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

Mémoire

En vue de l'obtention du Diplôme de Licence
En Recherche Opérationnelle

Thème

problème du recouvrement minimum dans les graphes bipartis

Présenté par : CHICHIOU Manel
KHELILI Chams el sabah

Encadré par : Mme A.HISSOUM

Année universitaire : 2020/2021

Table des matières

Introduction général

La recherche opérationnelle est un ensemble de méthodes (algorithmiques et mathématiques, après passage par la modélisation) afin de prendre des décisions optimales ou proches de l'optimum pour des problèmes complexes (production, transport, construction, communication, planification financiére, santé, militaire...). Ce domaine est fortement lié à l'ingénierie des systèmes et au management du système d'information.

La programmation linéaire est un domaine de la recherche opérationnelle et un cadre mathématique général pour la modélisation et la résolution de certains problèmes d'optimisation. L'application initiale était dans le domaine militaire(George Bernard-1947) , mais elle s'est rapidement tournée vers la planification industrielle et économique .

Le présent mémoire contient trois chapitres :

Le premier chapitre est consacré aux définitions de base sur la programmation linéaire , et sur deux méthodes de résolution des programmes linéaire : méthode du simplexe et la méthode des deux phases.

Le deuxième chapitre après avoir faire un bref rappel sur la théorie des graphes , on intéressera sur le thème de recouvrement minimum, toutes on notons les notions et les définitions nécessaires des graphes , plus précisément les graphes bipartis, puis, on va introduire le programme linéaire associé au problème du recouvrement minimum avec une application sur un example de graphe biparti , et on terminera avec l'utilité de problème posé dans la vie réelle .

Enfin, dans le dernier chapitre on s'intéressera à l'implémentation du programme linéaire adéquat et la résolution du problème avec Python (création d'une application qui trouve la solution optimale sur un graphe quelconque) .

Chapitre 1

La programmation linéaire et la méthode du Simplexe .

1.1 Introduction

La programmation linéaire est l'une des plus importantes techniques d'optimisation utilisées en recherche opérationnelle.
Un programme linéaire (PL) est un problème d'optimisation consistant à maximiser (ou minimiser) une fonction objectif linéaire de n variables de décision soumises à un ensemble de contraintes exprimées sous forme d'équations ou d'inéquations linéaires.

A l'origine , le terme programme a le sens de planification opérationnelle mais il est aujourd'hui employé comme synonyme de problème (d'optimisation). La terminologie est due à G. B. Dantzig, inventeur de l'algorithme du simplexe.

L'essentiel de ce chapitre présente les idées de base et les techniques de la programmation linéaire et de la méthode du Simplexe, qui est une méthode numérique de résolution de programmes linéaires.

1.2 Rappel sur la programmation linéaire

1.2.1 Définitions

Soient D un domaine de R^n et une fonction f : D $\rightarrow$ R .

Définition 1 :

Un programme mathématique noté (P), est le problème qui consiste à chercher un élément x^* de D tel que $f(x^*)$ soit le plus grand possible en cas de maximisation (resp. le plus petit possible en cas de minimisation). C'est-à-dire $\forall x \in D : f(x) \leq f(x^*)$ en cas de maximisation (resp. $\forall x \in D : f(x) \geq f(x^*)$ en cas de minimisation) .

Définition 2 :

Soit (P) un programme mathématique.

- Un point $x \in D$ est appelé "solution réalisable" de (P).
- Le domaine D est appelé "domaine des solutions réalisables" de (P).
- Une solution x^* de (P), lorsqu'elle existe, est appelée "solution optimale" de (P).
- La fonction f est appelée "fonction objectif " de (P).
- $f(x^*)$, si x^* existe, est appelée "valeur du programme mathématique" (P).

Remarque :

résoudre le programme mathématique $\min_{x \in D} f(x)$, revient à resoudre le programme mathématique $\max_{x \in D}(-f(x))$.

1.2.2 Formulation d'un programme linéaire (PL)

Un problème de programmation linéaire est défini par les éléments suivants :

— **Les variables de décisions** $(x_1, x_2, ..., x_n)$ sont réelles .
Certaines variables sont toujours positives $(x_i > 0)$, d'autres sont toujours négatives $(x_i < 0)$ et les autres peuvent avoir n'importe quel signe $(x_i \in \mathbb{R})$.

— **Les contraintes** peuvent être de type :

- inégalités "$\geq$" ($\sum_{j=1}^{n} A_i^j x_j \geq b_i$) .
- inégalités "$\leq$" ($\sum_{j=1}^{n} A_i^j x_j \leq b_i$) .
- égalités "$=$" ($\sum_{j=1}^{n} A_i^j x_j = b_i$) .

— **L'objectif** (critère, indice de performance, ou fonction économique ...) donné par :

$$\sum_{j=1}^{n} c_j x_j = Z$$

Le but d'un problème d'optimisation est d'optimiser l'objectif sur l'ensemble admissible (maximiser ou minimiser).

1.2.3 La forme standard d'un PL

L'écriture matricielle d'un programme sous forme standard est donné comme suit :

$$(P_s) \begin{cases} cx = Z(max) & x \geq 0 \\ Ax = b \end{cases}$$

Où c et b sont deux vecteurs et A est une matrice ayant des valeurs entières .

1.2.4 La forme canonique d'un PL

L'écriture matricielle d'un programme sous forme canonique est donné comme suit :

$$(P_c) \begin{cases} cx = Z(max) & x \geq 0 \\ Ax \leq b \end{cases} \left(resp. \ (P_s) \begin{cases} cx = Z(min) & x \geq 0 \\ Ax \geq b \end{cases} \right)$$

Où c et b sont deux vecteurs et A est une matrice ayant des valeurs entières .

Pour la résolution d'un programme linéaire on utilise des méthodes mathématiques ou des algorithmes qui servent à trouver les ou la solution (s) optimale ou proche de l'optimum au problème examiné. Si on parle de résolution graphique alors on doit se limiter à une représentation à deux variables et au plus à trois variables.

Parmi les méthodes les plus utilisées , nous citerons un processus itératif qui convergera vers la (les) solution(s) de base admissible(s) fournissant ainsi l'optimum de la fonction objectif en cas d'existence. C'est la méthode du simplexe.

1.3 Rappel sur la méthode du simplexe

L'algorithme du simplexe est la plus célèbre méthode pour la résolution numérique des programmes linéaires.
Analytiquement parlant, l'algorithme consiste à passer itérativement d'une solution de base réalisable non dégénérée x, à une solution de base réalisable non dégénérée voisine x_0 , en laquelle la valeur de Z est plus élevée, partant d'un point x_0.

La méthode du simplexe se décompose en deux phases essentielles :

1.3.1 Phase I : (procédure d'initialisation)

Elle consiste à Déterminer une première solution de base réalisable , i.e les coordonnées d'un premier sommet de l'ensemble des solutions réalisables D. Si cette procédure échoue, cela signifie que D est vide, donc le problème est impossible (les contraintes ne peuvent être satisfaites simultanément). Sinon, le domaine des solutions réalisables n'est pas vide ; et soit x_0 une première solution de base réalisable (trouvée).

1.3.2 Phase II : (procédure itérative)

Elle consiste à Calculer, à partir d'une solution de base réalisable (trouvée en phase I), une autre solution de base réalisable donnant une valeur meilleure de la fonction objectif. Géométriquement, une itération consiste à passer d'un sommet de D à un autre sommet de D adjacent au précédent (dans le sens où elles sont les deux extrémités d'une arête de D). Leurs coordonnées correspondent à deux solutions de base réalisables dont les ensembles d'indices de base (donc aussi 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.
Comme tout algorithme, il existe une procédure d'arrêt de l'algorithme : deux tests d'arrêt mettent en évidence, respectivement que :

— une solution optimale est déterminée

— il n'existe pas de solution optimale finie : D est non borné et Z peut y tendre vers l'infini.

Notre étude de la méthode du simplexe abordera en premier lieu la Phase II avant la PhaseI, vu que cette dernière utilise la première.

1.3.3 Théorèmes

- **Théorème fondamental de la PL :**
  Etant donné un PL :
  $$(P_s)\begin{cases} cx = Z(max) & x \geq 0 \\ Ax = b \end{cases}$$
  Tel que Ax = b est un système plein rang
  1) Si (P) admet une solution réalisable, alors il admet une solution réalisable de base .
  2) Si (P) admet une solution optimale alors il admet une solution optimale de base .
- Dans un problème de maximisation (ou Minimisation) Une base réalisable J, telle que le vecteur cout relatif à la base J est négatif (positive) ou nul est appelée une base optimale.
- Soit J une base réalisable de (P). Si le vecteur coût relatif à J est négatif ou nul, alors la solution de base associée à J est une solution optimale de (P).

1.3.4 L'algorithme du Simplexe

1) Initialement :
le PL $(P)\begin{cases} cx = Z(max) - \alpha^* & x \geq 0 \\ Ax = b \end{cases}$
est écrit SFC% par rapport à une base réalisable J , avec : z(x) =α^* (évaluation initial) .
Application col : $\{1, ..., m\} \to J$ donnée .
Matrice des coefficients : M $x_{col(i)} = b_i$ ($\forall i = \overline{1, m}$) et $x_{\bar{j}} = 0$ comme une solution de base réalisable associée a J (initiale).

2) Choisir s tel que $c^s > 0$:
2.1) si s n'existe pas. Terminer. J base optimale. Poser x^* = x
x^* solution de base optimale associée à J ; z(x)= α^* évaluation optimale.
2.2) sinon, aller à **3)**

3) soit I = { i|A_i^s>0 } :
3.1) si I = $\emptyset$. Terminer. (P) n'a pas de solution optimale.
3.2) sinon, aller à **4)**

4) soit L = $\left\{ l | \frac{b_l}{A_l^s} = min_{i \in I}\{\frac{b_i}{A_i^s}\} \right\}$; Choisir r $\in$ L .
effectuer le pivotage(m + 1, n + 1, r, s ; M) = M' ; $J' = (J \cup \{s\}) \backslash \{col(r)\}$ nouvelle base réalisable .
$x'_s = \frac{b_r}{A_r^s}$; $x'_{col(i)} = b_i - A_i^s x_s^i$; $x'_j = 0$; $\alpha = \alpha^* + c^s \frac{b_r}{A_r^s}$ nouvelle évaluation .
poser J := J' , col(r) :=s ; x :=x' , $\alpha^* := \alpha$, M := M' . aller à **2)**
Fin de l'algorithmme .

Exemple :

Considérons le programme linéaire suivant, ainsi que sa forme standard

$$(P)\begin{cases} 6x_1 - x_2 \leq 6 \\ 2x_1 - x_2 \leq 4 \\ x_1 - x_2 \leq 5 \\ x_1 - 3x_2 = Z(max) \end{cases} \qquad (P_s)\begin{cases} 6x_1 - x_2 + x_3 = 6 \\ 2x_1 - x_2 + x_4 = 4 \\ x_1 - x_2 + x_5 = 5 \\ x_1 - 3x_2 + = Z(max) \end{cases}$$

On a une base réalisable J= {3,4,5} , $A^j = I_3$ est inversible.

$$(A^j)^{-1} \times b = \begin{pmatrix} 6 \\ 4 \\ 5 \end{pmatrix} \times c_j = 0$$

On pourra alors appliquer l'algorithme du simplexe car on a une base initiale :

T_1	x_1	x_2	x_3	x_4	x_5	b
x_3	6	-1	1	0	0	6
x_4	2	1	0	1	0	4
x_5	1	-3	0	0	1	5
Z	1	-3	0	0	0	0

Comme c'est une fonction à maximiser, alors le cout relatif à la base optimale doit être inférieur ou égale à zéro, ce qui n'est pas le cas dans le tableau T1 alors on va appliquer un pivotage (4, 4, 1, 1, T1) .
la variable sortante de base sera x_3 tandis que la variable entrante sera x_1.

T_1	x_1	x_2	x_3	x_4	x_5	b
x_1	1	-1/6	1/6	0	0	1
x_4	0	4/3	-1/3	1	0	2
x_5	0	-5/6	-1/6	0	1	4
Z	0	-17/6	-1/6	0	0	-1

On a $c \leq 0$ donc $x^* = (1, 0, 0, 2, 4)^t$ est une solution optimale de base avec $Z^* = 1$

et $c^j \leq 0$ donc la solution optimale x^* est réalisable et unique.

1.4 Conclusion

la programmation linéaire ou la modélisation mathématique et ses différents outils , aide à résoudre plusieurs problématiques réels dans plusieurs domaines .
en particulier certains problèmes dits problèmes de répartition dans les graphes .

Chapitre 2

Problème du recouvrement minimum dans les graphes bipartis .

2.1 Introduction

Nous introduisons d'abord dans ce chapitre les termes de base de la théorie des graphes, puis on va résoudre le problème d recouvrement minimum dans les graphes bipartis liés à l'application de résolution dans l'exemple correspondant réel.

2.2 Généralité

Les graphes représentent un outil puissant qui peut modéliser de nombreux problèmes combinés qui seraient autrement difficiles à réaliser avec des techniques classiques .

En plus d'être un objet mathématique, les graphes sont également des structures de données puissantes dans de nombreux domaines. En effet , les graphes sont irremplaçables dés qu'il s'agit de décrire la structure d'un ensemble complexe, en exprimant les relations, les dépendances entre ses éléments : réseaux de communication, réseaux routiers ...

L'étude mathématique des graphes fait partie de la théorie des graphes. La théorie étudie les propriétés des graphes et établit des théorèmes, des conditions nécessaires ou suffisantes à leur sujet. Il fournit également des algorithmes Pour les problèmes spécifiques à la théorie des graphes, tels que les problèmes de distribution dans les graphes .

2.2.1 Definition d'un graphe

Un graphe fini G = (V,E) est défini par l'ensemble fini V =v_1,v_2,..., v_n dont les éléments sont appelés sommets (Vertices en anglais), et par l'ensemble fini E =e_1, e_2,..., e_m dont les éléments sont appelés arêtes (Edges en anglais).
Une arête e de l'ensemble E est définie par une paire non ordonnée de sommets, appelés les extrémités de e. Si l'arête e relie les sommets a et b, on dira que ces sommets sont adjacents, ou incidents avec e, ou bien que l'arête e est incidente avec les sommets a et b. On appelle ordre d'un graphe le nombre de sommets n de ce graphe .

2.2.2 Représentation graphique

Les graphes tirent leur nom du fait qu'on peut les représenter par des dessins. À chaque sommet de G, on fait correspondre un point distinct du plan et on relie les points correspondant aux extrémités de chaque arête. Il existe donc une infinité de représentations d'un graphe.

2.2.3 Quelques types de graphes

Le graphe simple : Un graphe est simple si au plus une arête relie deux sommets et s'il n'y a pas de boucle sur un sommet. On peut imaginer des graphes avec une arête qui relie un sommet à lui-même (une boucle), ou plusieurs arêtes reliant les deux mêmes sommets. On appelera ces graphes des multigraphes .

Le graphe connexe : Un graphe est connexe s'il est possible, à partir de n'importe quel sommet, de rejoindre tous les autres en suivant les arêtes. (Un graphe non connexe se décompose en composantes connexes) .

Le graphe complet : Un graphe est complet si chaque sommet du graphe est relié directement à tous les autres sommets.

Le graphe biparti : Un graphe est biparti si ses sommets peuvent être divisés en deux ensembles X1 et X2, de sorte que toutes les arêtes du graphe relient un sommet dans X1 à un sommet dans X2 ie :

$X_1 \bigcup X_2 = X$ et $X_1 \bigcap X_2 = \emptyset \quad \forall u \in U \ , \ X_1 \in I(U) \Rightarrow X_2 \in T(U)$.

nous nous intéressons aux graphes bipartis ....

2.2.4 Matrice d'incidence sommets - arêtes

Soit G=(X,E) un graphe non orienté , avec |G|= n et |E|=m , sa matrice d'incidence $A(G) = (a_{ij})$ à n lignes (correspondants aux sommets) et m colonnes(associés aus arêtes) :

$$a_{ij} = \begin{cases} 1 & si \ \ \mathrm{e}_j \text{ est incident à } \mathrm{x}_i \\ 0 & sinon \end{cases}$$

Remarque :
La matrice d'incidence sommets - arêtes d'un graphe G = (X ; E) est totalement unimodulaire si et suelemenent si G est un graphe biparti .

Définition : Une matrice M est dite totalement unimodulaire si toute sous-matrice carrée extraite de M a un déterminant = 0, +1 ou -1.

2.3 Problème du recouvrement minimum

2.3.1 Définitions

- **Le recouvrement :**

  Un recouvrement des sommets par des arêtes est un sous-ensemble des arêtes F de E tel que pour chaque sommet v dans E , il existe au moins une arête e dans F tel que v est une extrémité de e .
  Observation : Un recouvrement ne peut exister que si G n'a pas sommets isolés.

- **Le recouvrement minimum :**

  En théorie, un recouvrement minimum d'un graphe est un ensemble des arêtes minimum (le plus petit nombre d'arête possibles) tels que chaque sommet du graphe est incident à au moins une arête de l'ensemble. En informatique, le problème de la couverture minimum est celui de trouver une couverture minimale. C'est un problème d'optimisation qui appartient à la classe des problèmes de couverture et qui peut être résolu dans le temps polynomial...

- on note $\rho(G)$ le cardinal du recouvrement minimum dans G .

$$\rho(G) = min\{|F| : F \ \ est \ \ un \ \ recouvrement\}$$

2.3.2 Modélisation sous forme de PL

Soit G=(X,E) un graph non orienté , avec |G|= n et |E|=m .
le problème de recouvrement est modélisé avec un programme linéaire en 0-1 .
les variables de décisions x_j représentent les arêtes $e_j \in E$, tell que :

$$x_j = \begin{cases} 1 & si\ l'arete\ j\ est\ dans\ le\ recouvrement \\ 0 & sinon \end{cases}$$

(on associe la variable x_j à la colonne j , j = $\overline{1,m}$) d'ou , la solution est représenté par le vecteur binaire X = [x_1,..., x_m]

la fonction objectif :

le but est d'avoir le nombre minimum des arêtes couvrant tous les sommets , donc la fonction objective est :

$$\sum_{j=1}^{m} x_j = W_{min}$$

le rôle des contraintes :

pour tout sommet i on parcourt tout les arêtes j incidentes , tell que au moins une arête est dans le recouvrement F . d'ou :

$$\sum_{j=1}^{m} a_{ij}x_j \geq 1$$

$$i = \overline{1,n}.$$

les contraintes sous forme matricielle :

la $1^{ère}$ contrainte sera écrit sous forme matricielle , comme suit :

$$A1 \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{11}, & a_{12}, \ldots, & a_{1m} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ xn \end{pmatrix} \geq 1$$

Ainsi de suit jusqu'à la n^{ime} contrainte :

$$An \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{n1}, & a_{n2}, \ldots, & a_{nm} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ xn \end{pmatrix} \geq 1$$

d'ou, (PR) le programme linéaire associé au problème de recouvrement :

$$(PR) \begin{cases} \sum_{j=1}^{m} x_j = W(min) \\ \sum_{j=1}^{m} a_{ij}x_j \geq 1 \qquad i = \overline{1,n} \\ x_j \in \{0,1\} \qquad\qquad j = \overline{1,m} \end{cases}$$

soit A la matrice d'incidence sommets - arêtes de graphe G .
A est la n.m-matrice où :

$$x_j = \begin{cases} 1 & si\ l'arête\ \mathrm{e}_j\ \ est\ \ incidente\ \ au\ \ sommet\ v_i\ \ ,\ \ i = \overline{1,n}\ \ ,\ \ j = \overline{1,m} \\ 0 & sinon \end{cases}$$

Si e est le m-vecteur colonne dont toutes les composantes sont égales à 1 .
(PR) s'écrit sous forme matricielle :

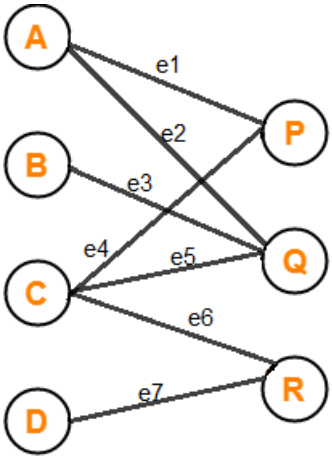
$$(PR) \begin{cases} \sum_{j=1}^{m} x_j = W(min) \\ Ax \geq e \\ x_j \in \{0,1\} \qquad j = \overline{1,m} \end{cases}$$

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

$$(PRC) \begin{cases} \sum_{j=1}^{m} x_j = W(min) \\ Ax \geq e \\ 0 \leq x_j \leq 1 \qquad j = \overline{1,m} \end{cases}$$

2.4 Application de PL sur un graphe biparti

soit le Graphe biparti G :



on aura sa matrice d'incidence sommets - arêtes comme suit :

$$A = \left[\begin{array}{c|ccccccc} & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 \\ \hline v_1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline v_2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline v_3 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ \hline v_4 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline v_5 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline v_6 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ \hline v_7 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]$$

cette matrice est unimodulaire .

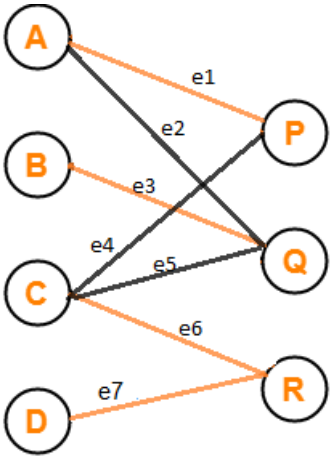
d'ou, le programme linéaire associé au problème de recouvrement de graphe est :

$$(PR') \begin{cases} \sum_{j=1}^{7} x_j = W(min) \\ x1 + x2 \geq 1 \\ x3 \geq 1 \\ x4 + x5 + x6 \geq 1 \\ x7 \geq 1 \\ x1 + x4 \geq 1 \\ x2 + x3 + x5 \geq 1 \\ x6 + x7 \geq 1 \\ x_j \in \{0,1\} \quad j = \overline{1,m} \end{cases}$$

et sa forme matriciel est : (en variables continues)

$$(PRC') \begin{cases} \sum_{j=1}^{7} x_j = W(min) \\ A \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x7 \end{pmatrix} \geq \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \\ 0 \leq x_j \leq 1 \quad j = \overline{1,7} \end{cases}$$

Aprés la résolution par la mèthode du simplexe (on peut le résoudre par un solveur qui utilise principalement les méthodes du simplexe) , on trouve la solution est X=[1,0,1,0,0,1,1] avec : W(min)=4 .
i.e : le recouverement contient les arêtes e_1 , e_3 , e_6 , e_7 , on aura :



2.5 Projection du problème sur la réalité

On va citer un exemple de notre vie réelle pour mieux comprendre l'application de problème de recouvrement minimum :
Il s'agit d'un musée que l'on cherche à surveiller (par des caméras de surveillance) d'une façon économique . une caméra (arête) placée dans un couloir peut surveiller les deux carrefours (sommets) placés à ses extrémités , donc le problème revient à chercher le nombre minimum possible de caméras utilisées afin que tous les carrefours soient surveillés , et la où vient le rôle de notre problème du recouvrement minimum qui va modéliser le problème posé .

De manière générale, la projection de notre problème sur nos domaines de la vie réelle est multiple , on peut citer quelques domaines comme :

- ◇ les réseaux de communications ou de télécommunication.
- ◇ les réseaux de transport.
- ◇ la gestion d'horaires et la planification de tâches.
- ◇ l'industrie dans touts ses champs.

Donc le problème du recouvrement minimum dans les graphe posséde énormèment d'applications pratiques d'où l'intérêt qu'il suscite aprés de la communauté scientifque .

2.6 Conclusion

Finalement on a trouvé une solution optimale à cette problématique de recouvrement minimum sur un graphe biparti donnée en utilisant le programme linéairee associè à ce problème et la mèthode de simplexe .

Dans le prochain chapitre, on va implémenter la solution donnée par le programme linéaires associé au problème de Recouvrement minimum sur un graphe quelconque avec un nombre quelconque de sommets et d'arêtes , en utilisant cette fois un logiciel de programmation .

Chapitre 3

Implémentation du programme linéaire sur un logiciel de programmation .

3.1 Introduction

Tout problème mathématique peut être transformé en un algorithme qui peut être implémenté dans un logiciel de programmation . La mise en œuvre de ce problème consiste à traduire chaque étape de l'algorithme correspondant au problème en un ensemble d'instructions (programme). On peut citer plusieurs logiciels développés pour le traitement de ces problèmes tel que : **Java**, **Python** , **C** , **C++** , **Pascal** , **Matlab** ...

Dans ce chapitre on va implémenter la solution donnée par le programme linéaire associé au problème de recouvrement minimum en utilisant le logiciel **Python** qui nous donne la possibilité de créer une interface graphique, qui permet à n'importe qu'elle utilisateur de résoudre ce problème de manière simple juste en insérant les données nécessaires au programme, et ainsi d'avoir une solution optimale de PL .

3.2 Description du logiciel

Python est un langage de programmation open source créé par un programmeur en 1991 (le plus employé par les informaticiens). Il s'agit d'un langage de programmation interprété (qui ne nécessite pas d'être compilé pour fonctionner). Un programme "interpréteur" permet d'exécuter le code Python sur n'importe quel ordinateur .

Tkinter (Tool kit interface) est la bibliothèque graphique libre d'origine pour le langage Python, permettant la création d'interfaces graphiques. Elle vient d'une adaptation de la bibliothèque graphique Tk écrite pour Tcl (Tool Command Language).

3.3 Application sous Python

3.3.1 Création de l'interface graphique

En utilisant Tkinter , Nous avons créé une interface graphique qui demande à l'utilisateur d'entrer le nombre de sommets (n) et d'arêtes (m) , Et d'introduire la matrice d'incidence sommets-arêtes. ainsi, nous avons integré le PL associé au problème de recouvrement minimum dans la partie programmation de l'application .

Pour personaliser l'affichage , modifier les couleurs , les tailles , les positions et changer le type de texte il suffit d'éditer les propriétés de chaque commande de Tkinter .

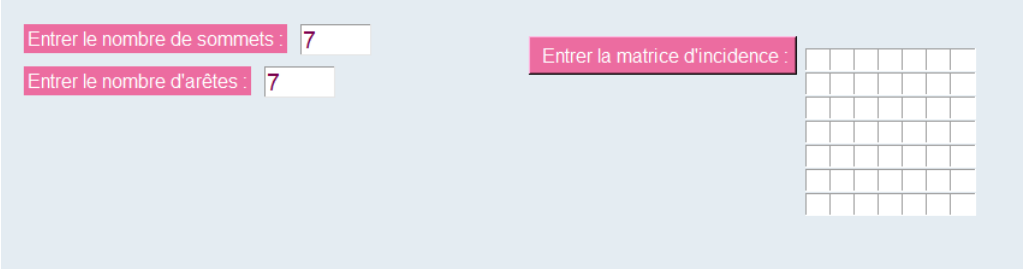
3.3.2 Mode d'utilisation de l'application

Comme on a dit précédement ce programme est fait pour n'importe utilisateur qui veut résoudre le problème de recouvrement minimum dans un graphe quelconque . Dans cette partie nous allons exprimer comment faire fonctionner notre application pas à pas :

<u>Introduction des données</u> :

La figure ci-dessous représente l'interface graphique lors de l'exécution :

- Dans la boîte de dialogue il est demandé d'introduire le nombre de sommets du graphe qui représente les lignes de la matrice d'incidence sommets-arêtes, et le nombre d'arêtes du graphe qui représente les colonnes de la matrice d'incidence .
- Après avoir introduit la taille de la matrice, en cliquant sur le bouton " **Entrer la matrice d'incidence** " le programme donne la main à l'utilisateur pour faire entrer la matrice d'incidence sommets-arêtes . Comme suit :



L'affichage des résultats :

Après l'introduction des données citées auparavant , les résultats seront affichées en cliquant sur le bouton "**trouver le recouveremnt minimum**" , on obtient dans le bloc en dessous le vecteur solution x* (les arêtes touchées par le recouvrement minimum) et la valeur de la fonction objectif Z* qui représente le cardinal du recouvrement (le nombre minimum des arêtes couvrant tous les sommets du graphe : $\rho(G)$) .

La figure ci-dessous représente l'affichage des résultats :

Remarque :

Dans cette figure on a introduit les données d'un graphe biparti qui a 6 sommets , 6 arêtes et une matrice d'incidence sommets-arêtes qui est totalement unimodulaire . L'application nous a donnée la solution X^*=[1,0,0,1,0,1] avec Z^*=3 .
i.e : le recouverement minimun contient 3 arêtes e_1 , e_4 , e_6 .

Comme d'autre **fonctionalité** dans l'application , on a les boutons :

- **Exemple :** Si on clique sur le bouton "Exemple" , l'application nous affiche l'exemple qu'on a vu dans le deuxième chapitre, qui est considéré prédéfini dans l'application . Il admet une solution optimal affichée dans la case solution X* et une valeur Z* optimale associée . Comme il est claire dans la figure suivante :

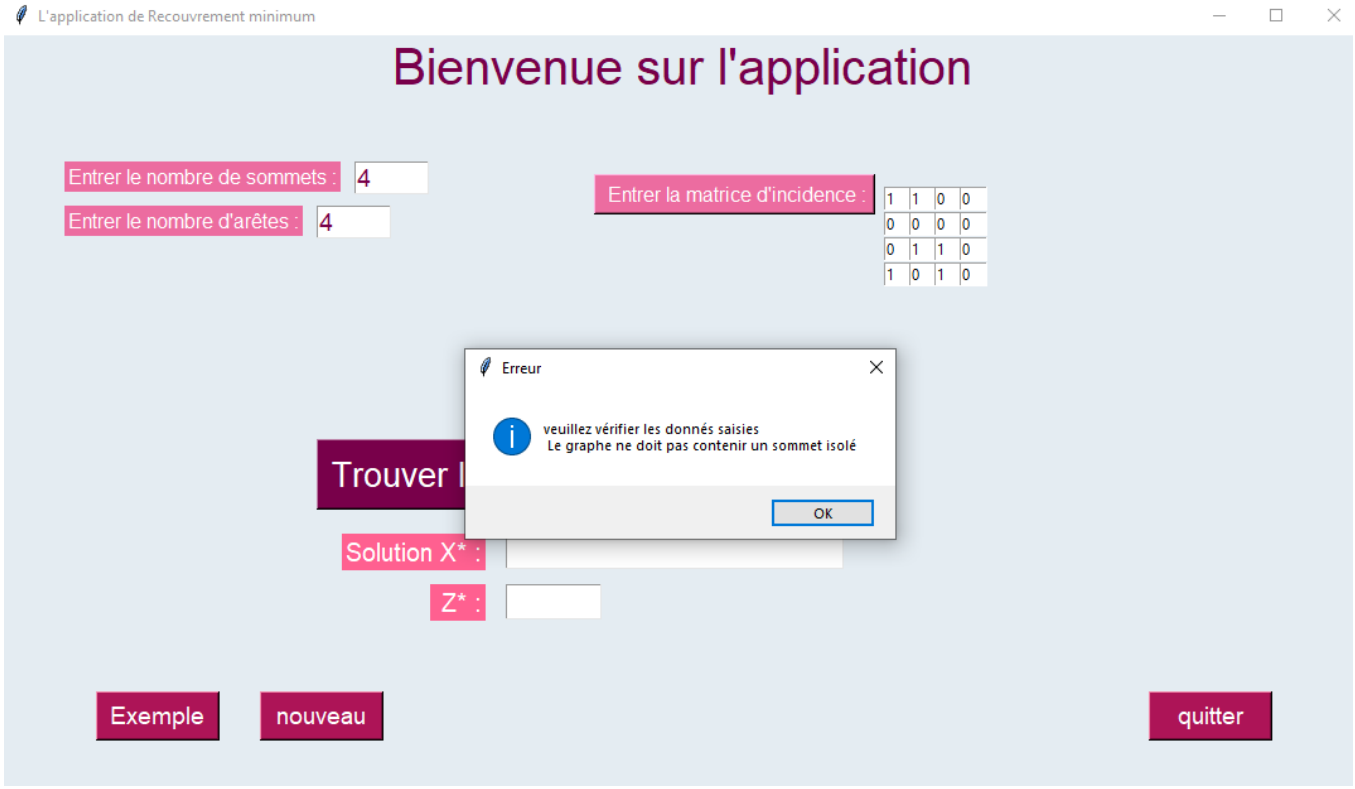
- **Nouveau :** En appuyant sur le bouton "Nouveau" on reprend depuis la première étape qui consiste à remplir les champs des nombres de sommets et d'arêtes et insérer une nouvelle matrice d'incidence sommets-arêtes quelconque .
- **Quitter :** permettre de fermer la fenêtre et quitter l'application .

Les messages d'erreurs :

L'application donne un résultat fiable pour un graphe quelconque , sauf si l'utilisateur saisi des données erronées. Dans ce cas là, un message d'erreur va s'afficher sur l'écran en précisant l'erreur , et on cite parmi les erreurs :

- Le cas où il saisi dans la matrice d'incidence un nombre différent de 1 ou 0 .
- Le cas où il saisi une matrice d'incidence qui a au moins une ligne nulle (le graphe contient au moins un sommet isolé) .

La figure ci dessous représente le message d'erreur affiché dans le deuxième cas :



3.4 Conclusion

Dans ce chapitre , on a créé une application associée au problème de recouvrement minimum en utilisant le logiciel Python .
cette application est trés simple et compréhensible, ce qui facilite à l'utilisateur la recherche d'une solution au problème posé (si elle existe), ainsi de trouver la solution optimale en moins du temps .

Conclusion générale

Cette thèse nous a permis de nous familiariser avec deux outils de la recherche opérationnelle, qui sont la modélisation des problèmes par les graphes et la résolution à travers la programmation linéaire . Et de faire le passage de l'un à l'autre afin de trouver une solution optimale qui déjouant les contraintes.

Dans ce mémoire nous avons présenté une méthode de résolution de problèmes de recouvrement minimum, cette méthode est basée sur le simplexe . Ce qui nous a montré que la méthode de simplexe est trop efficace pour les problèmes de la recherche du recouvrement minimum dans les graphes bipartis .

L'objectif de ce travail est de présenter une solution du problème posé, et d'implémenter les résultats à l'aide d'une application fiable plus simple à utiliser que nous avons nous même développé en python .

Finallement, nous souhaitons que ce présent mémoire soit bénéfique pour l'ensemble des lecteurs de ce projet .

BIBLIOGRAPHIE

1) Opt Lin-Cours-Sec-A (2019-2020)

2) K.G. Murty. Linear programming. John Willy and Sons, 1983

3) Daniel A.Spielman et Shang-Hua Teng « Smoothed analysis of algorithms : Why the simplex algorithm usually takes polynomial time », Journal of the ACM, vol. 51, no 3, 2004

4) Christelle Guéret, Christian Prins et Marc Sevaux : "Programmation Linéaire". Eyrolles, 2000. ISBN 2-212-09202-4, 365 pages.

5) Eric Jacquet-Lagrèze. Programmation Linéaire - Modélisation et mise en œuvre informatique Collection : P.I.Q. Poche - Editeur : Economica

6) Armen S. Asratian, Tristan M. J. Denley et Roland Häggkvist, « Bipartite Graphs and their Applications », Cambridge university press, 1916

7) D. Avis, T. Imamura. Operations Research Letters 35 (2007)

8) Recouverement minimum L3RO