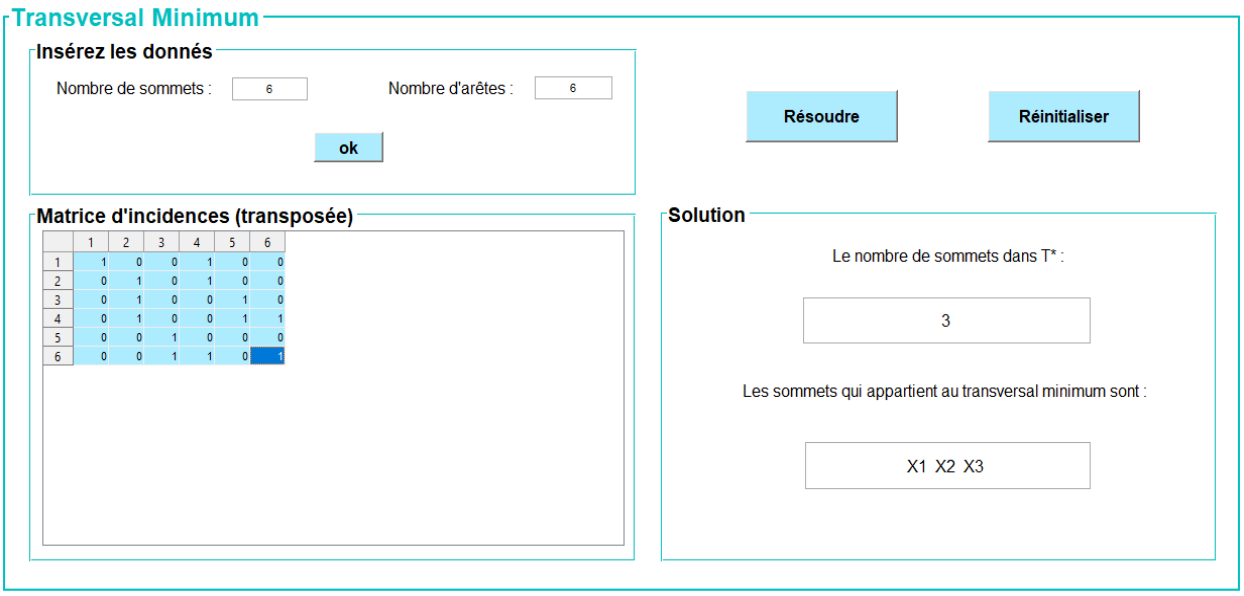
- Lors de l'insertion, si le nombre des sommets ou d'arêtes est inférieur ou égal à 0, alors le message d'erreur suivant s'affiche :



- Lors de l'insertion, si une ligne ou plus est nulle, alors le message d'erreur suivant s'affiche :



3.3 Exemple :



L'application nombre minimum des sommets qui sont réalisé, et les sommets qui entrent Dans le transversal minimum.

Conclusion

Dans ce chapitre nous avons réalisé et présenté notre application qui permet de résoudre le problème du transversal minimum dans les graphes bipartis.

Conclusion générale

Dans ce mémoire, on a d'abord fait un rappel sur la programmation linéaire et la théorie Des graphes, par la suite utilisé une combinaison des deux notions pour modéliser et résoudre le problème du transversal minimum.

Nous poursuivons notre étude par l'implémentation de la recherche d'un Transversal Minimum dans les graphes bipartis sous le langage **MATLAB**.

Notre modeste recherche nous a donné l'occasion de voir la richesse du sujet, tout en Sachant que ce dernier est très important et efficace grâce à sa diversité dans son application dans le monde réel.

Finalement, nous espérons que les travaux en cours seront bénéfiques à tous les lecteurs De ce projet.

Bibliographie

[1] V. Chvatal, Linear Programming, Freeman, 1983.

[2] R. Faure, B. Lemaire et C. Picouleau, Précis de recherche opérationnelle Méthodes et exercices d'application, Dunod, 2008.

[3] M. Gondran, M. Minoux, Graphes et Algorithmes, Eyrolles, 1995.

[4] Christine Solnon, Théorie des graphes set optimisation dans les graphes.

[5] M. Minoux, Programmation Mathématiques : Théorie et Algorithmes, Tec & Doc Lavoisier ; édition : 2e édition, 14 décembre 2007.

[6] M. Sakarovitch, Optimisation combinatoire-Graphes et programmation linéaire, Hermann, 1984.

[7] JOEL M.Zinsalo Cour Recherche Opérationnelle, Ecole polytechnique D'Abomey-Calavi.

[8] Nadjette, PSEP2 Recherche opérationnel, MAI2014.

[9] Shanthi A.S., Matching Problems in Certain Interconnection Networks and Chemical Graphs, University of Madras, Department of Mahematics.