République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة بجاية
Université de Béjaïa
Tasdawit n'Bgayet

Université Abderrahmane Mira de Bejaia
Faculté des Sciences Exactes
Département de Recherche Opérationnelle

MÉMOIRE DE FIN D'ÉTUDES

En vue de l'obtention du diplôme de

MASTER en Recherche Opérationnelle

Option : Modélisation Mathématique et Evaluation des Performances des Réseaux

THÈME

Quelques Techniques d'optimisation dans les Réseaux Et Applications

Présenté par :

***M^r* YAHIAOUI Seyf-eddine**
***M^r* MAOUNI Naim**

Devant le jury composé de :

Président **: *M^r* TALEM Djamel**
Rapporteur **: *D^r* KABYL Kamal**
Examinateur **: *M^r* TAOUINET Smail**
Examinateur **: *M^r* BOUTEBEL Djalil**

Promotion 2017 — 2018

Remerciements

Louange A Dieu, le miséricordieux, sans Lui rien de tout cela n'aurait pu être.

La réalisation de ce mémoire fut une occasion merveilleuse de rencontrer et d'échanger avec de nombreuses personnes nous ne saurions pas les citer toutes sans dépasser le nombre de pages raisonnablement admis dans ce genre de travail.

Nous reconnaissons que chacune à des degrés divers, mais avec une égale bienveillance, apporté une contribution positive à sa finalisation. Nos dettes de reconnaissance sont, à ce point de vue, énormes à leur égard.

Nous tenons à remercier Dr Kabyle pour la finesse de ses attitudes sur le plan aussi bien humain que scientifique. Ses remarques successives ont permis d'améliorer les différentes versions de ce travail. Il a toujours trouvé comme promoteur le juste équilibre entre la liberté qu'il nous a laissé dans le choix des grandes orientations et dans la détermination des pistes à suivre .

Nous tenons à remercier également les membres de jury d'avoir accepté d'évaluer ce travail et pour toutes leurs remarques et critiques.

Nous tenons à remercier tous ceux qui sont présents avec nous aujourd'hui.

Merci à vous tous

Dédicaces

A cœur veillant, rien d'impossible.
A conscience tranquille, tout est accessible.
Quand il y a la soif d'apprendre.
Tout vient à point à qui sait attendre.
Les études sont avant tout notre unique et seul atout.
Souhaitant que le fruit de nos efforts fournis jour et nuit
Nous mènera vers le bonheur fleuri.

Nous dédions ce modeste travail :

A nos très chères mères et nos pères qui nous ont toujours encouragés pour que nous réussissions dans nos études.

A ceux qui ont veillés pour notre bien-être.

A nos frères et nos très chères sœurs pour leurs patiences et leurs confiances.

A toute nos familles , à tous nos amis de proche ou de loin.

Tous ceux qui nous aiment et que nous aimons ; et à tous ceux qui nous ont encouragés et crus en nous durant notre vie.

Seyf-eddine YAHIAOUI
Naim MAOUNI

TABLE DES MATIÈRES

TABLE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION GÉNÉRAL

La Recherche Opérationnelle (R.O) attend toujours sa définition « définitive ». Pour le moment, on se contentera de la définir en ces mots : « ensemble de méthodes et de techniques "rationnelles" d'analyse des phénomènes de management du système d'information, utilisable pour élaborer de meilleures décisions. »

Ses propositions scientifiques sont des modèles issus d'organisations réelles, aptes à analyser des situa-tions souvent complexes aux termes desquelles les décideurs parviennent à se fixer des choix les plus efficaces. Outils et domaines où intervient la recherche opérationnelle sont aussi nombreux que vastes, certains sont insoupçonnés encore ! L'un de ces outils, la Théorie des graphes, a déjà établi sa renommée d'instrument efficace résolvant la plupart des problèmes discrets posés à la recherche opérationnelle. Claude BERGE reconnaît que la R.O [36]. Contribue aux progrès de la Théorie des graphes, avec ce qu'elle suscite de nouvelles recherches.

La Théorie des graphes est aujourd'hui très présente dans notre société moderne. Cette branche des mathématiques, dont on fait remonter l'origine à EULER (1736), a connu un essor spec-taculaire au cours du demi-siècle passé, particulièrement grâce à l'impulsion que lui a donnée Claude Berge qui la popularisa. Parce qu'elle permet de modéliser divers problématiques, elle intervient dans toutes les politiques : transports, affectations, travaux publics, organisation de la circulation routière, affrè-tement, etc.

Les graphes constituent donc une méthode efficiente, bien que pour l'instant, le gros de son concours repose sur les arcs et les sommets. Il permet de représenter la structure et les connexions d'un ensemble complexe en exprimant les relations entre ses éléments : réseaux de communication, réseaux routiers, circuits électriques, etc. Le but de notre travail est de résoudre un problème réel, en recourant justement à cette Théorie, traitant précisément de l'optimisation de réseaux.

Cette étude s'articule autour de trois chapitres. Le premier donne un aperçu assez concis sur la théorie des graphes (définitions et concepts de base). Au deuxième chapitre, on s'intéressera aux réseaux (Graphes connexes) valuées en donnant les nombreuses méthodes de résolution et leur ap-

plication sur les problèmes modélisés. Au troisième et dernier chapitre, nous aborderons quelques situations réelles qui résument le but de notre travail, à savoir l'optimisation des réseaux, qui vont être résolus par la suite en utilisant les algorithmes cités dans le deuxième chapitre, ce qui conduira à élaborer des résultats pour chaque problème.

Nous finissons par restituer les différents résultats de recherche du plus court chemin sur le réseau routier algérien, obtenus à travers l'exécution de l'algorithme de Dijkstra implémenté sous le logiciel **JAVA**.

CHAPITRE 1

DÉFINITIONS DE BASE ET NOTATIONS

Introduction

Les graphes ont ceci de propre qu'ils permettent de modéliser toute situation à interactions entre ses objets. Les techniques utilisées, relevant de la théorie des graphes, arrivent à répondre à beaucoup de problèmes algorithmiques. En étudier les propriétés de ces problèmes revient à étudier les propriétés structurelles de leurs topologies représentées par des graphes. Nous introduisons successivement, au fil de ce chapitre, les différentes notions couramment utilisées dans la Théorie des graphes et dont nous aurons à nous en servir dans ce document.

1.1 Définitions et notations

1.1.1 Graphe

Définition 1.1.1. un graphe non orienté G (ou plus simplement graphe) est un couple d'ensemble finis (X(G),E(G)) où E(G) est constitué de paires de X(G). Les éléments de X(G) sont appelés sommets de G et ceux de E(G) arêtes de G. Si aucune confusion n'est possible on note X au lieu de X(G) et E au lieu de E(G). On utilisera les notations simplifiées xy ou yx pour l'arête $\{x,y\}$. Une arête de type xx est appelée boucle de G. L'ordre d'un graphe est le cardinal de son ensemble de sommets, noté |X(G)|. Si $e{=}xy$ est une arête de G on dit que y et x sont voisins dans G et qu'il forment les extrémités de e. Deux arêtes sont dites incidentes si elles ont une extrémité en commun. On définit le voisinage d'un sommets x dans le graphe G comme l'ensemble de ses voisins, on le note $N_G(x)$ ou N(x) s'il n'existe pas d'ambiguïté sur le graphe considéré. Le degré d'un sommet x dans G, noté d_G(x) est le cardinal de $N_G(x)$. Le maximum des degrés des sommets de G est noté par Δ(G) et le minimum des degrés des sommets de G est noté par δ(G) . Un sommet de degré 0

est dit sommet isolé et un sommet de degré 1 est dit sommet pendant. Si tous les sommets ont le même degré, on dira que le graphe G est régulier et si $\Delta(G)=\delta(G)$, alors G est dit $\Delta(G)$-régulier. Dans un graphe G, une arête ***e*** reliant un sommet ***y*** à un sommet ***x*** est notée par ***(y x)*** et dans ce cas on dit :

— y et x sont adjacents.

— y et x sont les extrémités de e ; e incidente à y et x.

— Deux arêtes sont dites adjacents si elles sont incidentes à un même sommet.[21],[31]

la figure 1.1 montre un graphe non orienté à 4 sommets et 5 arêtes.

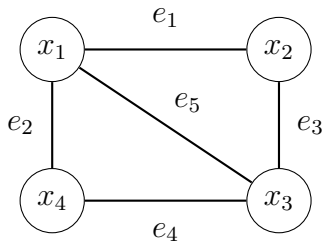


FIGURE 1.1 – Graphe non orienté

Définition 1.1.2. Dans le cas d'un graphe orienté, $G = (X, E)$ est formé d'un ensemble fini X de sommets et d'un ensemble de $E \subseteq X \times X$ d'arcs (flèches). Soit ***x*** un sommet d'un graphe orienté. On note d^+***(x)*** le nombre d'arcs ayant ***x*** comme extrémité initial, et d^-***(x)*** le nombre d'arcs ayant ***x*** comme extrémité terminale. Ainsi, on a : ***d(x)***$=d^+$***(x)***$+d^-$***(x)***.{[2], [5]}

la figure 1.2 montre un graphe orienté à 4 sommets et 5 arcs.

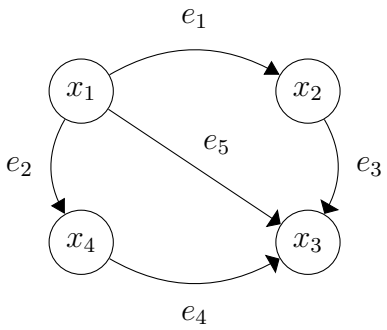


FIGURE 1.2 – Graphe orienté

•**Sous-graphe :**[13] Soit G=(X,E) un graphe, alors G'=(X',E') est dit sous-graphe de G X' inclut dans X et l'ensemble des arêtes E' est formé par toutes les arêtes de E dont les extrémités sont dans X'.
la figure 1.3 montre le sous-graphe de la figure 1.1

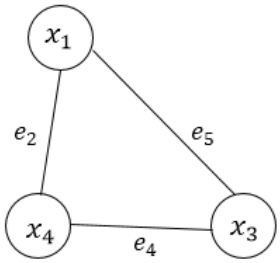


FIGURE 1.3 – sous-graphe

•**Graphe partiel :**Soit G=(X,E) un graphe, alos H=(Y, A) est dit graphe partiel de G=(X,E) si Y=X et $A \subset E$.
la figure 1.4 montre un graphe partiel de graphe de la figure 1.1

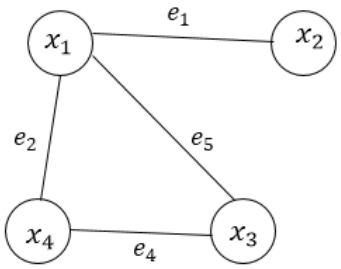


FIGURE 1.4 – Graphe partiel

•**Complémentaire d'un graphe :** Soit $G = (X, E)$ un graphe, le complémentaire de G est donné par $\overline{G} = (X, \overline{E})$ tel que $xy \in \overline{E}$ si et ssi $xy \notin E$.

1.1.2 Chaînes, cycles, chemins et circuits

Définition 1.1.3. [29] Soit G=(X,E) un graphe :

• **Chaîne** : est une séquence de sommets $x_0, x_1, ..., x_k$ Telque $\forall i \in 0, 1, ..., k-1$, x_i est relié à x_{i-1} par une arête et à x_{i-1} par une autre arête.

La longueur de la chaîne est égale au nombre d'arcs qui la composent.

- Si aucune des sommets composant la séquence n'apparaît plus d'une fois, la chaîne est dite chaîne ***élémentaire***.
- Si aucune des arêtes composant la séquence n'apparaît plus d'une fois, la chaîne est dite chaîne ***simple***.
- Une chaîne ***eulérienne*** est une chaîne empruntant une fois et une fois seulement chaque arête de G.
- On appelle chaîne ***hamiltonienne*** une chaîne passant une fois, et une fois seulement, par chacun des sommets de G.

•***Cycle*** : Un cycle est une chaîne dont les extrémités coïncident.

- Un cycle ***élémentaire*** est un cycle minimal pour l'inclusion, c'est-à-dire ne contenant aucun autre cycle.
- Un cycle ***eulérien*** est une chaîne eulérienne dont les extrémités coïncident.
- Un cycle ***hamiltonien*** est un cycle qui passe une fois, et une seule fois, par chacun des sommets de G.

•***Un Chemin*** :Un chemin conduisant du sommet a au sommet b est une suite ayant pour éléments alternativement des sommets et des arcs, commençant et se terminant par un sommet, et telle que chaque arc est encadré à gauche par son sommet origine et à droite par son sommet destination.

- Si aucun des sommets composant la séquence n'apparaît plus d'une fois, le chemin est dit chemin ***élémentaire***.
- Si aucun des arêtes composant la séquence n'apparaît plus d'une fois, le chemin est dit chemin ***simple***.
- Un chemin ***eulérien*** est un chemin simple qui passe une et une seule fois par chaque arc du graphe.
- On appelle chemin ***hamiltonien*** un chemin passant une fois, et une fois seulement, par chacun des sommets de G.

•***Circuit*** : Un circuit est un chemin dont les extrémités coïncident (G orienté).

- Un circuit est dit ***élémentaire***, si on ne rencontre pas deux fois le même sommet.
- Un circuit ***eulérien*** est un chemin eulérien dont les extrémités coïncident.
- Un circuit ***hamiltonien*** est un circuit qui passe une fois, par chacun des sommets de G.

- Remarque :

— Un graphe possédant un cycle eulérien (cas non-orienté) ou un circuit eulérien (cas orienté) est appelé graphe eulérien.

— On dit qu'un graphe G est hamiltonien s'il contient un cycle hamiltonien (cas non-orienté) ou un circuit hamiltonien (cas orienté).

— Un chemin hamiltonien (une chaîne hamiltonienne) est donc un chemin (une chaîne) élémentaire de longueur *(n-1)*.

1.1.3 Connexité

Un graphe G=(X,E) est dit connexe si pour toute paire de sommet (x,y), il existe une chaîne joignant x et y.La figure suivante montre bien un graphe connexe.[26]

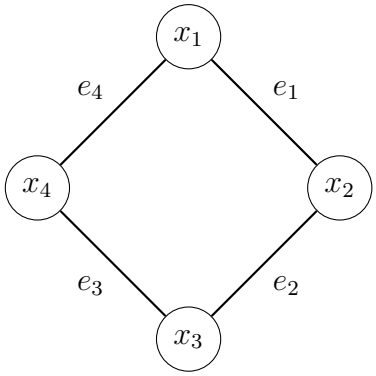


FIGURE 1.5 – Graphe connexe

1.1.4 Forte connexité

Un graphe G=(X,E) est dit fortement connexe si $\forall\, x,y \in X$, il existe un chemin de x vers y et de **y** à **x**. Une composante fortement connexe (cfc) est un sous-ensemble de sommets tel qu'il existe un chemin entre deux sommets quelconques. Une composante fortement connexe maximale (cfcm) est un ensemble maximal de cfc. Les différentes cfcm définissent une partition de X. Un graphe est fortement connexe s'il comporte une seule cfcm.

Comme algorithme de recherche de nombre de cfc, nous avons l'algorithme de marquage suivant :

•**Algorithme** (Algorithme de marquage){[26], [29]}

- Répéter

1) partir d'un sommet quelconque n'appartenant pas à une cfc, le marquer par ±.

2) Sur l'ensemble des sommets non marqués par ± et tant qu'on peut marquer un sommet faire :

(a) Marquer par + tout sommet suivant d'un sommet marqué par +.

(b) Marquer par - tout sommet précédent d'un sommet marqué par -.

Tout sommet marqué par ± appartient à une cfc.

- Jusqu'à ce que tout sommet appartienne à une cfc.

Après avoir utilisé l'algorithme précédent, on obtient le résultat montré sur le graphe suivant : Les différents composantes connexes sont : c_1=$\{x_1, x_2, x_4\}$;c_2=$\{x_8\}$; c_3=$\{x_6, x_7\}$

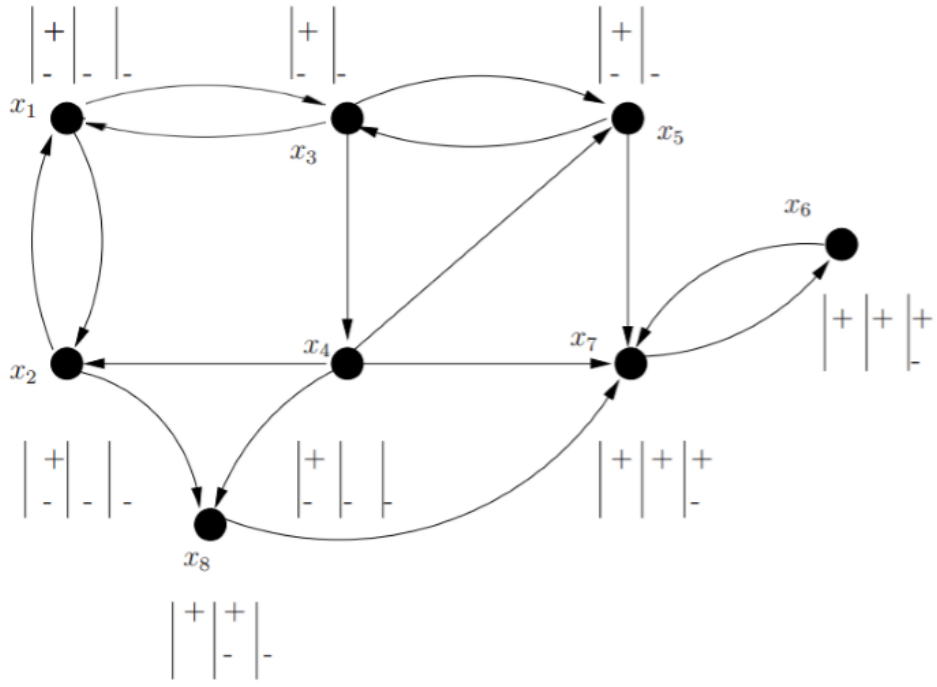


FIGURE 1.6 – Composantes fortement connexes

1.1.5 Couplage

Un couplage M d'un graphe G=(X,E) est un ensemble d'arêtes deux à deux non adjacentes. Un sommet de graphe est dit saturé par le couplage M, si il est l'extrémité d'une arête de M, si tous les sommets de G sont saturé alors le couplage est dit parfait.

— Un couplage est dit maximum s'il est de cardinalité maximum..

— Le couplage maximum est le couplage couvrant le plus grand nombre insaturé de sommets possibles, en laissant donc le moins de sommets non saturés.[9]

le graphe de la figure 1.7 montre un couplage parfait.

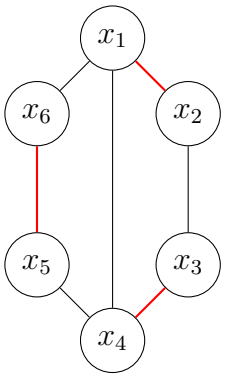


FIGURE 1.7 – Couplage parfait

Les arêtes rouges donnent un couplage parfait

1.1.6 Distances dans les graphes

Étant donné deux sommets x et y d'un graphe G=(X,E), on appelle distance entre x et y, la longueur d'une plus courte (x,y)-chaîne (en terme de nombre d'arcs) qui les relies et on la note $d_G(x,y)$ (ou $d(x,y)$ s'il n'y a pas de confusion). Une telle chaîne s'appelle *géodésique*.

— L'excentricité d'un sommet `x` noté $e_G(x)$ (ou e(x) s'il n'y a pas de confusion) est le nombre suivant :

$$e_G(x) = \max_{y \in X(G)} d_G(x,y).$$

— Le diamètre de G noté D(G) (ou D s'il n'y pas de confusion) est la plus grande excentricité :

$$D(G) = \max_{x \in X(G)} [e(x)].$$

Un sommet $x \in$ X(G) est dit diamétral de $y \in$ X(G) si $d_G(x,y)$=D(G). Si chaque sommet de G admet un unique sommet diamétral, on dira que G est diamétral.

— Le rayon de G noté R(G) (ou R s'il n'y pas de confusion) est la plus petit excentricité :

$$R(G) = \min_{x \in X(G)} [e(x)]$$

— Le centre de G est l'ensemble des sommets de G dont l'excentricité est égale au rayon.

— La distance entre deux arêtes x_1y_1 et x_2y_2 d'un graphe G=(X,E) est définie par :

$$d_G(x_1y_1,x_2y_2)=\min\{d_G(x_1x_2),d_G(x_1y_2),d_G(y_1x_2),d_G(y_1y_2)\}.\{[22],[14],[1]\}$$

1.2 Quelques graphes particuliers

1.2.1 Graphe complet

Un graphe G=(X,E) est dit complet si pour toute paire de sommets (x_i,x_j), il existe un arête entre x_i et x_j. Un graphe complet d'ordre n est noté $\mathtt{K}_n$.[22]
la figure 1.8 montre un graphe complet à 5 sommets.

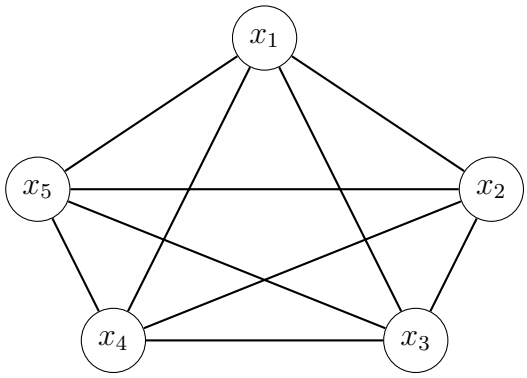


FIGURE 1.8 – Graphe Complet K_5

1.2.2 Graphe biparti

Un graphe G=(X,E) est dit biparti s'il existe une partition de l'ensemble des sommets de G en deux sous-ensembles X_1 et X_2, tel que toute arête de G a une extrémité dans X_1 et l'autre extrémité dans X_2.[22]
la figure 1.9 montre un graphe biparti.

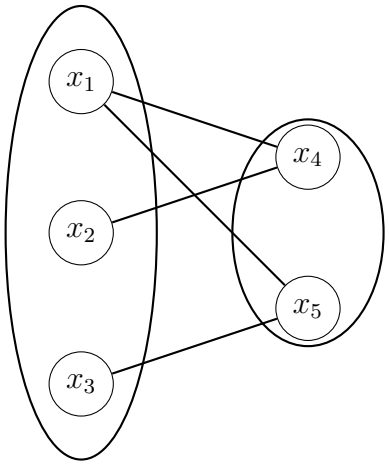


FIGURE 1.9 – Graphe biparti

1.2.3 Graphe biparti complet

Soit $G = (X_1 \cup X_2, E)$ un graphe biparti. Si pour tout $x_1 \in$X$_1$ et pour tout $x_2 \in$X$_2$ on a x_1 et x_2 est une arête ($\forall x_1 \in$X$_1$,$\forall x_2 \in$X$_2$,(x_1,x_2) $\in$E), alors le graphe biparti $G = (X_1 \cup X_2, E)$ est dit biparti-complet. Un graphe simple biparti-complet avec |X$_1$|=p,|X$_2$|=q se dénote par $\mathtt{K}_{p,q}$.[22]
la figure 1.10 montre le graphe biparti complet $K_{3,2}$.

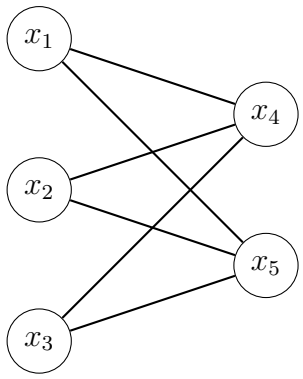


FIGURE 1.10 – Graphe biparti complet $K_{3,2}$

1.2.4 Graphe biparti équilibré

Un graphe biparti G est dit équilibré si chaque ensemble de la bipartition est de même taille. Les graphes ayant des cycles de longueur paire, c'est-à-dire les graphes biparti complet, pour lesquels $|X_1| = |X_2|$, sont des exemples de graphes équilibrés.[13]
la figure 1.11 montre un graphe biparti équilibré.

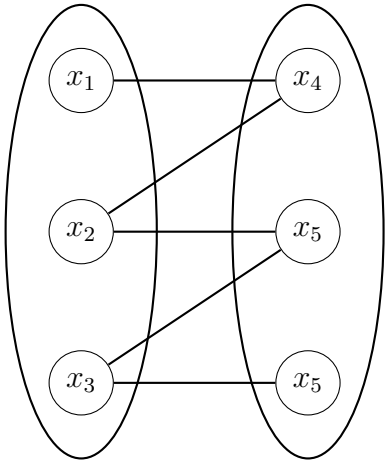


FIGURE 1.11 – Graphe biparti équilibré

1.2.5 Graphe planaire

On dit qu'un graphe G est planaire s'il est possible de le représenter sur un plan de sorte que les sommets soient distincts, les arêtes des courbes simples, et que deux arêtes ne se rencontrent pas en dehors de leurs extrémités.[13]
la figure 1.12 montre un graphe planaire.

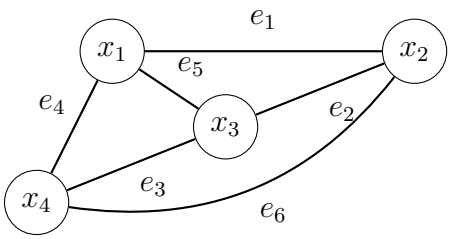


FIGURE 1.12 – Graphe planaire

par contre le graphe de la figure 1.13 n'est pas planaire :

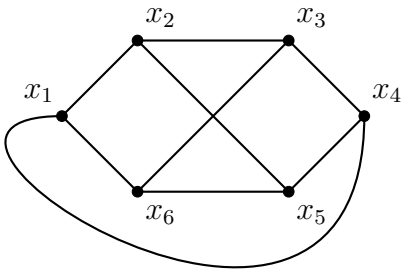


FIGURE 1.13 – Graphe non planaire

nous remarquons que les deux acrs $(x_2, x_5) et (x_3, x_6)$ sont croisés

1.2.6 Graphe K-régulier :

Un graphe G=(X,E) est dit K-régulier si :

$$\forall\ x \in \mathtt{X} \text{ on a } d_G(x) = \mathtt{K}$$

1.2.7 Arbres et Arborescence

•Arbres :[29],[32]

soit G un arbre à n sommets $n \geqslant 2$. les propriétés suivantes sont équivalentes :

1) G connexe et sans cycle

2) G sans cycle et possède n-1 arêtes

3) G connexe et possède n-1 arêtes

4) G sans cycle et si ajoutant une arêtes, on crée un et un seul cycle

5) G connexe et si on supprime une arête, il n'est plus connexe

6) Il existe une chaîne et une seule entre toutes paires de sommets.

Un graphe G vérifiant les une des propriétés ci-dessus est un arbre

la figure 1.14 montre un arbre à 8 sommets

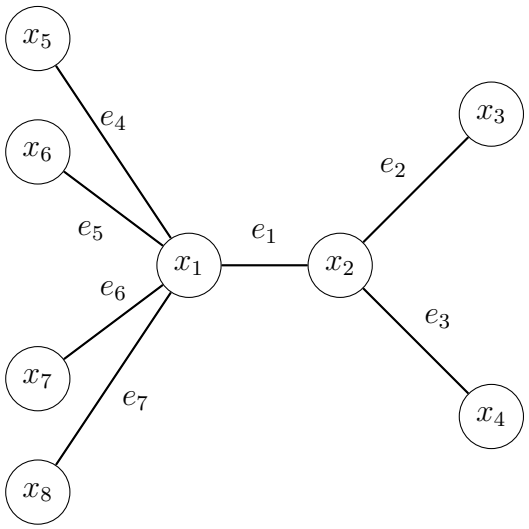


FIGURE 1.14 – Arbre

•Arborescence :[29],[32]

Une arborescence est un graphe orienté sans circuit admettant une racine $x_0 \in X$ telle que pour tout autre sommet $x_i \in X$,il existe un chemin unique allant de x_0 vers x_i. Si l'arborescence comporte n sommets alors elle comporte exactement $(n-1)$ arcs.
le graphe de la figure 1.15 est une arborescence.

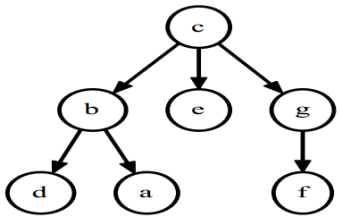


FIGURE 1.15 – Arborescence

1.3 Représentation Matricielle des graphes

1.3.1 Matrice d'adjacence

Considérons un graphe orienté G=(X,E) comportant $\boldsymbol{n}$ sommets. La matrice d'adjacence de G est définie par M=(m_{ij}), telle que :

$$m_{ij} = \begin{cases} 1 & Si \quad (i,j) \in E \\ 0 & Sinon \end{cases}$$

Pour un graphe non orienté la matrice d'adjacence est définie comme suit :

$$m_{ij} = \begin{cases} 1, & si(i,j)ou(j,i) \in E \\ 0, & sinon \end{cases}$$

Une telle matrice, ne contenant que des "0" et des "1" est appelée, de manière générale, une matrice booléenne.[12]

Soit le graphe orienté suivant a quatre sommets

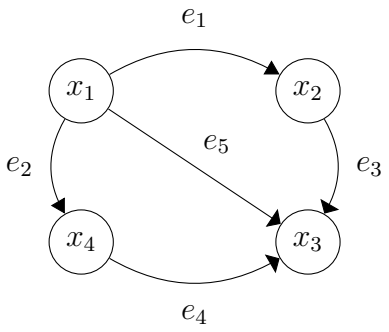


FIGURE 1.16 – graphe orienté

La matrice d'adjacence associée au graphe de la figure 1.16 est :

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Soit le graphe non orienté suivant

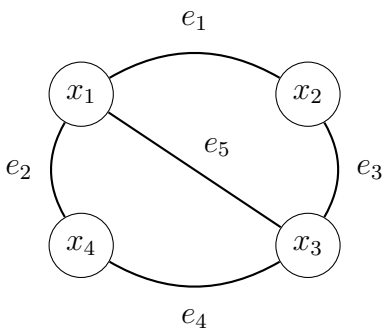


FIGURE 1.17 – graphe non orienté

La matrice d'adjacence associée au graphe de la figure 1.16 est :

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

1.3.2 Matrice d'incidence

Considèrerons un graphe orienté sans boucle G=(X,E) comportant $\boldsymbol{n}$ sommets $\{x_1,x_2,x_3,...,x_n\}$ et $\boldsymbol{m}$ arêtes $\{e_1,e_2,e_3,...,e_m\}$, On appelle matrice d'incidence de G la matrice M=(m_{ij}), telle que :

$$m_{ij} = \begin{cases} 1 \quad Si \quad x_i \text{ } est\ l'extrémité\ initiale\ de \text{ } \mathrm{e}_j; \\ -1 \quad Si \quad x_i \text{ } est\ l'extrémité\ terminale\ de \text{ } \mathrm{e}_j; \\ 0 \quad Si \quad x_i \text{ } n'est\ pas\ une\ extrémité\ de \text{ } \mathrm{e}_j. \end{cases}$$

pour un graphe non orienté sans boucles, la matrice d'incidence (aux arêtes) est définie par :[12]

$$m_{ij} = \begin{cases} 1 \quad Si \quad x_i \text{ } est\ une\ extrémité\ de \text{ } \mathrm{e}_j; \\ 0 \quad Sinon. \end{cases}$$

Pour un graphe orienté
La matrice d'incidence du graphe de la figure 1.16 , est la suivante :

$$\begin{array}{c} \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{array} \begin{array}{c} \begin{array}{ccccc} e_1 & e_2 & e_3 & e_4 & e_5 \end{array} \\ \begin{pmatrix} +1 & +1 & 0 & 0 & +1 \\ -1 & 0 & +1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & -1 & 0 & +1 & 0 \end{pmatrix} \end{array}$$

Pour un graphe non-orienté
La matrice d'incidence du graphe de la figure 1.17, est la suivante :

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Conclusion

La Théorie des graphes aura donc occupé l'essentiel de ce chapitre, notamment pour ce qui est de la présentation des quelques notions phares et essentielles au sujet de notre travail. Définitions et concepts majeurs la concernant ont donc été avancés. Il restera que leur confrontation à la mise en pratique en sortirait avec d'autres constats autrement plus précis et édifiants. C'est ce que nous ferons au chapitre qui suit, dans l'attente de tirer conclusion.

CHAPITRE 2

OPTIMISATION DANS LES RÉSEAUX

Introduction

Il est tout à fait possible de modéliser la plupart des problèmes en utilisant des graphes valués. Cas parmi les plus anciens et importants de la Théorie des graphes, celui du cheminement dans la recherche du chemin le plus raccourci, un des plus anciens de par son application dans le domaine de l'ordonnancement, etc. Les algorithmes de recherche de la solution optimale seront différents en fonction du problème traité et des caractéristiques du graphe qui modélise la situation.

2.1 Réseau

Définition 2.1.1. [33] Un réseau est un graphe connexe valué G = (X,E,γ) dans lequel il y a un sommet s (appelé la racine) tel que d^-(s)=0 et un sommet p (appelé le puits) tel que d^+=0. Pour u $\in$ U, le nombre γ(u) est appelé la capacité de la flèche u. la figure (2.1)montre un réseau.

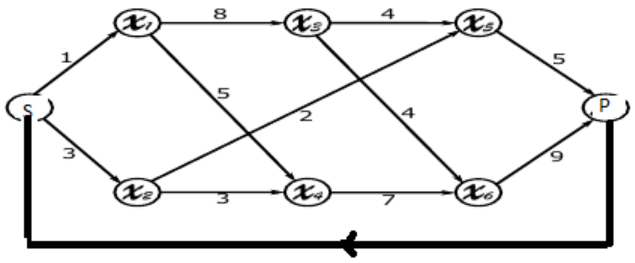


FIGURE 2.1 – Réseau

- Remarque : L'arc en gras est un arc de retour dans le cas des réseaux de transport.

2.2 Graphes Probabilistes

Définition 2.2.1. [17] Un graphe probabiliste est un graphe orienté et valué tel que la somme des coûts des arcs issus d'un sommet donné est égal à 1. La matrice de transition d'un graphe probabiliste G=(X,E,γ) d'ordre n, est la matrice p=($p_{i,j}$)$\in$m$_n(R)$où :

$$p_{ij} = \begin{cases} \gamma(i,j) \quad si \quad (i,j) \in A \ \ avec \ \ \gamma(i,j) \in [0,1] \\ 0 \quad si \quad (i,j) \notin A \end{cases}$$

On considère 4 points équirépartis sur un cercle. Un Athlète saute à chaque instant, d'un point à l'un de ses voisins avec la probabilité 1/2 pour chaque voisin.

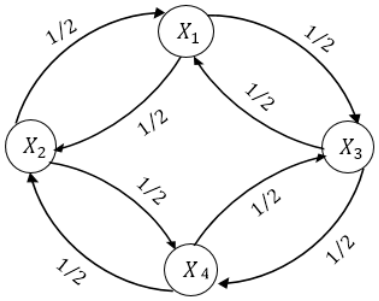


FIGURE 2.2 – Graphe probabiliste

Définition 2.2.2. La matrice de transition est une matrice carrée $T = m_{ij}$ dont le coefficient m_{ij} est la probabilité de transition du sommet j vers le sommet i.

on note un exemple : T= $\begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix}$

La matrice de transition a les propriétés suivantes :

- tous les coefficients sont compris entre 0 et 1
- pour chaque ligne, la somme des coefficients vaut 1.

La matrice de transition de la figure (2.2), est la suivante :

$$\begin{pmatrix} 0 & 1/2 & 1/2 & 0 \\ 1/2 & 0 & 0 & 1/2 \\ 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 1/2 & 0 \end{pmatrix}$$

La relation de récurrence existe entre l'état probabiliste initial P_0 et l'état probabiliste P_n à l'instant n .[17], [18], [20]

$$P_n = P_0 M^n$$

2.3 Méthodes d'optimisation dans les réseaux

Il existe plusieurs problèmes d'optimisation, conconsidérons les problèmes suivants :

2.3.1 Problème de plus court chemin

Un problème de plus court chemin est un problème algorithmique de la théorie des graphes. L'objectif est de calculer un chemin entre des sommets d'un graphe qui minimise ou maximise une certaine fonction.

Il existe de nombreuses variantes de ce problème. Etant donné un graphe connexe et fini. Un chemin le plus court entre deux sommets donnés est un chemin qui minimise la somme des valeurs des arcs traversés. Pour calculer un plus court chemin, il existe de nombreux algorithmes, selon la nature des valeurs et d'éventuelles contraintes supplémentaires. Dans de nombreux cas, il existe des algorithmes de complexité en temps polynomiale, comme l'algorithme de Dijkstra dans des graphes avec valeur positifs sur les arcs.

Définition 2.3.1. [34] Étant donné un graphe G=(X,E), on associé à chaque arc $e \in$ E le nombre *l(e)* appelé (longueur de l'arc). On dit G est valué par les longueurs *l(e)*. Si e= (i,j) on utilisera également la notation l_{ij} pour la longueur de l'arc e. Le problème du plus court chemin entre deux sommets i et j sera de trouver un chemin $\mu(i,j)$ de i à j dont la longueur totale est :

$$l(\mu) = \sum_{e \in \mu(i,j)} l(e)$$

soit minimum.
Les algorithmes de résolution seront différents suivant les propriétés du graphe :

- $l(e) \geq 0. \forall e \in E$,
- $l(e) = 1. \forall e \in E$,
- G et l(e) quelconques,
- G sans-circuit,et suivant le problème considéré :
- Recherche du plus court chemin d'un sommet à un autre,
- Recherche du plus court chemin d'un sommet à tous les autres,
- Recherche du plus court chemin entre tous les couples de sommets

•Algorithme de Dijkstra

On applique cet algorithme pour déterminer une arborescence des plus courtes distances sur un réseau R=(X,E,d) où les longueurs des arcs sont positives ou nulles (d(u)$\geq 0 \ \forall$ u $\in$ U)

$\Rightarrow$ **Le principe** : L'idée de l'algorithme de Dijkstra est de calculer de proche en proche, l'arborescence des plus courtes distances, issue du sommet **s** à un sommet donné **p**.

Une particularité de cet algorithme est que les distances s'introduisent dans l'ordre croissant.

$\Rightarrow$ **Énoncé :**

Données : Un réseau $R = (X, E, d) \quad avec \quad d(u) \geq 0 \quad \forall u \in U$

Résultat : Arborescence des plus courtes distances A.

0) Initialisation :

Soit s un sommet de X.

On pose : S = $\{s\}$, π(s) = 0, π(x) = $\infty \ \forall$ x $\in$ X/ $\{s\}$, A(s) = $\varnothing$ et α=s α est le dernier sommet introduit dans S

1) Examiner tous les arcs u dont l'extrémité initiale est égale à $\alpha(I(u) = \alpha)$ et l'extrémité terminale n'appartient pas à $S(T(u) = y \ avec \ y \notin S)$

2) Choisir un sommet z $\notin S$ tel que $\pi(z) = Min\{\pi(y)/y \notin S\}$

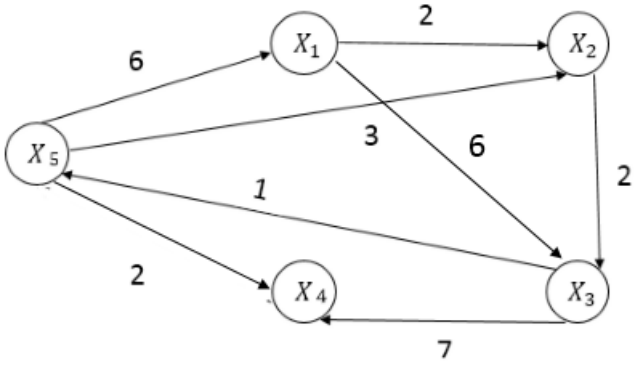
• $\pi(z) = \infty$: Terminer. Le sommet s n'est pas une racine dans R.

• $\pi(z) < \infty$: on pose $\alpha = z$ et $S = S \cup \{\alpha\}$

• $S = X$: Terminer. A définit l'arborescence des plus courts chemins issus de s.

• Si $S \neq$ X aller en (1).[12]

Soit le réseau R = (X,U,d) de la figure suivante :



⇒ **Indication :**

â signifie que a est un élément de S.

Initialisation :

Soit a un sommet de X.

Initialement on choisit un sommet à marquer x_1

On pose : $S = \{x_1\}, A(x_1) = \varnothing, \alpha = x_1$

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	∞	∞	∞	∞

⇒ **Itération 1 :**

Calculons le predecesseur du sommet x_1

On choisit le sommet à selectionne de telle sorte que le min de $\pi(x_2)$, on le calcule :

On examine les arcs (x_1,x_2) et (x_1,x_3).

$\pi(x_1) + d(x_1, x_2) = 0 + 2 = 2 < \pi(x_2) = \infty \Rightarrow \pi(x_2) = 2 : A(x_2) = (x_1, x_2)$

$\pi(x_1) + d(x_1, x_3) = 0 + 6 = 6 < \pi(x_3) = \infty \Rightarrow \pi(x_3) = 6 : A(x_3) = (x_1, x_3)$

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	2	6	∞	∞

On a $A = X/x_1 =$ l'ensemble de touts les sommets non marqués en enlevant le sommet marqué

$\pi(z) = \min\{\pi(y)/y \notin S\} = \min\{\pi(x_2), \pi(x_3), \pi(x_4), \pi(x_5)\}$

$= \min\{2, 6, \infty, \infty\} = 2 = \pi(x_2)$: Donc z = x_2

On pose alors : $\alpha = x_2 : S = \{z, x_2\} \neq X : A = \{(x_1, x_2)\}$

⇒ **Itération 2 :**

On examine l'arc (x_2, x_5)

$\pi(x_2) + d(x_2, x_5) = 2 + 2 = 4 < \pi(x_5) = \infty \Rightarrow \pi(x_5) = 4 : A(x_5) = (x_2, x_5)$

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	$\widehat{2}$	6	∞	4

$\pi(z) = \min\{\pi(y)/y \notin S\} = \min\{\pi(x_3), \pi(x_4), \pi(x_5)\}$

$=\min\{6, \infty, 4\} = 4 = \pi(x_5)$: Donc z = x_5

On pose alors : $\alpha = x_5; S = \{x_1, x_2, x_5\} \neq X : A = \{(x_1, x_2), (x_2, x_5)\}$

⇒ **Itération 3 :**

On examine les arcs (x_5, x_3) et (x_5, x_4).

$\pi(x_5) + d(x_5, x_3) = 4 + 1 = 5 < \pi(x_3) = \infty \Rightarrow \pi(x_3) = 5 : A(x_3) = (x_5, x_3)$

$\pi(x_2) + d(x_5, x_4) = 4 + 7 = 11 < \pi(x_4) = \infty \Rightarrow \pi(x_4) = 11 : A(x_4) = (x_5, x_4)$

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	$\widehat{2}$	5	11	$\widehat{4}$

$\pi(z) = \min\{\pi(y)/y \notin S\} = \min\{\pi(x_3), \pi(x_4)\}$

$=\min\{5, 11\} = 5 = \pi(e)$: Donc z $= x_3$

On pose alors : $\alpha = x_3 : S = \{x_1, x_2, x_5, x_3\} \neq X : A = \{(x_1, x_2), (x_2, x_5)(x_5, x_3)\}$

⇒ **Itération 4 :**

On examine l'arc (x_3,x_4)

$\pi(x_4) + d(x_3, x_4) = 5 + 2 = 7 < \pi(x_4) = 11 \Rightarrow \pi(x_4) = 7 : A(x_4) = (x_3, x_4)$

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	$\widehat{2}$	$\widehat{5}$	7	$\widehat{4}$

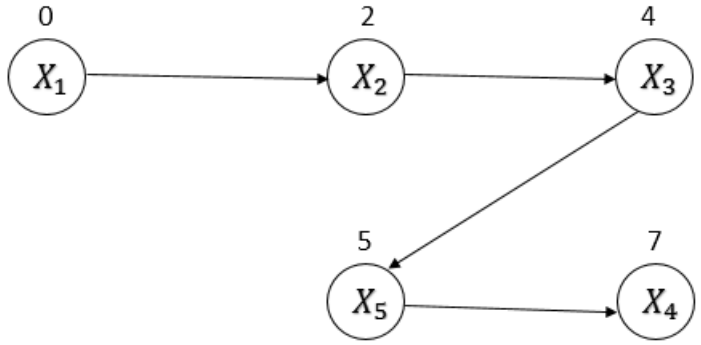
$\pi(z) = \min\{\pi(y)/y \notin S\} = \min\{\pi(x_4)\} = \pi(x_4) = 7$: Donc z$= x_4$

On pose alors : $\alpha = x_4 : S = \{x_1, x_2, x_5, x_3, x_4\} \neq X : A = \{(x_1, x_2), (x_2, x_5)(x_5, x_3), (x_3, x_4)\}$

Enfin : S = X Terminer.

x	x_1	x_2	x_3	x_4	x_5
$\pi(x)$	$\widehat{0}$	$\widehat{2}$	$\widehat{5}$	$\widehat{7}$	$\widehat{4}$

Les π(X) sont les plus courtes distances. L'arborescence des plus courts chemins, issue du sommet X_1 dans le réseau est la suivante :



•Algorithme de Bellman :

On applique cet algorithme pour la recherche d'une arborescence des plus courts chemins dans un réseau R=(X,E,d) sans circuit.

⇒ **Le principe :**

L'idée de l'algorithme de Bellman, est de calculer de proche en proche , l'arborescence des plus courtes distances, issue du sommet s à un sommet donné p.

On calcule la plus courtes distances du sommet s à y, que si on a déjà calculé les plus courtes distances du sommet s à tous les prédécesseur du sommet y.

⇒ **Enoncé :**

Données : Un réseau R=(X,U,d) sans circuit absorbant avec d(u)∈R

Résultat : Arborescence des plus courtes distances A.

0)Initialisation :

Soit s un sommet de X,on pose $S = \{s\}$ et $\pi(s) = 0 \quad A = \phi$.

1) Chercher un sommet hors de S dont tous les prédécesseurs sont dans S.

• Si un tel sommet n'existe pas ; terminer

Dans ce cas soit S=X, ou le sommet s n'est pas une racine dans R.

• Si un tel sommet existe : Aller en(2)

2) On pose : $\pi(x) = Min\{\pi(I(u)) + d(u)\}$.

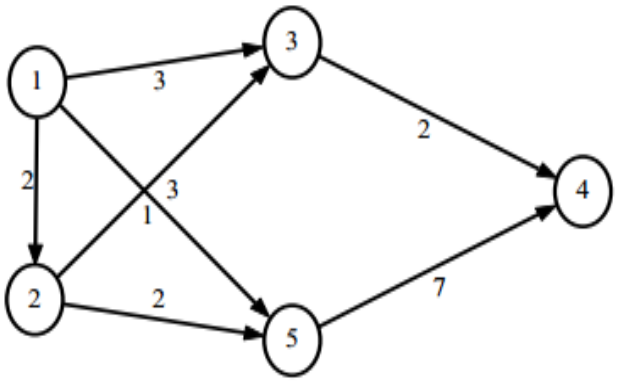
- $I(u)$: Sommet initial de l'arc u.

- $d(u)$: poids de l'arc u.

soit u l'arc pour lequel $\pi(x) = \pi(I(u)) + d(u)$

A : = A $\cup\{u\}; S \cup \{x\}$ Aller à (1).[12]

Soit le réseau R=(X,E,d) de la figure suivante :



⇒ **Initialisation :**

Soit 1 un sommet de X. On pose S=$\{1\}$; $\pi(1)$=0 ; A= ϕ

⇒ **Itération 1 :**

Le sommet 2 est hors de S et tous leurs prédécesseurs sont dans S, alors

$\pi(2)$= $\pi(1)$+ d(1,2)= 0+2=2 ⇒ u=(1,2)

S = S ∪ $\{2\}$= $\{1,2\}$; A ∪ $\{(1,2)\}$=$\{(1,2)\}$

⇒**Itération 2 :**

Les sommets 3 et 5 sont hors de S et tout leurs prédécesseurs sont dans S, alors

$\pi(3) = Min\{\pi(1) + d(1,3); \pi(2) + d(2,3)\}$

$= Min\{0+3; 2+3\} = Min\{3;5\} = 3 \quad \Rightarrow u = (1,3)$

$\pi(5) = Min\{\pi(1 + d(1,5); \pi(2) + d(2,5)\}$

$= Min\{0+1; 2+2\} = Min\{1;4\} = 1 \quad \Rightarrow u = (1,5)$

$S = S \cup \{3,5\} = \{1,2,3,5\}; A = A \cup \{(1,3),(1,5)\} = \{(1,2),(1,3),(1,5)\}$

⇒ **Itération 3 :**

Le sommet 4 est hors de S et tous ses prédécesseurs sont dans S

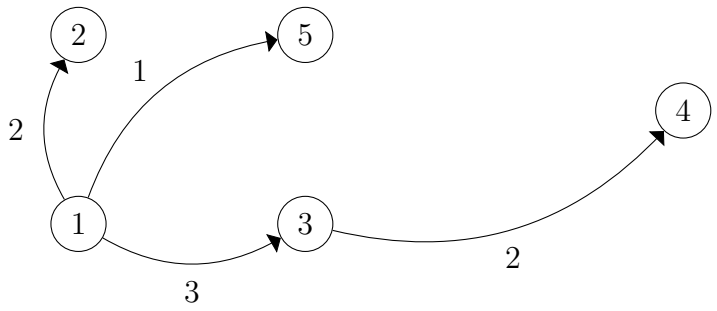
$\pi(4) = Min\{\pi(3) + d(3,4); \pi(5) + d(5,4)\}$

$= Min\{3+2; 1+7\} = Min\{5;8\} = 5 \quad \Rightarrow u = (3,5)$

$S = S \cup \{4\} = \{1,2,3,4,5\}; A = A \cup \{(3,4)\} = \{(1,2),(1,3),(1,5)(3,4)\}$

On a : $|S| = |X|$; Terminer.

On obtient donc, l'arborescence de la figure suivante :



•Algorithme de Bellman Ford

On applique l'algorithme de Bellman Ford pour la recherche d'un plus court chemin sur un réseau quelconque avec $\mathbf{d(e)\in R}$.

Cet algorithme permet soit :

- De mettre en évidence un circuit absorbant si celui-ci existe.
- De déterminer une arborescence des plus courts chemins de racine s dans un réseau s'il ne contient pas de circuit absorbant.

⇒ **Principe :**

le principe général de l'algorithme consiste à améliorer une arborescence réalisable (initial) (X,A) de racine s jusqu'à l'obtention d'un arborescence optimal des plus courts chemins, issue de s si celle-ci existe.

⇒ **Enoncé :**

Données : Un réseau R=(X,U,d) avec d(u)∈R

Résultat : Arborescence des plus court distances A, ou circuit absorbant.

0) Initialisation :

Soit (X,A) une arborescence de racine s dans le réseau R et π(x) les longueurs des chemins de s à x dans l'arborescence (X,A)

1) chercher un arc u=(i,j) dans le réseau R, n'appartenant pas à A, tel que :

δ(u)= π(j)-π(i)-d(i,j)> 0

- Si un tel arc n'existe pas ; Terminer,(X,A) est optimal.
- Si un tel arc existe ; Aller à (2)

2) Tester Si (X,A ∪{u}) contient un circuit

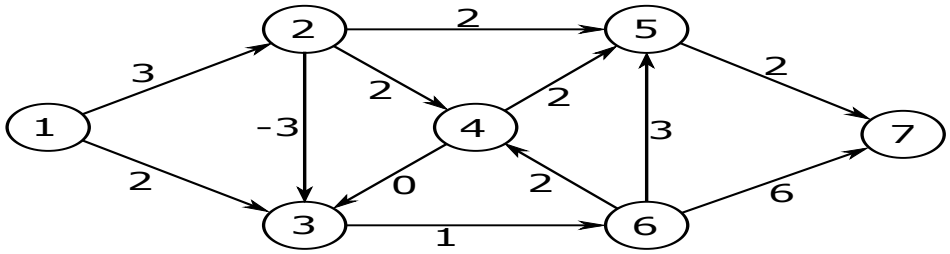
- Si oui ; examiner si ce circuit est absorbant (s'il l'est : terminer le problème n'admet pas de solution)
- Sinon ; Aller à (3)

3) Chercher un arc v∈ A tel que T(v)=j=T(u). On pose : A=A ∪{u} / {v}

Soit X'={j} ∪ { descendant de j dans l'arborescence A }

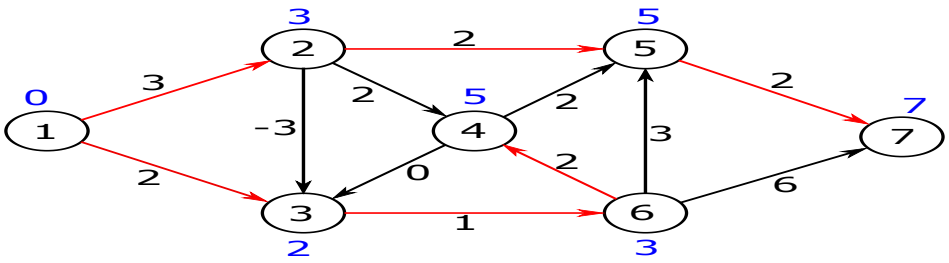
On pose : π(y)=π(y) - δ(u) ∀ y ∈ X' Aller en (1).[12]

Soit le réseau R=(X,E,d) suivant :



• **Initialisation** :

Soit A=$\{(1,2),(1,3),(3,6),(6,4),(2,5),(5,7)\}$ une arborescence initiale.



• **Itération 1 :** Dans le réseau R, les arcs n'appartenant pas à A sont :

(2,4),(2,3),(4,3),(4,5),(6,5),(6,7) tels que :

δ(2,3)=π(3)-π(2)-d(2,3)= 2-3+3 = 2

δ(2,4)=π(4)-π(2)-d(2,4)= 5-3-2 = 0

δ(4,3)=π(3)-π(4)-d(4,3)= 2-5-3 = -3

δ(4,5)=π(5)-π(4)-d(4,5)= 5-5-3 = -3

δ(6,5)=π(5)-π(6)-d(6,5)= 3-5-3 = -5

δ(6,7)=π(7)-π(6)-d(6,7)= 7-3-6 = -2

L'arc u=(2,3)$\notin$ A et δ(u)$>$ 0 et A $\cup$ $\{(2,3)\}$ ne contient pas de circuit, donc l'arc u=(2,3) entre dans l'arborescence .

Soit l'arc v=(1,3)$\in$ A, T(v)=3 donc l'arc (1,3)sort de l'arborescence .

Soit X'=$\{3\}$ $\cup$ { descendants de sommet 3 dans l'arborescence A }=$\{3,4,6\}$

On pose :

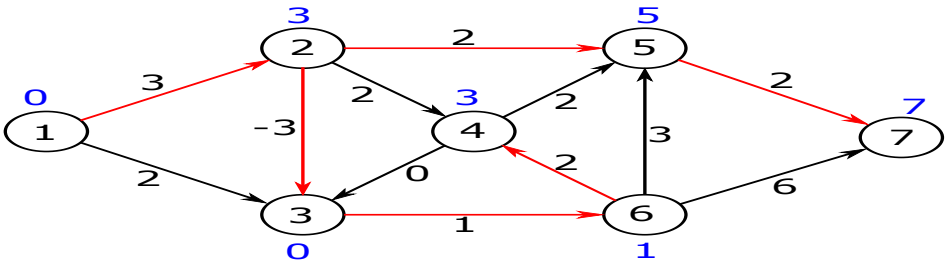
π(3)= π(3)-δ(2,3)= 2-2 = 0

π(6)= π(6)-δ(2,3)= 3-2 = 1

π(4)= π(4)-δ(2,3)= 5-2 = 3

On obtient ainsi la nouvelle arborescence

A= A $\cup$ $\{u\}/\{v\}$ = $\{(1,2),(2,5),(5,7),(2,3),(3,6),(6,4)\}$



- **Itération 2 :**

Dans le réseau R, les arcs n'appartenant pas à la nouvelle arborescence sont :

(2,4),(1,3),(4,3),(4,5),(6,5),(6,7) tels que :

δ(1,3)=π(3)-π(1)-d(1,3)= 0-0-2 = -2

δ(2,4)=π(4)-π(2)-d(2,4)= 3-3-2 = -2

δ(4,3)=π(3)-π(4)-d(4,3)= 0-3-0 = -3

δ(4,5)=π(5)-π(4)-d(4,5)= 5-3-2 = 0

δ(6,5)=π(5)-π(6)-d(6,5)= 5-1-3 = 1

δ(6,7)=π(7)-π(6)-d(6,7)= 7-1-6 = 0

L'arc u=(6,5)$\notin$ A et δ(u)$>$ 0 et A $\cup$ $\{(6,5)\}$ ne contient pas de circuit , donc l'arc u = (2,5) entre dans l'arborescence.

Soit l'arc v=(2,5)$\in$ A tel que T(v)= 5, donc l'arc (2,5) sort de l'arborescence.

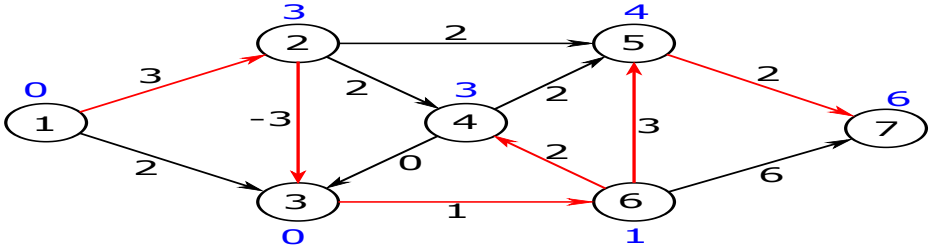
Soit X'= $\{5\}$ $\cup$ { descendants de sommet 5 dans l'arborescence A } = $\{5,7\}$

On pose : $\pi(5)$= (5) - $\delta(6,5)$ = 5-1 = 4

$\pi(7)$- δ(6,5)= 7-1 = 6

On obtient ainsi la nouvelle arborescence

A=A $\cup$ $\{u\}$ / v = A = $\{(1,2),(6,5),(5,7),(2,3),(3,6),(6,4)\}$



• **Itération 3 :**

Dans le réseau R les arcs n'appartenant pas à la nouvelle arborescence A sont :

(2,4),(1,3),(4,3),(4,5),(2,5),(6,5) tels que :

δ(1,3)=π(3)-π(1)-d(1,3)= 0-0-2 = -2

δ(2,4)=π(4)-π(2)-d(2,4)= 3-3-2 = -2

δ(4,3)=π(3)-π(4)-d(4,3)= 0-3-0 = -3

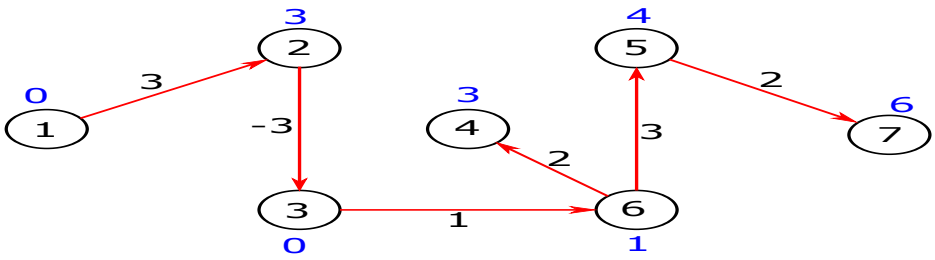
δ(4,5)=π(5)-π(4)-d(4,5)= 4-3-2 = -1

δ(2,5)=π(2)-π(2)-d(2,5)= 4-3-2 = -1

δ(6,7)=π(7)-π(6)-d(6,7)= 6-1-6 = -1

Il n'existe aucun arc u n'appartenant pas à A et δ(u)> 0.D'où l'arborescence déjà trouvé est optimale.

L'arborescence optimal est A=$\{(1,2),(6,5),(5,7),(2,3),(3,6),(6,4)\}$. Elle est montrée dans la figure ci-dessous



- **Remarque :**

La recherche d'un plus long chemin sur le réseau R=(X, E, d) revient à recherche un plus court chemin sur R=(X, E, -d).

2.3.2 Problème de l'arbre couvrant minimum

Le problème de l'arbre couvrant minimal est un des plus vieux problèmes en théorie des graphes. La problématique se pose comme suit : étant donné un graphe connexe avec un nombre de sommets et un nombre d'arêtes ayant des poids de valeurs dans l'ensemble des entiers relatifs, l'arbre couvrant minimal consiste à trouver l'ensemble des arêtes permettant de rejoindre tous les sommets sans former de cycle, et ce, avec un coût minimal. Ce problème trouve des applications pratiques variées : il est directement applicable pour l'optimisation et la conception de divers types de réseaux (électrique, internet, etc.).

Données : un graphe simple connexe G = (X,E) valué par l'application d à valeur dans les réels strictement positifs, que peuvent représenter des coûts, des distance, des temps, etc.

Sortie : un graphe partiel de G, T = (X,A) où A $\subset$ E, tel que d(T) =$\sum_{e \in T} d(e)$ soit minimum.

Définition :[22] Un graphe simple valué est un triplet (X,E,d) où G = (X,E) est un graphe simple et d : E $\longrightarrow$ [0,∞[est une fonction appelée la fonction de cout.

Proposition

Un graphe admet un arbre couvrant si, et seulement si, il est connexe.[32]

•Algorithme de kruskal :

Le principe :

L'idée de l'algorithme de kruskal est tout d'abord de numéroter les arcs par ordre des poids croisant. Ensuite de construire progressivement l'arbre A en rajoutant dans leurs ordre, les arcs un par un. Un arc est rajouté seulement si son adjonction à A ne détermine pas de cycle, c'est-à-dire si A ne perd pas sa notion d'arbre, sinon on passe à l'arc suivant dans l'ordre de la numérotation.

Enoncé :

0) Initialisation : numéroter les arcs de G dans l'ordre des poids croissants :

c(e_1) $\leq$ c(e_2) $\leq$ ...$\leq$ c(e_m). soit E = $\varnothing$; i=1.

1) Si (X,E $\cup$ { e_1 }) contient un cycle aller en (3) sinon aller en (2)

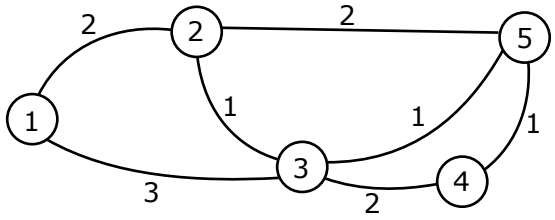
2) on pose E= E $\cup$ { e_1 } aller en (3)

3) Si i=m terminé, A=(X,E) est l'arbre de poids minimum c(E)= $\sum_{e_i \in E} c(e_i)$ pour $e_1 \in$ E sinon i := i+1 aller en (1)

L'algorithme s'arrête lorsque le nombre d'arcs retenus égal à n-1.[32]

Soit G=(X,E) un graphe connexe représentant le projet d'installation de lignes téléphonique. Les poids représentent le coût d'installation des lignes. On veut donner un plan d'installation minimisant le coût total de l'installation. comme s'est montré dans la figure ci-dessous.

Initialisation



On ordonne les arêtes du graphe selon les poids croissants :

i	1	2	3	4	5	6	7
e_i	(2,3)	(3,5)	(4,5)	(3,4)	(2,5)	(1,2)	(1,3)
$c(e_i)$	1	1	1	2	2	2	3

Soit E = $\varnothing$; i=1 ; m=7

Itération 1 :

On a choisit l'arête e_1 = (2,3) de poid minimal ; Soit E = E $\cup$ { e_1 } = $\{(2,3)\}$ le graphe (X,E) ci contre ne contient pas de cycle, on pose alors E=$\{(2,3)\}$.

$\| E \| = 1$

On a : i $\neq$ m alors on pose i=i+1=2.

Itération 2 :

On a :e_2 = (3,5) ;

Soit E = E $\cup$ { e_2 } = $\{(2,3),(3,5)\}$ le graphe (X,E)ci contre ne contient pas de cycle, on pose alors E = $\{(2,3),(3,5)\}$.

$\| E \| = 2$

On a : i $\neq$ m alors on pose i=i+1=3.

Itération 3 :

On a :e_3 = (4,5) ;

Soit E = E $\cup$ { e_3 } = $\{(2,3),(3,5),(4,5)\}$ le graphe (X,E)ci contre ne contient pas de cycle, on pose alors E = $\{(2,3),(3,5),(4,5)\}$.

$\| E \| = 3$

On a : i $\neq$ m alors on pose i=i+1=4.

Itération 4 :

On a :e_4 = (3,4) ;

Soit E = E ∪ { e_4 } = $\{(2,3),(3,5),(4,5),(3,4)\}$ le graphe (X,E)ci contre contient un cycle.

On a : i ≠ m alors on pose i=i+1=5.

Itération 5 :

On a :e_5 = (2,5) ;

Soit E = E ∪ { e_5 } = $\{(2,3),(3,5),(4,5),(2,5)\}$ le graphe (X,E)ci contre contient un cycle.

On a : i ≠ m alors on pose i=i+1=6.

E=$(2,3),(3,5),(3,4)$

Itération 6 :

On a :e_6 = (1,2) ;

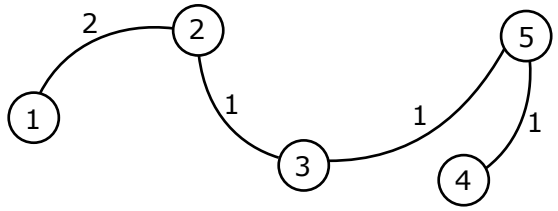
Soit E = E ∪ { e_5 } = $\{(2,3),(3,5),(4,5),(1,2)\}$ le graphe (X,E)ci contre ne contient pas de cycle, on pose alors E = $\{(2,3),(3,5),(4,5),(1,2)\}$.

‖ E ‖ = n-1, Terminé

L'arbre de poids minimum est représenté par le graphe A=(X,E).

Le coût total de toute l'installation est :

c(W) = c(e $_1$)+c(e $_2$)+c(e $_3$)+c(e $_6$) = 1+1+1+2 = 5. d'aprés ce processus on n'en obtient le resultat final un arbre de pods minimum comme s'est indiqué dans la figure qui suit :



Remarques :

- L'arbre couvrant de poids minimal n'est pas forcément unique.
- Un arbre couvrant de poids minimum est unique si et seulement si les poids de ces arêtes sont deux à deux distincts.
- La solution du problème de recherche de l'arbre de poids maximum s'obtient en utilisant le même algorithme de l'arbre de poids minimum après avoir multiplié les poids des arcs par (-1).

2.3.3 Problème d'ordonnancement

Les problèmes d'ordonnancement se rencontrent dans différents domaines, dès lors qu'il est nécessaire d'organiser et/ou de répartir le travail entre plusieurs entités. Plusieurs définitions ont été proposées pour le problème d'ordonnancement on en cite :

«Ordonnancer, c'est programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécutions. Les différentes données d'un problème d'ordonnancement sont les tâches, les contraintes potentielles, les ressources et la fonction économique. »

« Ordonnancement concerne l'affectation de ressources limitées aux tâches dans le temps, c'est un processus de prise de décision dont le but est d'optimiser un ou plusieurs objectifs ».

Eric pinson,le définit comme étant « L'organisation dans le temps de l'exécution d'un projet ». Pour Norman Sadeh, c'est « l'allocation dans le temps de ressources permettant l'exécution d'un ensemble de tâches ». Pour Gotha, c'est « programmer des exécutions des tâches en leur allouant les ressources requises et en fixant leur dates de début ».

Bref, à partir de toutes ces définitions, on peut dire que l'ordonnancement, décrit l'exécution de tâches et l'affectation de ressources au cours du temps compte tenu de contraintes et demanière à satisfaire des objectifs.[35]

L'ordonnancement par la méthode PERT

un graphe de PERT est un graphe caractérisé par des sommets (étapes) reliés entre eux par des arcs orientés (tâches), chaque arc étant défini par son début, sa fin et sa durée et les sommets entre les arcs définissant les relations d'antériorité. Ce graphe ne comportera ni retour ni circuit et on ne rencontrera qu'un seul arc entre deux sommets.

Ainsi, Pour élaborer et exploiter un réseau PERT, on peut distinguer Six grandes étapes :

1. Etablir la liste des tâches ;
2. Déterminer les conditions d'antériorité ;
3. Tracer le réseau PERT ;
4. Calculer les dates des tâches et déterminer le chemin critique ;
5. Calculer les marges totales de chaque tâche ;
6. Construire le planning du projet.[35]

Les taches :

Une tâches est l'activité élémentaire caractérisée par :

— Sa durée d_i

— Sa date de début t_i et sa date de fin f_i. La date de fin est égale à $t_i + d_i$.

— Éventuellement les ressources nécessaires a son accomplissement.

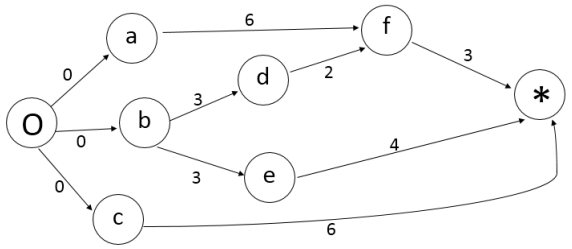
Les tâches sont supposées non-préemptives : une fois que l'exécution a commencé, elles se poursuit sans interruption jusqu'à la fin. Il existe entre les tâches d'un projet des relations de précédence, qui signifient par exemple qu'une tâche ne peut démarrer que si certaines autres tâches sont finies.[27], [11], [6]

- la représentation du réseau PERT

La méthode PERT est une technique de gestion de projet qui permet de visualiser la dépendance des tâches et de procéder à leur ordonnancement ; c'est un outil de planification. On utilise un graphe de dépendances. Pour chaque tâche, on indique une date de début et de fin au plus tôt et au plus tard. Le diagramme permet de déterminer le chemin critique qui conditionne la durée minimale du projet. Son but est de trouver la meilleure organisation possible pour qu'un projet soit terminé dans les meilleurs délais, et d'identifier les tâches critiques, c'est-à-dire les tâches qui ne doivent souffrir d'aucun retard sous peine de retarder l'ensemble du projet.

- Représentation par un graphe potentiels-tâches

On peut représenter un problème d'ordonnancement simple par un graphe oriente $G = (X, A)$:
- Chacun des sommets de X représente une tâche.
- Les arcs A représentent les contraintes de potentiels (de précédence en général) :
$A = (i, j) \in X * X, \exists$ une contrainte $t_j - t_i \geqslant a_{ij}$
Le poids associe à un arc (i, j) est $p_{ij} = a_{ij}$
- on a ajouté à cela deux sommets O et $*$ représentant les tâches fictives début et fin de projet.
Le graphe potentiels-tâches du problème d'ordonnancement est donné ci-dessous :



- Recherche d'ordonnancement réalisable

METHODOLOGIE :

Définition du problème

↓

(Division en taches. (Travaux homogènes)

↓

Fixation de la durée de chaque tâche. (Calendrier)

↓

Détermination de l'ordre chronologique

↓

(Tableau des tâches antériorité)

↓

Primarisation des niveaux

↓

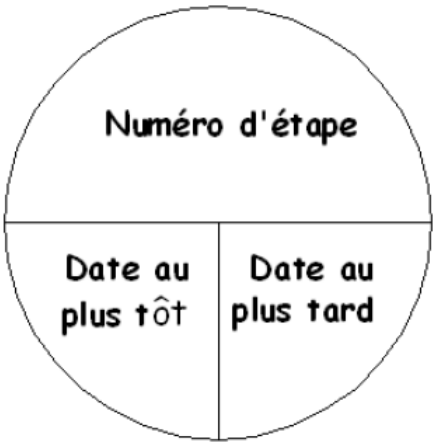
Montage du réseau

↓

Détermination du chemin critique.

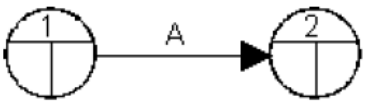
a : Les étapes :

On appelle étape, le début ou la fin d'une tâche. Habituellementon numérote les étapes de telle façon que le numéro du sommet au début d'un arc soit inférieur au numéro du sommet en fin de l'arc. On indique aussi leur temps de réalisation au plus tôt et au plus tard.

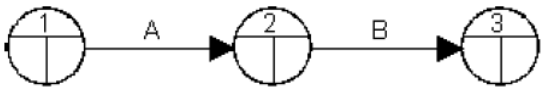


b : Présentations et règles

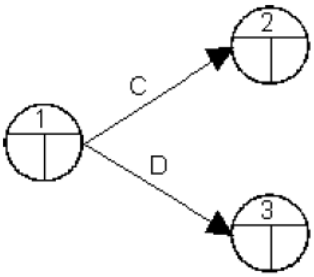
-Toute tâche a une étape de début et une étape de fin, Une tâche suivante ne peut démarrer que si la tâche précédente est terminée.



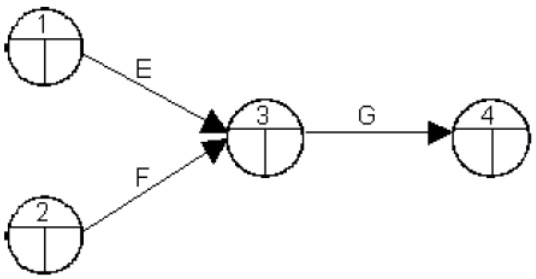
-Deux tâches qui se succèdent immédiatement sont représentées par des flèches qui se suivent.



-Deux tâches C et D qui sont simultanées (c'est à dire qui commencent en même temps) sont représentées de la manière suivante :



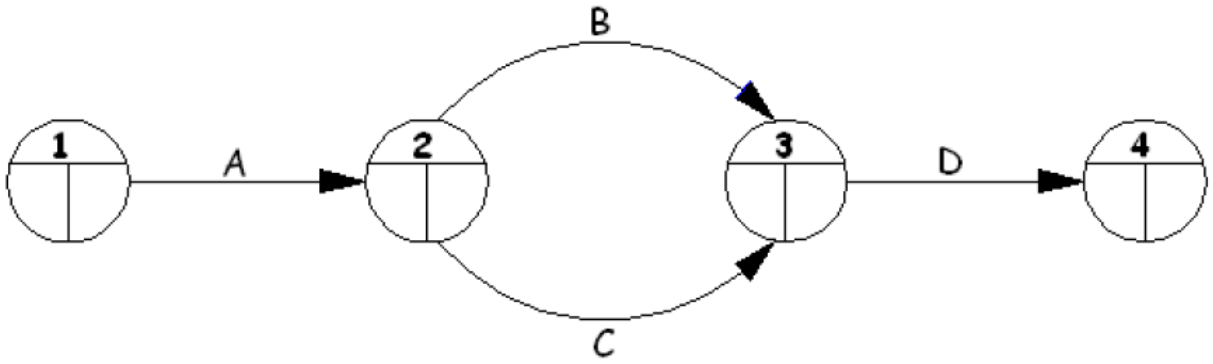
-Deux tâches E et F qui sont convergentes (c'est à dire qui précèdent une même tâches G) sont représentées de la manière suivante :



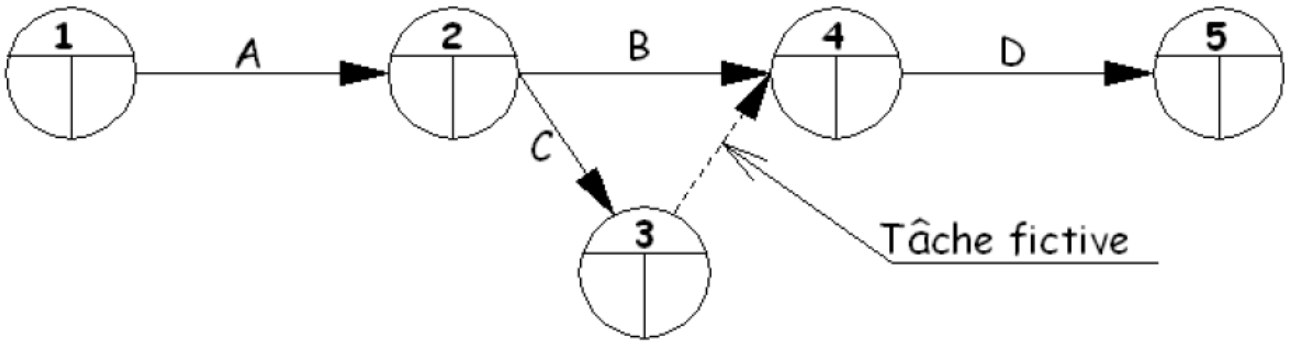
-Parfois, il est nécessaire d'introduire des tâches fictives (Trait en pointillé). Une tâche fictive a une durée nulle. Elle ne modifie pas le délai final.

-1er cas : deux opérations sont parallèles

Les tâches B et C suivent la tâche A et précèdent la tâche D. La représentation ci-dessous est fausse :



Il faudra lui préférer celle-ci :

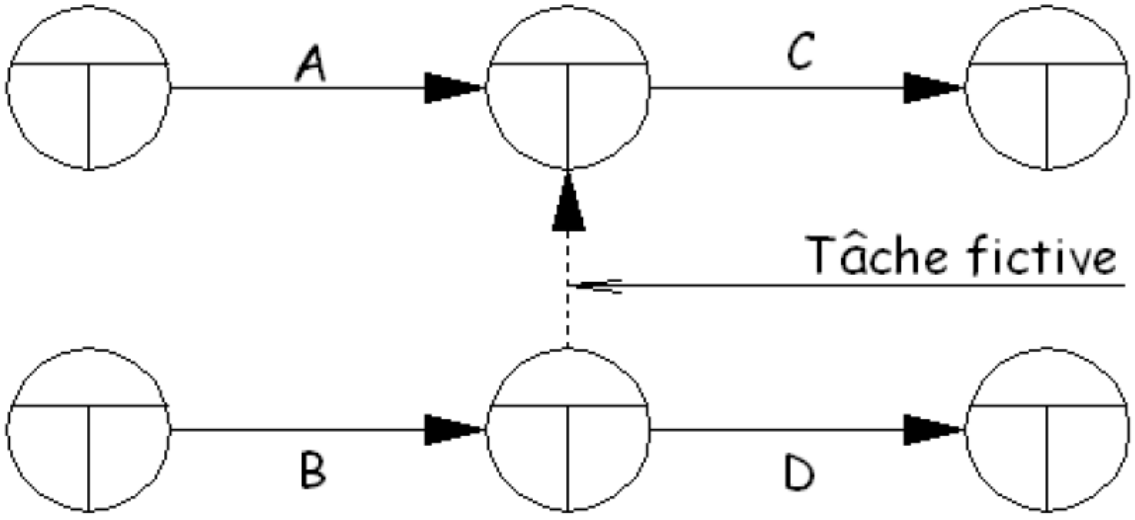


tâche fictive :est une tâche supplémentaire ne dépend aucun temps, elle ne modifie pas le délai final.

-2ème cas : des opérations sont dépendantes et indépendantes

Exemple : la tâche C succède à A et à B, la tâche D succède à B mais est indépendante de A.

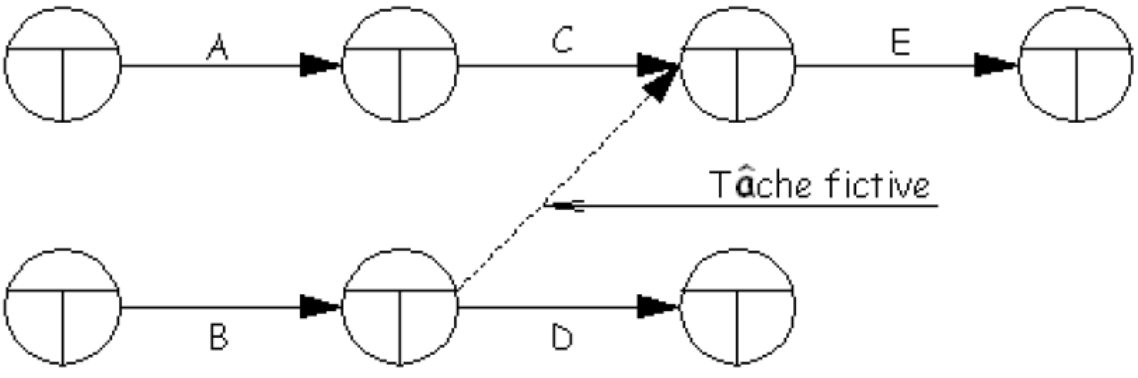
Il faudra alors représenter ces relations ainsi :



-3ème cas : un délai doit être respecté entre deux tâches

Exemple : la tâche A est suivie de la tâche C, la tâche B est suivie de la tâche D, la tâche E suit la tâche C et ne peut commencer qu'après un certain délai après la fin de la tâche B

Il faudra alors représenter ces relations ainsi :



C : Classement des activités par niveaux

Le **niveau d'une tâche** correspond au plus grand nombre de tâches rencontrées sur un même itinéraire depuis le début du projet, plus un. Pour déterminer le niveau des tâches, on procède comme suit. On place au premier niveau les tâches qui n'ont aucun ancêtre et on raye ces tâches de la liste des tâches. On continue comme suit :

. Etape 1 : on raye, dans la colonne des ancêtres, les tâches qui viennent d'être affectées au dernier niveau analysé ;

. Etape 2 : les tâches du nouveau niveau sont les tâches non rayées de la colonne des tâches qui n'ont plus d'ancêtre ; après affectation au nouveau niveau, ces tâches sont rayées dans la colonne des tâches ;

. Etape 3 : s'il reste des tâches non rayées dans la colonne des tâches, on repart à l'étape 1. Sinon le processus est terminé.

D : Calcul des dates des tâches

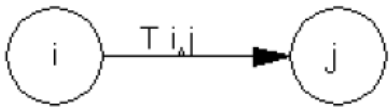
Dans la méthode PERT, on calcule deux valeurs pour chaque étape :

.La date au plus tôt : il s'agit de la date à laquelle la tâche pourra être commencée au plus tôt, en tenant compte du temps nécessaire à l'exécution des tâches précédentes.

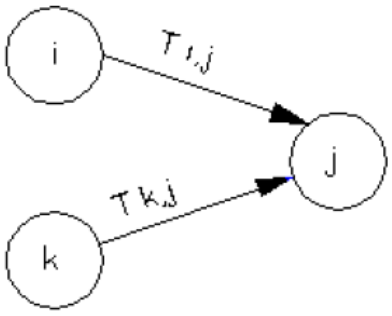
.La date au plus tard : il s'agit de la date à laquelle une tâche doit être commencée à tout prix si l'on ne veut pas retarder l'ensemble du projet.

Pour déterminer la date au plus tôt d'une tâche, il faut parcourir le diagramme de gauche à droite et calculer le temps du plus long chemains avec l'algorithme de Ford pour le calcul des chemins optimaux).

-Soit il n'y a qu'une seule tâche (un seul chemin) entre Deux étapes alors date au plus tôt j = date au plus tôt i + durée tâche $T_{i,j}$. La figure suivante montre ce cas



- Soit il y a plusieurs chemins pour aboutir à l'étape j alors date au plus tôt j = max ((date au + tôt i + durée $T_{i,j}$) ; (date au plus tôt k + durée $T_{k,j}$)). La figure suivante montre ce cas



Pour déterminer la date au plus tard d'une tâche, il faut parcourir le diagramme de droite à gauche, et soustraire de la date au plus tard de la tâche suivante la durée de la tâche dont on calcule la date au plus tard. S'il y a plusieurs sous-chemins, on effectue le même calcul pour chacun et on choisit la date la plus petite. -Soit Un seul arc sort du sommet i alors date au + tard j = date au plus tard j - durée $T_{i,j}$. - Soit il y a plusieurs arcs qui sortent de l'étape i alors date au plus tard i = min ((date au + tard j - durée $T_{i,j}$) ; (date au + tard k - durée $T_{i,k}$)).

E : Identification du chemin critique et des marges

Les tâches possédants une date au plus tôt égale à leur date au plus tard font partie du chemin critique, c'est-à-dire le chemin sur lequel aucune tâche ne doit avoir de retard pour ne pas retarder l'ensemble du projet.
Alors **le Chemin critique** représente un ensemble de tâches qui doivent être achevées selon les prévisions afin que le projet soit terminé à temps, chaque tâche du chemin critique est une tâche critique. Une fois ces valeurs définies, on en déduira :
- Le chemin critique du projet : Suite des tâches du réseau ne comportant aucune marge (date au plus tôt = date au plus tard).
La durée totale des tâches critiques donne la durée minimale de réalisation du projet. Le moindre retard au démarrage de l'une de ces tâches entraîne un retard équivalent sur la date de fin du projet.

- Les tâches à marge : Tâches disposant d'un battement possible dans le temps (date au plus tôt < date au plus tard). Nous présentons dans notre étude deux types de marge :

- **La marge totale** : Retard maximum que l'on peut prendre pour débuter une tâche sans remettre en cause les dates au plus tard des tâches suivantes. Elle est égale à la différence entre la date au plus tôt et la date au plus tard d'une tâche.

- **La marge libre** : Retard maximum que l'on peut prendre pour débuter une tâche sans remettre en cause les dates au plus tôt des tâches suivantes. Elle est égale à la différence entre :

— La plus petite date au plus tôt des tâches suivantes

— La date au plus tôt de la tâche dont on calcule la marge à laquelleon rajoute sa durée

Conclusion

Ce chapitre a été consacré aux méthodes de résolution de différents problèmes d'optimisation dans les réseaux, étayés de quelques exemples pour mieux comprendre leur résolution.

CHAPITRE 3

RÉSOLUTION DE QUELQUES PROBLÈMES CONCRETS

Introduction

Dans ce chapitre, nous allons présenter quelques situations concrètes que nous allons modéliser sous forme de graphe afin d'optimiser certains paramètres avec des algorithmes cités précédemment.

3.1 Résolution d'un Problème de transport

Les sept pays, les Pays-Bas, la Belgique, l'Allemagne, la Suisse, l'Italie, l'Espagne et la France, produisent, exportent, importent et consomment le beurre. Certains de ces pays surproduisent et sont obligé d'exporter, d'autres doivent importer. Le gouvernement de chaque pays décide de créer un organisme dont le rôle est d'aider les échanges. Pour cela, dans un premier temps, chaque pays passe un accord commercial avec chacun de ses voisins fixans des aides pour l'exportation vers des pays déficitaires et des taxes pour l'exportation vers des pays surproducteurs.
Les données sont résumées dans le tableau 3.1. Les nombres dans chaque case du tableau représentent les taxes, les aides et les coût de transport sous forme :

taxe / aide / coût

	PB	B	A	S	I	F	E
PB		0/15/2	0/5/4				
B	15/0/2		8/03			22/0/2	
A	5/0/4	0/10/3		5/0/3		15/0/4	
S			0/5/3		0/15/2	5/0/5	
I				15/0/2		8/0/8	
F		0/10/2	0/15/4	0/5/5	0/8/6		0/5/6
E						5/0/6	

TABLE 3.1 – Taxes , aides et coûts de transport pour exportation du beurre

modélisation du problème :

Les dépenses pour acheminer le beurre d'un pays à un autre se calcule par la formule :

taxe - aide + coût

Nous obtenons donc le graphe de la figure 3.1 qui montre l'éxportation du beurre.

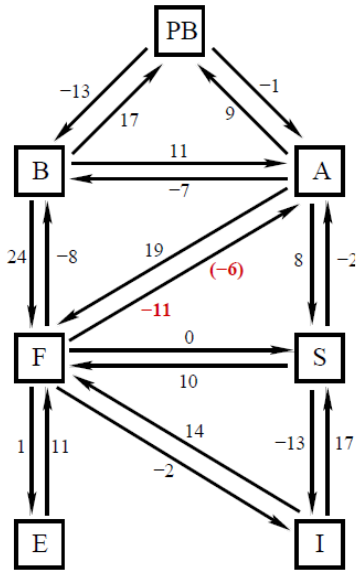


FIGURE 3.1 – Exportation du beurre

Les sigles signifient PB = Pays - Bas, B = Belgique, A = Allemagne, F = France, S =Suisse, E = Espagne,I = Italie.

1. On remarque que ce graphe contient un circuit absorbant de valeur -2 :

$$A \longrightarrow^{8} S \longrightarrow^{-13} I \longrightarrow^{14} F \longrightarrow^{-11} A.$$

Ainsi, le problème d'un plus court chemin n'a pas de solution : en circulat par ce circuit on peut avoir du beurre et de l'argent du beurre. 2. En réduisant de 15 à 10 l'aide à l'exportation de la France vers l'Allemagne on rend le poids du traget $F \longrightarrow A$ égale à -6 au lieu de -11, ce qui détruit le circuit absorabant. 3. La table 2 montre le dérulement de l'algorithme de Ford-Bellmann. La 7ème étape est l'étape de contrôle : elle montre qu'il n'y a pas de circuits absorbant. Finalement, le plus court chemin des Pays-Bas vers l'Espagne est le suivant :

$$PB \longrightarrow^{-13} B \longrightarrow^{11} A \longrightarrow^{8} S \longrightarrow^{-13} I \longrightarrow^{14} F \longrightarrow^{1} E.$$

La valeur de ce chemin est égale à 8. Aussi, ce chemin représente l'arborescence des plus courts chemins. Voici les résultats finals obtenus dans la table 3.2

étape	PB	B	A	S	I	F	E
0	0	∞	∞	∞	∞	∞	∞
1	0	-13	-1	∞	∞	∞	∞
2	0	-13	-2	7	∞	11	∞
3	0	-13	-2	6	-6	11	12
4	0	-13	-2	6	-7	8	12
5	0	-13	-2	6	-7	7	9
6	0	-13	-2	6	-7	7	8
7	0	-13	-2	6	-7	7	8

TABLE 3.2 – Exécution de l'Algorirthme de Bellman-Ford

3.2 Résolution d'un problème d'ordonnancement

L'étude et la réalisation d'un projet exigent un grand nombre de travaux de natures très diverses, faisant intervenir un grand nombre de participants. De plus, les tâches des uns et des autres sont le plus souvent liées, voire conditionnées les unes par les autres.
Il est donc impérativement nécessaire d'ordonner les actions de chacun et de matérialiser dans un langage approprié les décisions prises et les conséquences qui en découlent. Elaborés en phase préparation du travail, les « plannings » sont des documents essentiels pour la coordination ultérieure et pour la gestion des projets. Dans la plupart des cas,comme l'indique la table ci-après, un projet de construction doit être réalisé dans un délai déterminé par le maître d'ouvrage en accord avec le maître d'oeuvre.

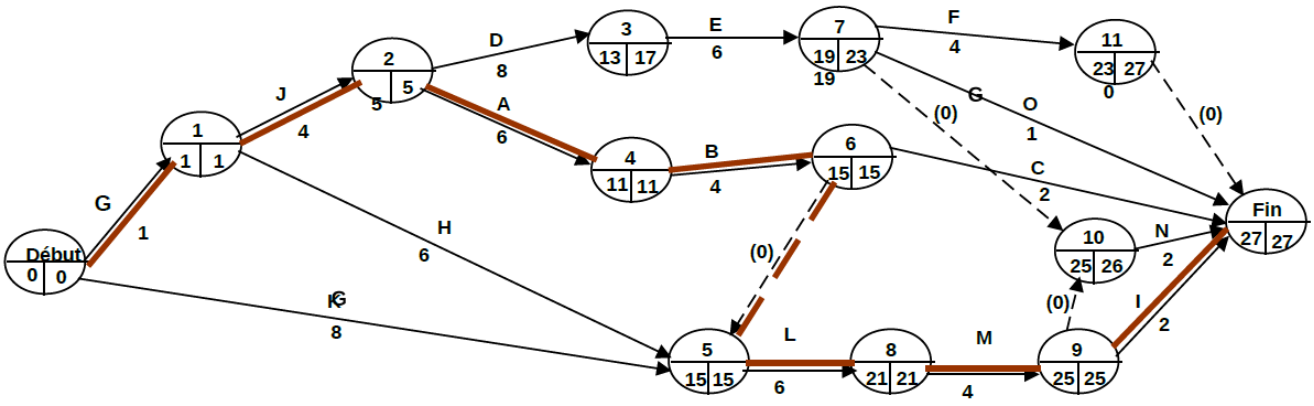
Rubrique	Tâches	Description	Durée	Tâches précédentes
	Début	Lancement du projet	0	-
	A	Dépose ancien carrelage	6	J
	B	Pose carrelage	4	A
	C	Joints carrelage	2	B
Murs	D	Décollage ancien papier	8	J
	E	Pose faïence	6	D
	F	Pose nouveau papier ancien papier	4	E
Plomberie	G	Dépose ancien évier	1	Début
	H	Déplacement arriv et évac	6	G
	I	Pose et raccordement évier	2	M
Mobilier	J	Dépose ancien éléments	4	G
	K	Assemb. caissons et tiroirs	8	Début
	L	Pose éléments bas	6	B,J,H,K
	M	Pose plan de travail	4	L
	N	Étanchéité plan de travail	1	E,M
	O	Pose éléments hauts	1	E,J
	Fin	Fin du projet	0	C,F,I,N,O

Les tâches sont classées par niveaux, transcrits dans l'ordre suivant :
N0= G,K ; N1= H,J ; N2= A,D ; N3= B,E ; N4= C,F,L,O ; N5= M ; N6= I,N .

Table des suivants

Tâches	Suivant
A	B
B	C,L
C	-
D	E
E	F,N,O
F	-
G	H,J
H	L
I	-
J	A,D,L,O
K	L
L	M
M	I,N
N	-
O	-

Le Réseau PERT est donné comme suit :



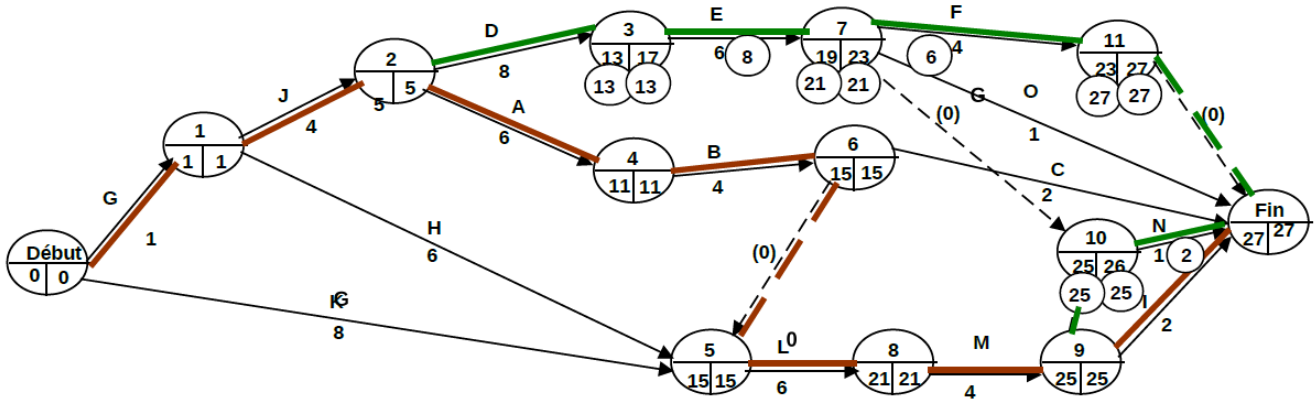
Tâches	minimale		normale		maximale	
	Durée min	Coût max	Durée normal	Coût normal	Durée max	Coût min
A	4	10	6	6	8	4
B	2	11	4	8	6	6
C	1	4	2	3	4	1
D	4	14	8	9	10	8
E	4	15	6	11	8	7
F	2	10	4	6	6	4
G	1/2	3	1	2	2	1
H	4	13	6	8	8	7
I	1	7	2	4	4	3
J	2	9	4	6	6	4
K	4	11	8	9	10	8
L	4	18	6	10	8	7
M	2	10	4	7	6	5
N	1/2	3	1	2	2	1
O	1/2	3	1	2	2	1

Amélioration sans modification de la longueur du chemin critique :

A partir du graphe d'ordonnancement à durées normales, et en admettant que l'oeuvre doit obligatoirement être exécutée en 27 jours, comment réduire le coût ?

Tâche	$d\Delta$	$c\Delta$	$Duc = \Delta c/\Delta d$
C	2=3-1	2=4-2	1=2/2
D	1=9-8	2=10-8	0.5=1/2
E	4=11-4	2=8-6	2=4/2
F	2=6-4	2=6-4	1=2/2
H	1=8-7	2=8-6	0.5=1/2
K	1=9-8	2=10-8	0.5=1/2
N	1=2-1	1=2-1	1=1/1
O	1=2-1	1=2-1	1=1/1

Les coûts sont tout à fait réductibles sur le laps de 27 heures, tel qu'on peut le remarquer sur la figure suivante rapportant les ordonnancements.



D'où l'apparition de Deux chemins critique GJDEF-GJABLMN

2 * 2+2 * 1+1 * 1 = 4+2+1 = 7 Le coût économisé est de l'ordre 7.

Pour la durée totale, le coût devient : 93-7=84 unités

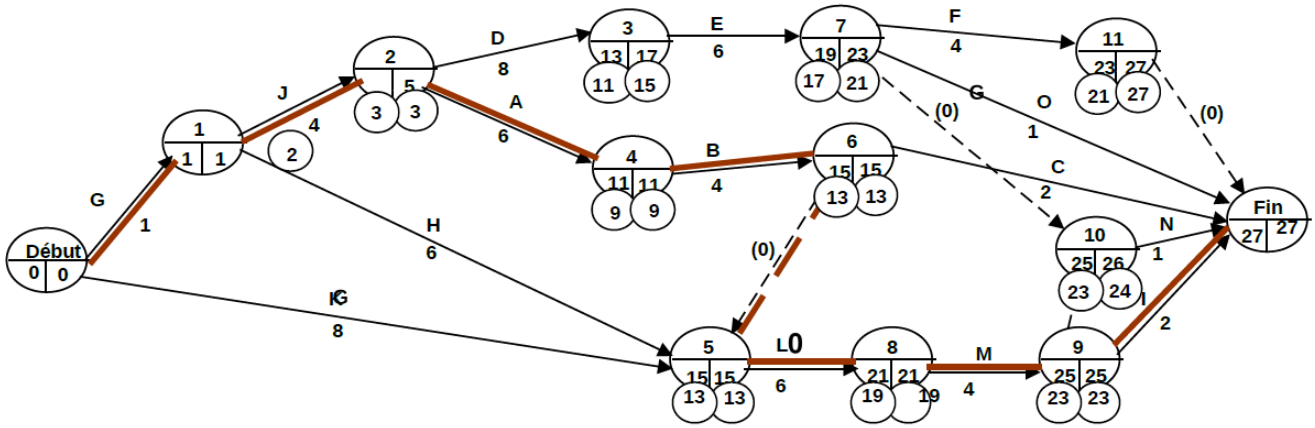
<u>NB</u> .Duc : différences unitaires de coûts

Amélioration avec réduction de la longueur du chemin critique :

A partir du graphe d'ordonnancement a durées normales, et si l'on suppose que la durée de réduction est déjà fixée, comment réduire ? le coût pour une durée bien déterminée. En supposant qu'une réduction de 2h est indispensable, tel qu ?on peut le lire sur le tableau infra, schématisé sur la figure suivante.

Tâche	$d\Delta$	$c\Delta$	$Duc = \Delta c/\Delta d$
G	1=3-2	0.5=1-1/2	1=1/0.5
J	3=9-6	2=4-2	1.5=3/2
A	4=10-6	2=6-4	2=4/2
B	3=11-8	2=4-2	1.5=3/2
L	8=18-10	2=6-4	4=8/2
M	3=10-7	2=4-2	1.5=3/2
I	3=7-4	1=2-1	1=3/1

On commence par réduire les durées des tâches critiques ayant plus faible duc. Les tâches J, B, M ont les mêmes différences unitaires de coûts, donc on choisit une parmi eux, par exemple : J. On réduit



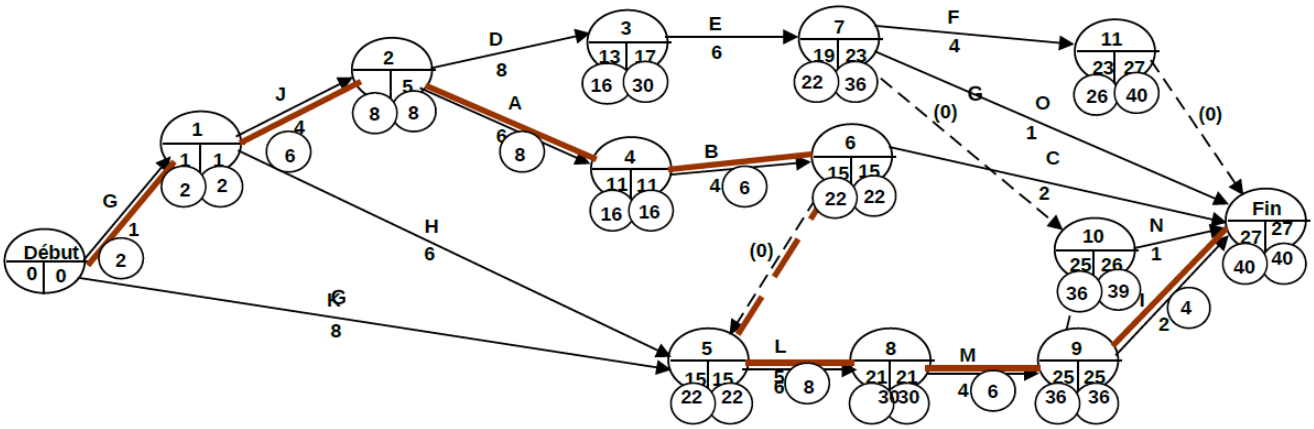
Pour une réduction d'une durée totale de 2h pour J le coût devient 93+2*1.5=96 unités

Amélioration avec augmentation du chemin critique :

A partir du graphe d'ordonnancement à durées normales, comment réduire le coût ?

Tâche	$d\Delta$	$c\Delta$	$Duc = \Delta c/\Delta d$
G	1=2-1	2=2-1	1=1/1
J	2=6-4	2=6-4	1=2/2
A	2=8-6	2=6-4	1=2/2
B	2=6-4	2=8-6	1=2/2
L	2=8-6	3=10-7	1.5=3/2
M	2=6-4	2=7-5	1=2/2
I	2=4-2	1=4-3	0.5=1/2

On commence par allonger les durées des tâches ayant la plus grande différence unitaire de coûts. On croit la durée de "L" en priorité de 2, pour obtenir un rabattement du coût comme la montre la figure plus bas.



Le coût économisé est de l'ordre de : 1.5*2+1*2+1*2+1*2+1*2+1*1+0.5*2=3+8+1+1=13. Pour une durée totale inchangée, le coût devient 93 - 13=80 unités.

3.3 Résolution d'un problème plus court chemin dans un réseau routier en Algérie

3.3.1 Plus court chemin d'une ville à une autre ville

Un promeneur décide de déplacer d'une ville à une autre en Algérie de demande quelle itinéraire doit-il suivre pour aller à une ville voisine au loitaine. Ce problème se modélise sous forme d'un graphe non orienté prenant la condition que la route est à deux sens telle que :

– Une ville représente un sommet .

– Deux ville ayant des frontières en commun, on relié les sommets représentant les deux villes par une arrête.

– la capacité de chaque arrête représente la distance en kilomètre entre les deux villes .

Pour simplifier et illustrer les différentes itérations de recherche de plus court chemin on utilisera le graphe de la figure 3.2 suivante.

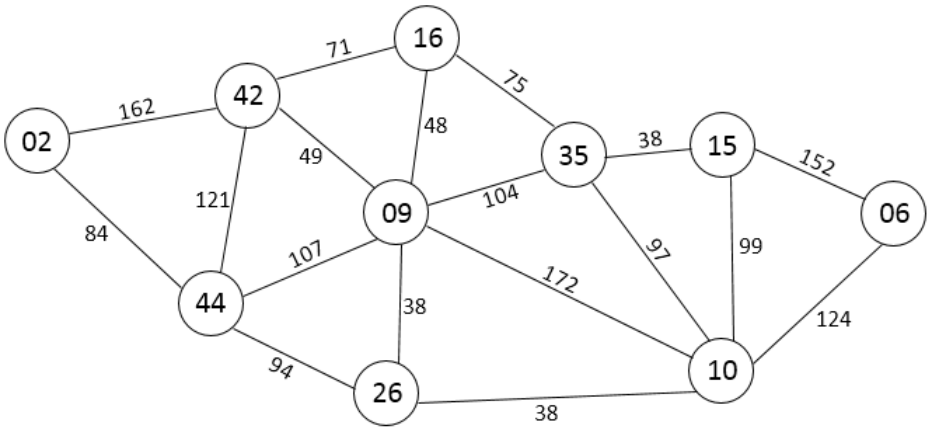


FIGURE 3.2 – Réseau routier simplifier

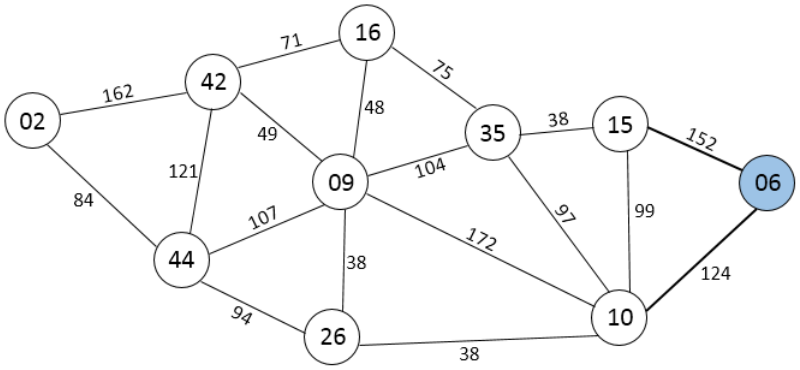
En remplaçant les noms des villes par leurs code administratif telle qu'ils sont indiquer au tableau ci-après.

Code	Wilaya	Code	Wilaya
06	Béjaia	10	Bouira
09	Blida	02	Chlef
15	Tizi Ouzou	44	Ain Defla
35	Boumerdes	26	Medea
16	Alger	42	Tipaza

3.3.2 Plus court chemin d'une ville à toutes les autres villes

La découverte de plus court chemin d'une ville à tout les autre villes, se fait à l'aide de l'algorithme de Dijkstra jusqu'à que ce tout les sommets soitent parcurus.

- itération 1

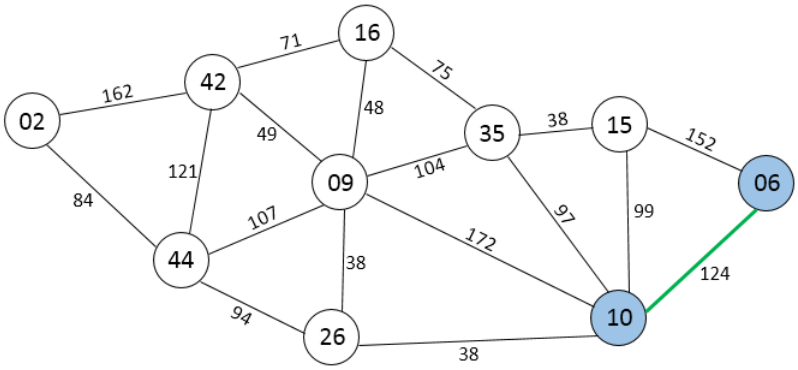


On examine les sommets voisins de sommet 06 ;
$\pi(15) = 152$, $\pi(10) = 124$.

$$\min(\pi(15), \pi(10)) = 124 = \pi(10)$$

.

Le sommet 10 va être marqué.



- itération 2

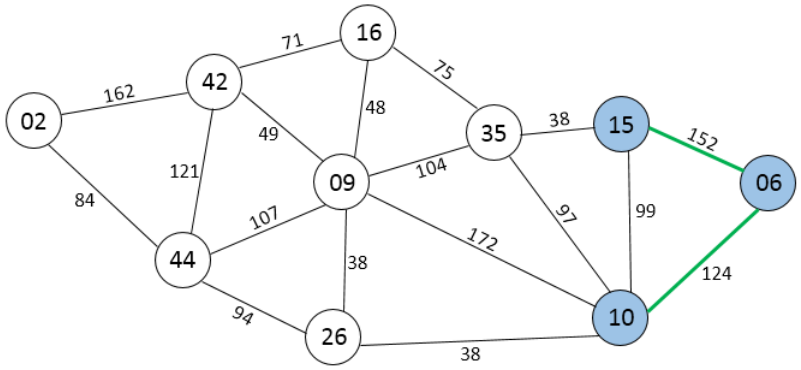
On examine les sommets voisins des sommets 06 et 10 ;

$\pi(3) = 124+97 = 221, \pi(26) = 124+38 = 162, \pi(09) = 172+124 = 296, \pi(15) = \min(152, 124+99) = 152.$

$$\min(\pi(15), \pi(35), \pi(26), \pi(09)) = 152 = \pi(15)$$

.

Le sommet 15 va être marqué.



- itération 3

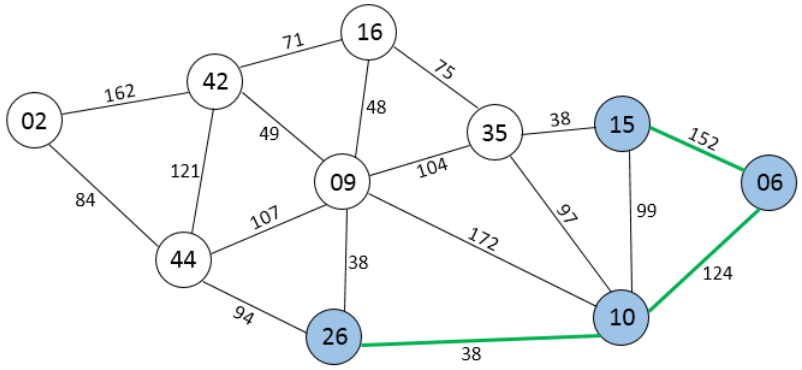
On examine les sommets voisins des sommets 06,10 et 15 ;

$\pi(35) = \min(152 + 38, 124 + 99 + 38, 124 + 97) = 190,\ \pi(26) = 124 + 38 = 162,\ \pi(09) = 124 + 172 = 296$

$$\min(\pi(35), \pi(26), \pi(09)) = 162 = \pi(26)$$

.

Le sommet 26 va être marqué.



- itération 4

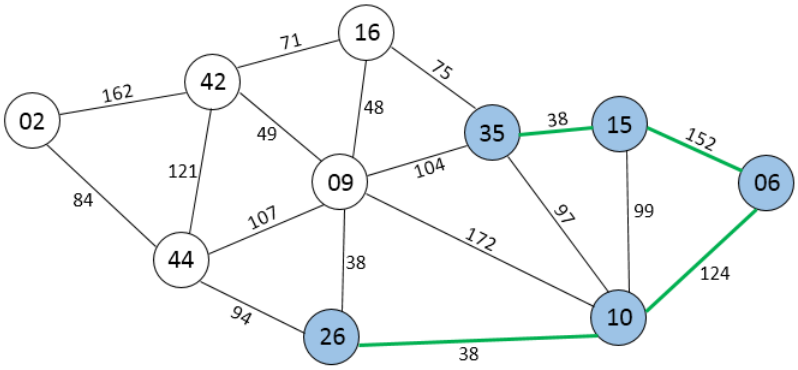
On examine les sommets voisins des sommets 06,10,15 et 26 ;

$\pi(35) = 190$, $\pi(09) = \min(124 + 172, 124 + 39 + 39) = 200$, $\pi(44) = \min(124 + 38 + 94, 152 + 99 + 38 + 94) = 256$

$$\min(\pi(35), \pi(09), \pi(44)) = 190 = \pi(35)$$

.

Le sommet 35 va être marqué.



- itération 5

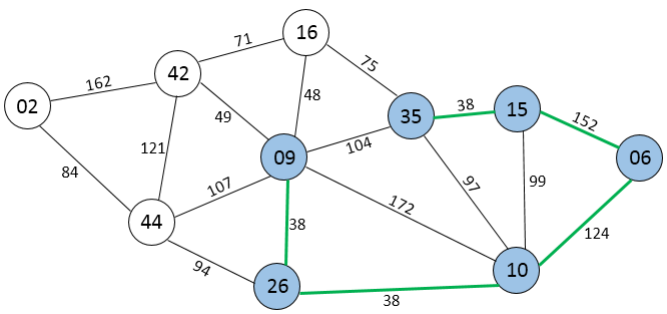
On examine les sommets voisins des sommets 06,10,26,15 et 35 ;

$\pi(09) = \min(152+38+104, 124+172, 124+38+38+200) = 200$, $\pi(16) = 152+38+75 = 265$, $\pi(44) = \min(124 + 38 + 94, 152 + 38 + 104 + 107) = 256$

$$\min(\pi(09), \pi(44)) = 200 = \pi(09)$$

.

Le sommet 09 va être marqué.



- itération 6

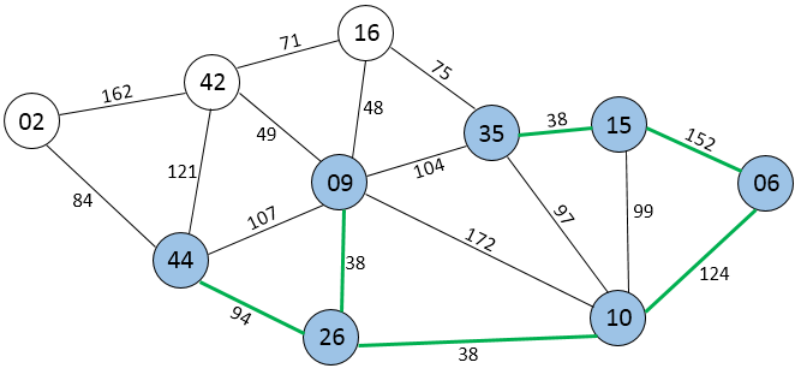
On examine les sommets voisins des sommets 06,10,26,15,35 et 09 ;

$\pi(16) = \min(152 + 38 + 75, 124 + 38 + 38 + 48) = 284$, $\pi(42) = 124 + 38 + 38 + 49 = 249$, $\pi(44) = \min(124 + 38 + 94, 124 + 38 + 38 + 107) = 241$

$$\min(\pi(16), \pi(42), \pi(44)) = 241 = \pi(44)$$

.

Le sommet 44 va être marqué.



- itération 7

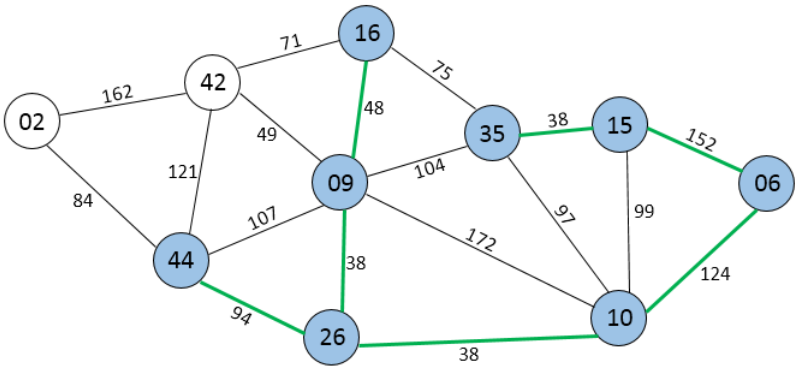
On examine les sommets voisins des sommets 06,10,15,26,35,09 et 44 ;

$\pi(16) = \min(152 + 38 + 75, 124 + 38 + 38 + 48) = 248V$,$\pi(42) = \min(142 + 38 + 94 + 121, 124 + 38 + 38 + 49) = 249$,$\pi(02) = 124 + 38 + 94 + 84 = 340$

$$\min(\pi(16), \pi(42), \pi(02)) = 2248 = \pi(16)$$

.

Le sommet 16 va être marqué.



- itération 8

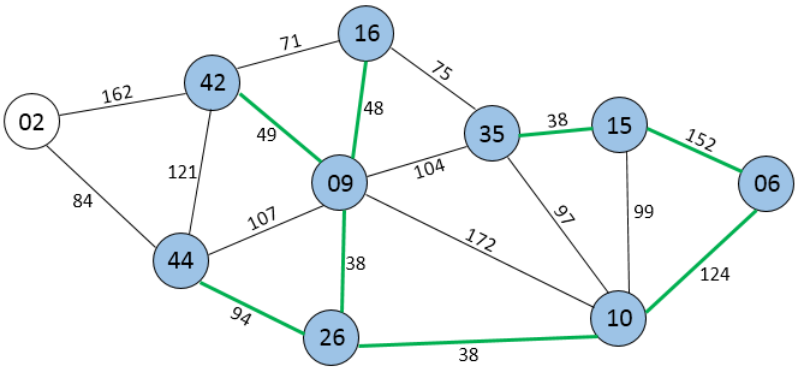
On examine les sommets voisins des sommets 06,10,15,26,35,09,44 et 16 ;

$\pi(42) = \min(124 + 38 + 38 + 48 + 71, 124 + 38 + 38 + 49, 124 + 38 + 94 + 121) = 249$,$\pi(02) = 124 + 38 + 94 + 84 = 340$

$$\min(\pi(42), \pi(02)) = 249 = \pi(42)$$

.

Le sommet 42 va être marqué.



- itération 9

On examine les sommets voisins des sommets 06,10,15,26,35,09,44, 16 et 42 ;

$\pi(16) = \min(124 + 38 + 38 + 49 + 162, 124 + 38 + 94 + 84) = 340$

Le sommet 02 va être marqué.

Tous les sommets sont marqués, alors on s'arréte Les plus courts chemins sont tels qu'ils sont illustré dans le graph de la dernier itération (itération 9).

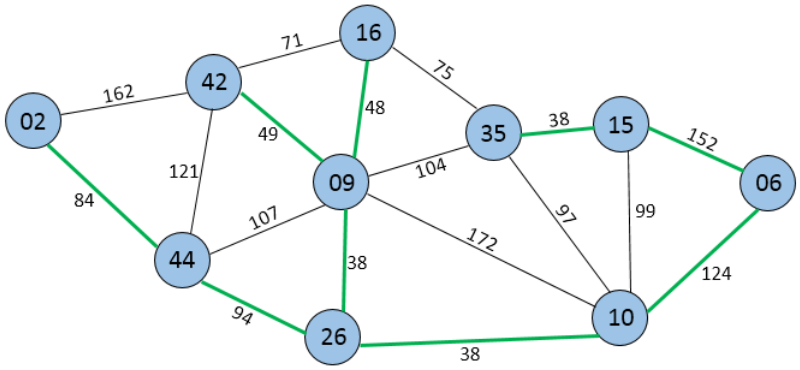


FIGURE 3.3 – Plus court chemin de la ville de "Béjaia" à toutes autre ville

Arborescence des plus couts chemin entre la wilaya de "Béjaia" et les autres wilaya :

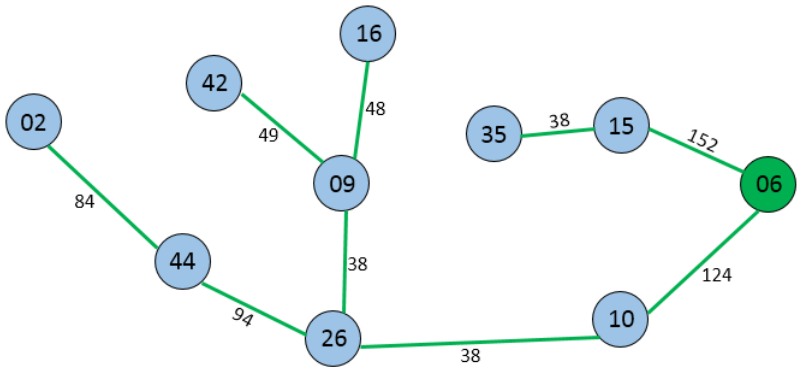


FIGURE 3.4 – Arborescence des plus couts chemin

3.4 Implémentation sur logiciel JAVA l'algorithme de Dijkstra

Pour simplifier la recherche de plus court chemin entres les différentes wilaya d'Algérie nous tenons à mettre en œuvre cette application.

- Interface JAVA

3.5 Recherche des plus courts chemin entre la wilaya de béjaia et les autres wilayas

A présent nous allons élaborer un exemple illustratif où on utilise cette application, Pour déterminer les plus courts chemins menant de la ville de Béjaia vers toutes les autres villes d'Algérie.

- Le tableau suivant représente les différents codes de chaque wilaya :

(01) Adrar	(13) Tlemcen	(25) Constantine	(37) Tindouf
(02) Chlef	(14) Tiaret	(26) Medea	(38) Tissemsilt
(03) Laghouat	(15) Tizi ouzou	(27) Mostaganem	(39) El-Oued
(04) Oum El Bouaghi	(16) Alger	(28) M'Sila	(40) Khenchela
(05) Batna	(17) Djelfa	(29) Mascara	(41) Souk Ahras
(06) Béjaia	(18) Jijel	(30) Ouargla	(42) Tipaza
(07) Biskra	(19) Setif	(31) Oran	(43) Mila
(08) Béchar	(20) Saïda	(32) El Bayadh	(44) Aïn Defla
(09) Blida	(21) Skikda	(33) Illizi	(45) Naâma
(10) Bouira	(22) Sidi Bel Abbès	(34) Bordj Bou Arreridj	(46) Aïn Témouchent
(11) Tamanrasset	(23) Annaba	(35) Boumerdès	(47) Ghardaïa
(12) Tébessa	(24) Guelma	(36) El Tarf	(48) Relizane

- La carte géographique de l'Algérie indiquant les frontières entre les wilayas :

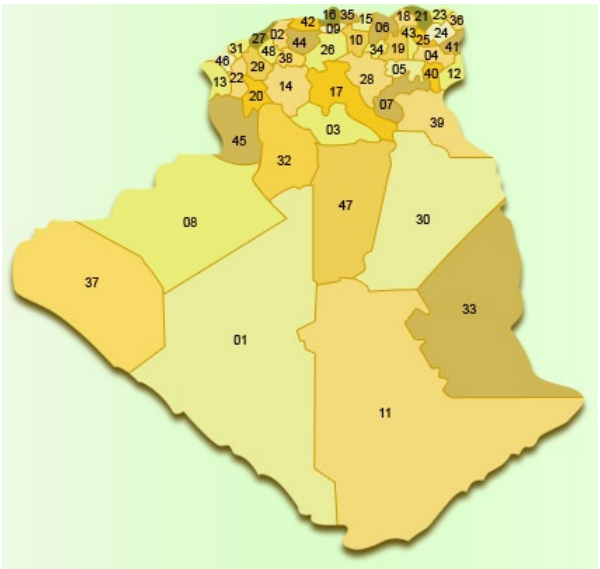
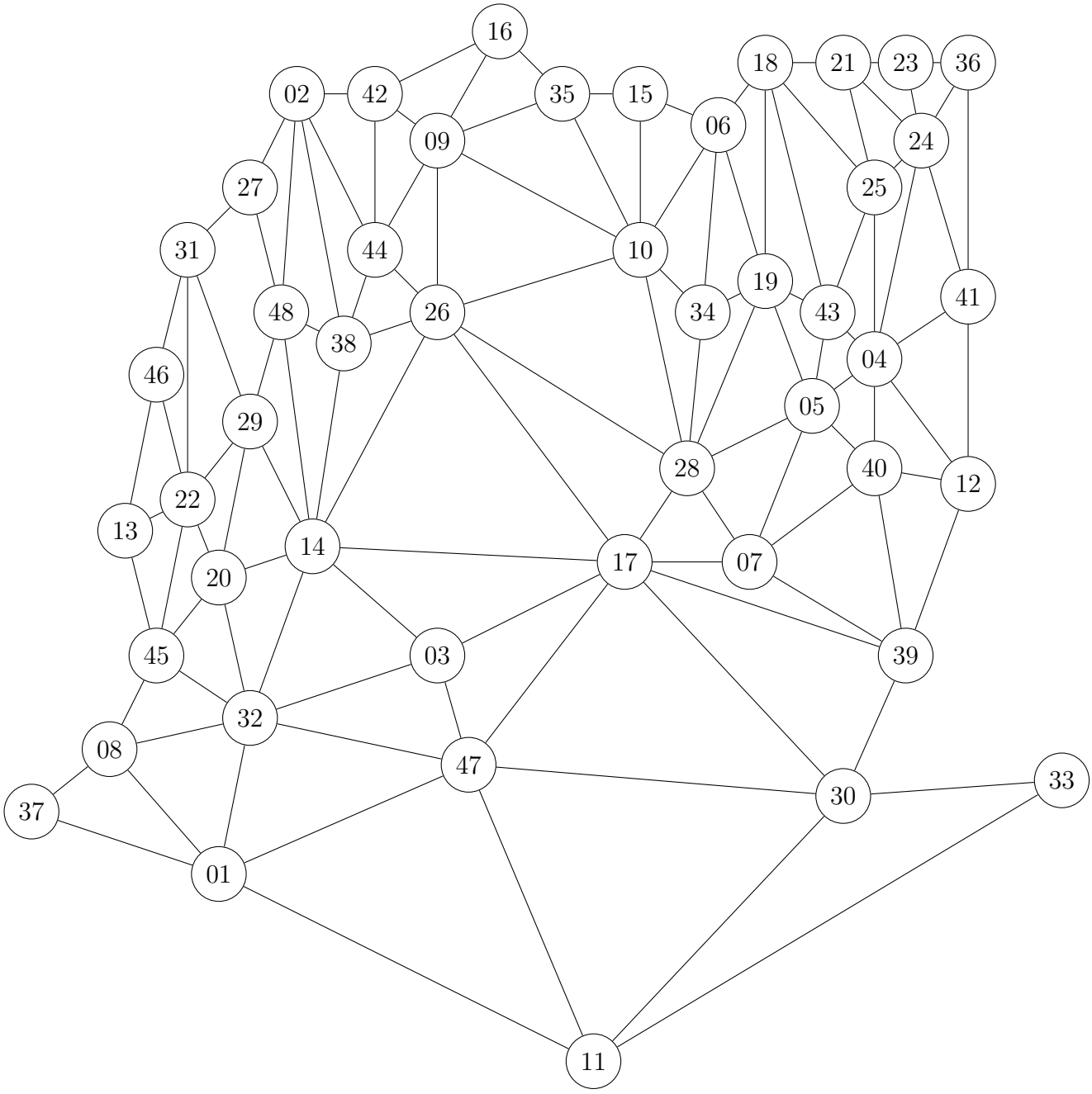


FIGURE 3.5 – Carte indiquant les frontières entre les wilayas

- Le graphe induit de la carte routière :

Le graphe induit de la carte routière en représentant les wilayas comme étant des sommets, et les routes comme étant des arrêtes, est le suivant :



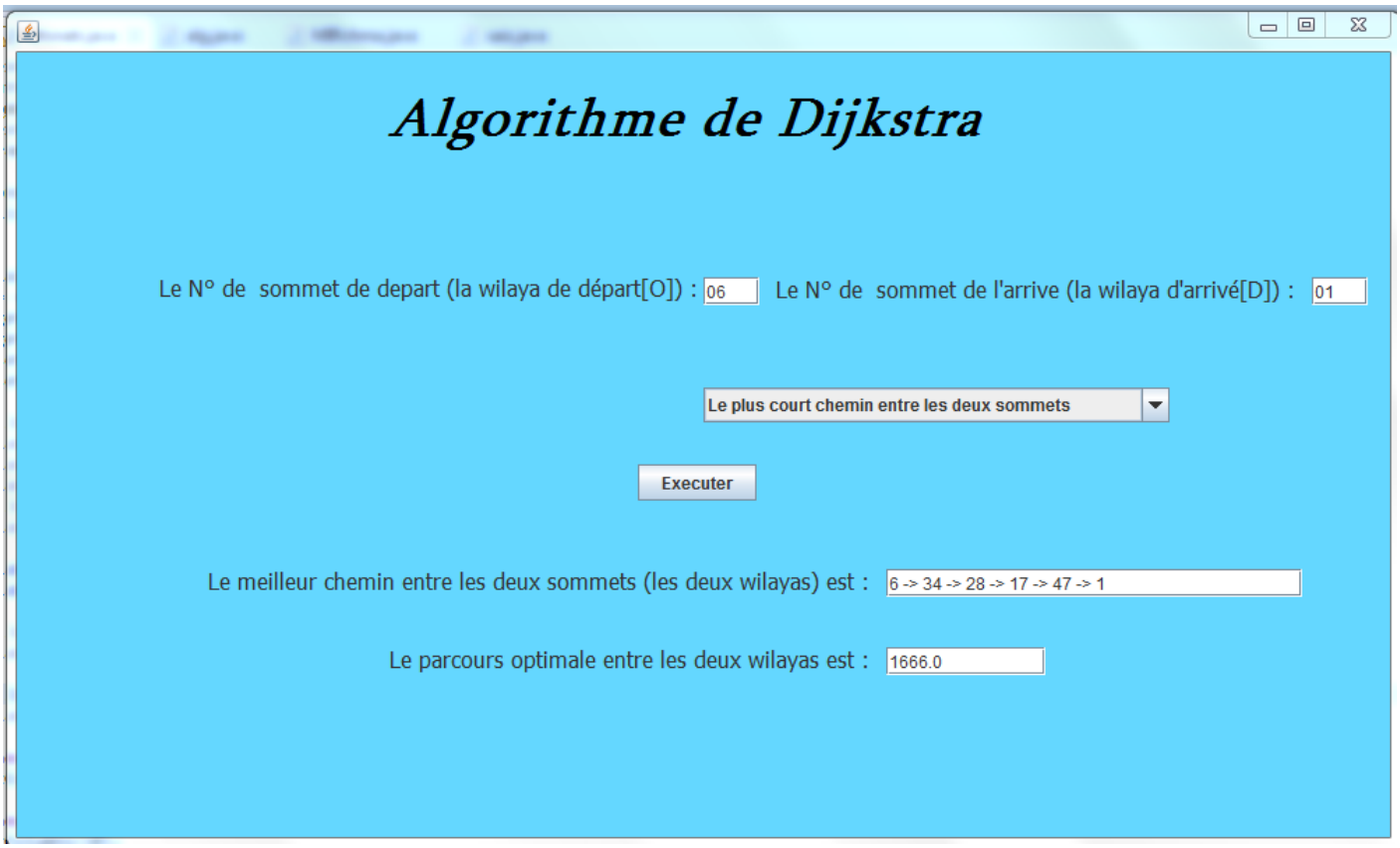
- Le tableau suivant représente la distance entre les wilaya qui ont des frontières en commun (en *kilomètre*) :

Wilaya	**La distance entre wilaya (voisine)**						
(01)	(11) 1042	(47) 944	(37) 1194	(08) 574	(32) 1014	\	\
(02)	(38) 111	(42) 164	(27) 134	(44) 84	(48) 98	\	\
(03)	(32) 277	(17) 217	(14) 321	(47) 191	\	\	\
(04)	(43) 172	(40) 61	(25) 122	(12) 151	(24) 140	(41) 153	(05) 114
(05)	(40) 103	(07) 116	(28) 178	(19) 131	(43) 152	\	\
(06)	(34) 144	(18) 92	(19) 110	(10) 124	(15) 152	\	\
(07)	(40) 190	(39) 225	(28) 180	(17) 273	\	\	\
(08)	(32) 441	(37) 804	(45) 358	\	\	\	\
(09)	(35) 104	(16) 48	(42) 49	(44) 107	(10) 172	(26) 38	\
(10)	(34) 96	(15) 99	(35) 97	(28) 132	(26) 146	\	\
(11)	(47) 1454	(33) 838	(30) 1279	\	\	\	\
(12)	(40) 113	(41) 129	(39) 316	\	\	\	\
(13)	(46) 68	(45) 214	(22) 91	\	\	\	\
(14)	(32) 215	(29) 129	(48) 104	(38) 57	(26) 204	(20) 156	(17) 253
(15)	(35) 38	\	\	\	\	\	\

(16)	(42) 71	(35) 75	\	\	\	\	\
(17)	(30) 545	(47) 341	(39) 469	(28) 179	(26) 220	\	\
(18)	(43) 98	(21) 147	(19) 135	(25) 131	\	\	\
(19)	(43) 113	(28) 126	(34) 71	\	\	\	\
(20)	(32) 199	(45) 246	(22) 97	(29) 76	\	\	\
(21)	(23) 99	(24) 99	(25) 82	(43) 111	\	\	\
(22)	(45) 273	(29) 90	(31) 81	(46) 61	\	\	\
(23)	(24) 68	(36) 70	\	\	\	\	\
(24)	(36) 99	(25) 110	(41) 75	\	\	\	\
(25)	(43) 50	\	\	\	\	\	\
(26)	(38) 149	(44) 94	(28) 231	\	\	\	\
(27)	(31) 83	(48) 60	(29) 79	\	\	\	\
(28)	(34) 58	\	\	\	\	\	\
(29)	(31) 104	(48) 67	\	\	\	\	\
(30)	(47) 288	(39) 260	(33) 1057	\	\	\	\
(31)	(46) 79	\	\	\	\	\	\
(32)	(45) 249	(47) 430	\	\	\	\	\

(36)	(41) 91	\	\	\	\	\	\
(38)	(48) 154	(44) 96	\	\	\	\	\
(39)	(40) 312	\	\	\	\	\	\
(42)	(44) 121	\	\	\	\	\	\

Exécution sur le programme

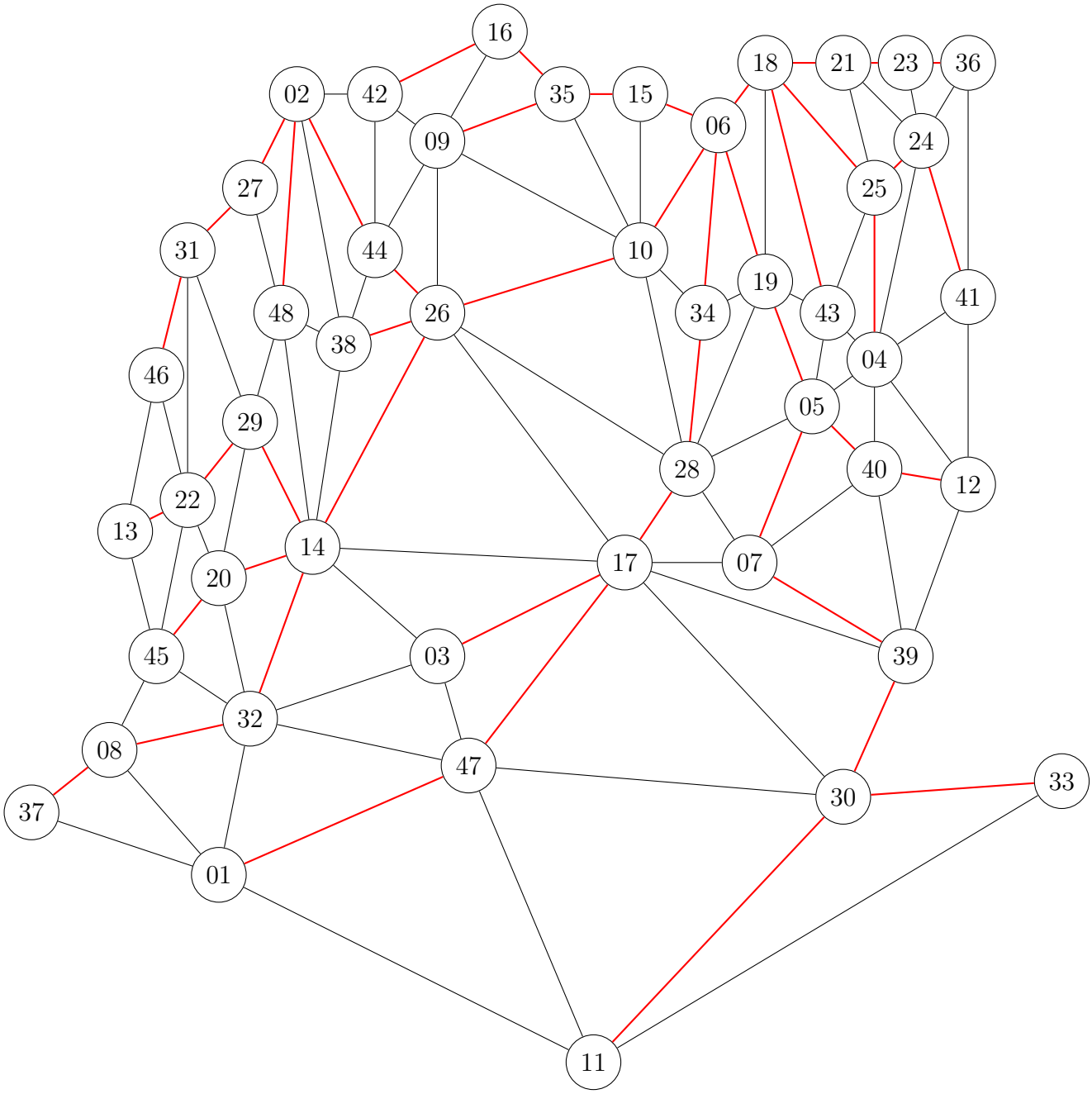


Après avoir exécuter le programme Les résultats obtenus sont illustrés sur le tableau suivant :

la destination	Chemin le plus court	la distance (Km)
Bejaia ⇒ Adrar	06 → 34 → 28 → 17 →47 → 01	1666
Bejaia ⇒ Chlef	06 → 10 → 26 → 44 → 02	448
Bejaia ⇒ Laghouat	06 → 34 → 28 → 17 → 03	598
Bejaia ⇒ Oum el-bouaghi	06 → 18 → 25 → 04	345
Bejaia ⇒ Batna	06 → 19 → 05	241
Bejaia ⇒ Biskra	06 → 19 → 05 → 07	357
Bejaia ⇒ Bechar	06 → 10 → 26 →14 → 32 → 08	1130
Bejaia ⇒ Blida	06 → 15 → 35 → 09	294
Bejaia ⇒ Bouira	06 → 10	124
Bejaia ⇒ Tamanrasset	06 → 19 → 05 → 07 → 39 → 30 → 11	2121
Bejaia ⇒ Tebessa	06 → 19 → 05 → 40 → 12	457
Bejaia ⇒ Tlemcen	06 → 10 → 26 → 14 → 29 → 22 → 13	784
Bejaia ⇒ Tiaret	06 → 10 → 26 → 14	474
Bejaia ⇒ Tizi-ouzou	06 → 15	152
Bejaia ⇒ Alger	06 → 15 → 35 → 16	265
Bejaia ⇒ Djelfa	06 → 34 → 28 → 17	381
Bejaia ⇒ Jijel	06 → 18	92
Bejaia ⇒ Setif	06 → 19	110
Bejaia ⇒ Saida	06 → 10 → 26 →14 → 20	630
Bejaia ⇒ Skikda	06 → 18 → 21	239
Bejaia ⇒ Sidi bel-abbes	06 → 10 → 26 → 14 → 29 → 22	693
Bejaia ⇒ Annaba	06 → 18 → 21 → 23	338
Bejaia ⇒ Guelma	06 → 18 → 25 → 24	333
Bejaia ⇒ Constantine	06 → 18 → 25	223
Bejaia ⇒ Medea	06 → 10 → 26	270
Bejaia ⇒ Mostaganem	06 → 10 → 26 → 44 → 2 → 27	582
Bejaia ⇒ M'sila	06 → 34 → 28	202
Bejaia ⇒ Mascara	06 → 10 → 26 → 14 → 29	603
Bejaia ⇒ Ouargla	06 → 19 → 05 → 07 → 39 → 30	842
Bejaia ⇒ Oran	06 → 10 → 26 → 44 → 02 → 27 → 31	665
Bejaia ⇒ El-bayadh	06 → 10 → 26 →14 → 32	689

Bejaia ⇒ Illizi	06 → 19 → 05 → 07 → 39 → 30 → 33	1899
Bejaia ⇒ B.B.arreridj	06 → 34	144
Bejaia ⇒ Boumerdes	06 → 15 → 35	190
Bejaia ⇒ El-tarf	06 → 18 → 21 → 23 → 36	408
Bejaia ⇒ Tindouf	06 → 10 → 26 → 14 → 32 → 8 → 37	1934
Bejaia ⇒ Tissemsilt	06 → 10 → 26 → 38	419
Bejaia ⇒ El-oued	06 → 19 → 05 → 07 → 39	582
Bejaia ⇒ Khenchela	06 → 19 → 05 → 40	344
Bejaia ⇒ Souk ahras	06 → 18 → 25 → 24 → 41	408
Bejaia ⇒ Tipaza	06 → 15 → 35 → 16 → 42	336
Bejaia ⇒ Mila	06 → 18 → 43	190
Bejaia ⇒ Ain defla	06 → 10 → 26 → 44	364
Bejaia ⇒ Naâma	06 → 10 → 26 → 14 → 20 → 45	876
Bejaia ⇒ Ain temouchent	06 → 10 → 26 → 44 → 2 → 27 → 31 → 46	744
Bejaia ⇒ Ghardaia	06 → 34 → 28 → 17 → 47	722
Bejaia ⇒ Relizane	06 → 10 → 26 → 44 → 2 → 48	546

- Le graphe qui montre les plus courts chemin entre la wilaya Béjaia et les autres wilaya :

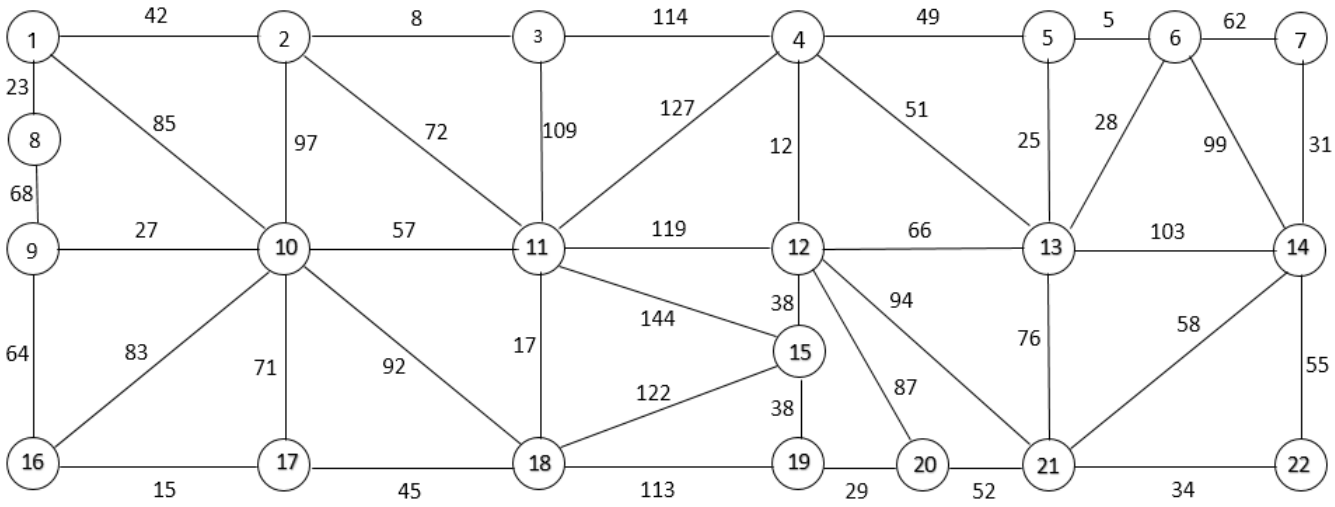


3.6 Résolution d'un problème de Télécommunication

- Problème :

Le but est de construire un réseau (G = (X ;E)) électrique qui relie toutes les villes et dont le coût de construction soit le plus petit possible. Un graphe est utilisé pour représenter le réseau. Chaque ville est représentée par un sommet et les arêtes sont les lignes électriques qu'il est possible de construire. Chacune de ces lignes a un coût : par exemple en millions dinars algeriens .

En termes de graphes, construire un réseau électrique optimal revient à calculer un arbre couvrant de poids minimal. Pour résoudre ce problème, En utilisent l'algorithme de Kruskal .
La situation est représente par le graphe suivant :



- L'implimentation de l'algorithme de kruskal en java

On simplifie la recherche de l'arbre couvrant minimun, ensuite on implimente l'algorithme de Kruskal pour le ghraphe ci-dessus en JAVA.

Source History

```
1  ckage kruskal;
2  port java.util.*;
3  port java.io.*;
4  blic class Kruskal {
5          public static void main(String[] args) {
6                  ArrayList<Edge> graphEdges = new ArrayList<>();
7                  graphEdges.add(new Edge(5, 6, 5)); graphEdges.add(new Edge(2, 3, 8));
8                  graphEdges.add(new Edge(4, 12, 12));graphEdges.add(new Edge(16, 17, 15));
9                  graphEdges.add(new Edge(11, 18, 17));graphEdges.add(new Edge(1, 8, 23));
10                 graphEdges.add(new Edge(5, 13, 25));graphEdges.add(new Edge(9, 10, 27));
11                 graphEdges.add(new Edge(6, 13, 28));graphEdges.add(new Edge(19, 20, 29));
12                 graphEdges.add(new Edge(7, 14, 31));graphEdges.add(new Edge(21, 22, 34));
13                 graphEdges.add(new Edge(12, 15, 38));graphEdges.add(new Edge(1, 2, 42));
14                 graphEdges.add(new Edge(17, 18, 45));graphEdges.add(new Edge(4, 5, 49));
15                 graphEdges.add(new Edge(4, 13, 51));graphEdges.add(new Edge(20, 21, 52));
16                 graphEdges.add(new Edge(14, 22, 55)); graphEdges.add(new Edge(10, 11, 57));
17                 graphEdges.add(new Edge(14, 21, 58)); graphEdges.add(new Edge(6, 7, 62));
18                 graphEdges.add(new Edge(9, 16, 64)); graphEdges.add(new Edge(12, 13, 66));
19                 graphEdges.add(new Edge(19, 15, 66));graphEdges.add(new Edge(8, 9, 68));
20                 graphEdges.add(new Edge(10, 17, 71));graphEdges.add(new Edge(2, 11, 72));
21                 graphEdges.add(new Edge(21, 13, 76));graphEdges.add(new Edge(10, 16, 83));
22                 graphEdges.add(new Edge(1, 10, 85));graphEdges.add(new Edge(12, 20, 87));
23                 graphEdges.add(new Edge(10, 18, 92));graphEdges.add(new Edge(12, 21, 94));
24                 graphEdges.add(new Edge(2, 10, 97));graphEdges.add(new Edge(13, 14, 103));
25                 graphEdges.add(new Edge(3, 11, 109));graphEdges.add(new Edge(19, 18, 113));
26                 graphEdges.add(new Edge(3, 4, 114));graphEdges.add(new Edge(11, 12, 119));
27                 graphEdges.add(new Edge(18, 15, 122));graphEdges.add(new Edge(4, 11, 127));
28                 graphEdges.add(new Edge(11, 15, 144));
```

On obtient le resultat ci-dessous :

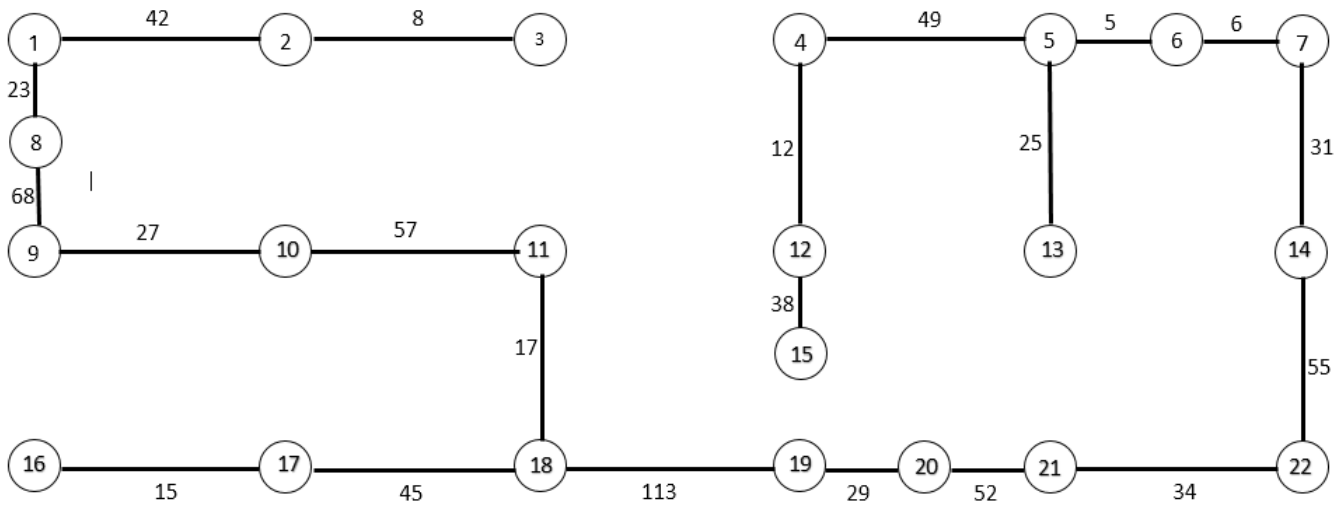
```
Final Minimum Spanning Tree (21 edges)
Edge (5, 6) weight=5
Edge (2, 3) weight=8
Edge (4, 12) weight=12
Edge (16, 17) weight=15
Edge (11, 18) weight=17
Edge (1, 8) weight=23
Edge (5, 13) weight=25
Edge (9, 10) weight=27
Edge (19, 20) weight=29
Edge (7, 14) weight=31
Edge (21, 22) weight=34
Edge (12, 15) weight=38
Edge (1, 2) weight=42
Edge (17, 18) weight=45
Edge (4, 5) weight=49
Edge (20, 21) weight=52
Edge (14, 22) weight=55
Edge (10, 11) weight=57
Edge (6, 7) weight=62
Edge (8, 9) weight=68
Edge (19, 18) weight=113

Total weight of all edges in MST = 807

Open "06outputMST.txt" for backup copy of answers
BUILD SUCCESSFUL (total time: 0 seconds)
```

$|W| = 21$

On a : $|W| = n - 1 = 22 - 1 = 21$. terminé

L'arbre de poids minimum est représenté par le graphe ci-dessous A = (X ;W).

et le poids minimale totale est = 807.



Conclusion

Il a été question dans ce chapitre de certains problèmes concrets que nous avons modélisés ainsi que de leur résolution à l'aide des méthodes vues dans le chapitre précédent. Afin de faciliter la recherche du chemin le plus court, on a donné un programme sous le **logiciel JAVA** qui nous permette de donner les solu-tions les plus rapides.

CONCLUSION GÉNÉRALE

En feuilletant cette étude, le lecteur se donnera déjà une vision d'ensemble sur la Théorie des graphes. Elle est d'un certain certain, ne serait-ce que par ce qu'elle permet d'optimisation des réseaux de bien des sortes. Le graphe met à portée des précisions sur des situations que l'on a long-temps crues hors de portée des calculs humains, bravant les risques ou se fiant aux bienveillances des oracles et autres superstitions pourtant muettes. Le graphe aura bel et bien « apprivoisé » bien des problèmes à travers la possibilité con-crète de modéliser une grande variété d ?entre eux, en procédant par l'étude des sommets et des arcs. Avec le temps, son efficacité a bien établi son avantage et sa réputation dans le domaine de la recherche Opéra-tionnelle (R.O) est emblématique. L'importance de la Théorie des graphes tient également au fait qu'elle est en mesure de fournir un cadre conceptuel d'analyse appropriée de tant de situations. Elle s'était, en effet, développée au sein de disci-plines diverses : application industrielle, problème du convoyage et de voyage de commerce, problèmes de répartition et d'affectation, etc. Actuellement, le gros du travail qu'elle est attendue à livrer se trouve dans la modélisation et la résolution des situations sécuritaires au sens large.

Nous nous sommes, par cette étude, affairés à donner un certain nombre d'outils (algorithmes) de la Théorie des graphes directement employables dans la résolution de problèmes dont l'éventualité frôle la certitude. Notre objectif, l'exactitude, consiste à déverrouiller des problèmes en utilisant, ici, l'optimisation mise à portée par la Théorie des graphes, plus particulièrement les réseaux. Différents cas ont été traités, dont les télécommunications, en saisissant l'opportunité que nous fournit l'algorithme de Kruskal.

L'affectation de travailleurs à des machines est moyen reconnu de maximiser la production, mais dépend de la personne maniant la ma-chine. Afin d'obtenir une affectation qui soit la plus satis-faisante. Le troisième problème, la réhabilitation d'une habitation, avec moyens et délais propres à réduire temps et coûts des travaux de remise à neuf, a pu être aisément modélisé sous forme d'un réseau PERT. Le quatrième cas, celui du remplacement d'un véhicule où l'on cherchait une

meilleure politique de rem-placement (plus court chemin) qui, par le biais de l'usage de l'algorithme de Bellman, fera un appréciable amortissement de coût. Le dernier problème, l'élaboration d'une application pour la résolution d'un pro-blème de recherche du plus court chemin entre wilayas d'Algérie, de telle sorte que le trajet soit aussi agréable que possible, sans perdre de vue son aspect pratique et sa performance. Dans ce cas, muni d'une interface claire et accessible, d'usage aussi facile que fréquent, l'algorithme de **Dijkstra** implémenté sous **JAVA** s'est révélé pertinent.

BIBLIOGRAPHIE

[1] Létocart.L."Algorithmique de graphes". LIPN-UMR CNRS 7030.

[2] Maquin.D. Éléments de théorie des graphes. Version provisoire du 3 mai 2003.

[3] Mesmay.A, Monerau.M. plongements de graphes et distorsion. Sous la direction de P.Pansu et E.Colin, juin 2008.

[4] Michel. Sakarovitch, "Graphes et programmation linéaire", Editions Hermann, France, 08/09/1988.

[5] Muller.D. Introduction à la théorie des graphes. Cahiers de la CRM, Cahier n°6-Commission Romande de mathématique.

[6] Mohamed Ali. Aloulou, « Introduction aux problèmes d'ordonnancement», Dunod Editions, Paris Dauphine,28 novembre 2005.

[7] Philippe. Compoint, "Les Graphes en recherche opérationnelle", Dunod Editions, Paris 1972.

[8] Philippe. Mahey, "La programmation linéaire ", Editions Technip, France 1967.

[9] Roy.B. Algèbre moderne et théorie des graphes. Tome II, Dunod, Paris, 1984.

[10] Scheid.F.J. Graphes et Recherche Opérationnelle. ESIAL 2A.

[11] Fontan.G. Combinatoire et Ordonnancement. Notes de Cours, DEA Automatique et Informatique Industrielle, Université Paul, Sabatier, Toulouse, 1987.

[12] Gandron.Michel. et Minoux.Michel, "Graphes et algorithmes", Editions Eyrolles, Paris VI 1986.

[13] Jacques. Labelle, "Théorie des graphe", Modulo Editeur, Québec 1981.

[14] Lopez.P. "Cours de graphes". LAAS-CNRS, 30 novembre 2005.

[15] Labelle.J. "Théorie des Graphes". Modulo, 1981.

[16] Coilland.H. Polycopié d'exercices de recherche opérationnelle. IUT Informatique Nancy 2 et Ecole des Mines de Nancy,juillet 1999.

[17] Christophe. Rossignol, « Graphes pondéré, étiquetés, probabilistes », Grenoble 1989.

[18] Desbazeille.G. Exercices et problèmes de recherche opérationnelle. Dunod, Paris, 1976.

[19] Didier. Muller, « Introduction à la théorie des graphes », City Editions, France 2007.

[20] Everard.F. Théorie des graphes.2010

[21] Bondy.A, et U.S.R. Murty , «Graph theory». 2008.

[22] Bretto.A, A.Faisant, F.Hennecart. Éléments de théorie des graphes. 15/05/2012

[23] Berge.C. Graphes and hypergraphs. dunond.paris, 1973.

[24] Berge.Claude., « Graphes et hypergraphes », Dunod Editions, Paris 1970.

[25] Berge.C. Graphes. ISBN 2-04-15555-4, Gauthiers-Villars, Bordas, Paris, 1983.

[26] Benjamin.K. Modélisation,Optimisation,Complexité et algorithme. 2005-2006.

[27] BOUDJEMA.W,BOURAS.S ," Optimisation dans les réseaux" ,Mémoire Master 2 en Recherche Opérationnelle université de bejaia.

[28] Carré.B. Graphs and Networks. Clarendon Press, Oxford, 1979.

[29] Christine Costa-Marie. Optimisation dans les graphes. ENSTA-UMA-CEDRIC-PARIS, (2011-2012).

[30] Cabane.R. Théorie des graphes, Techniques de l'ingénieur. Traitè Sciences fondamentales, document AF205.

[31] West.D. Introduction to graph theory, (2nd edition),prentice hall. 2000.

[32] Tekaya.Wajdi, "Problème d'arbre couvrant de poinds minimum ", these de doctorat, Université Paris dauphine.

[33] TAOUINET.Smail "Théorie des Graphes avancés", note de cours Université de Béjaia 2016.

[34] Brahmi.D "optimisation des réseaux ", note de cours Université de Béjaia 2017-2018.

[35] Chapitre1,Cractéristique du problème d'ordonnancement PDFhttps ://d1n7iqsz6ob2ad.cloudfront.net>

[36] Théorie des graphes et ses applications/Claude Berge,1958. monde,2002-08-12

Resumé

Notre travail a porté sur la « Théorie des graphes », branche fameuse de la Recherche opérationnelle qui s'implique dans l'écrasante majorité des domaines. Son mérite dans l'optimisation des réseaux est tout indiqué. Le champ d'application où nous l'avions éprouvée est à la fois le transport routier sur les réseaux du pays ainsi que les télécommunications. Le travail brassa également la question du calcul des chemins les plus courts. Une application réalisée sous le programme **JAVA** a été appliquée avec succès au cas du réseau routier algérien.

Mots-clés : réseau, optimisation, algorithme de Djikstra...

Abstract

Our work focused on the "Graph Theory", the well-known branch of Operational Research that is involved in the overwhelming majority of fields. His merit in optimizing networks is all right. The field of application where we had tested it is both the road transport on the net-works of the country as well as telecommunications. The work also brewed the question of evaluating the shortest paths. A **JAVA** application program has been successfully applied to the case of the Algerian road network.

Keywords : network, optimization, algorithm of Dijkstra....