الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
Ecole Supérieure en Génie
Electrique et Energétique d'Oran
Département du cycle préparatoire
المدرسة العليا في الهندسة الكهربائية و الطاقوية
بوهران
قسم التكوين التحضيري

POLYCOPIÉ
COURS ET EXERCICES

Informatique 3

INTRODUCTION À LA THÉORIE DES GRAPHES ET À LA PROGRAMMATION LINÉAIRE

Selon le programme pédagogique des classes préparatoires en sciences et technologies (juillet 2015)

Rédigé par
Dr. TANDJAOUI Amel Faiza

février 2024

TABLE DES MATIÈRES

INTRODUCTION GÉNÉRALE

La recherche opérationnelle est ensemble d'outils d'aide à la décision. Étant donné un problème, elle consiste à identifier la ou les meilleures solutions possibles. Elle lie plusieurs domaines, notamment l'informatique et les mathématiques.

Pour ce cours, nous étudierons deux de ces outils : les graphes et la programmation linéaire.

Les graphes servent à modéliser des problèmes sous forme graphique et à exploiter leurs propriétés connues et à les algorithmes développés pour des classes de problèmes similaires.

La programmation linéaire permet de modéliser des problèmes d'optimisation sous forme mathématique lorsque leur fonction objectif et leurs contraintes peuvent être formulées sous forme linéaire.

À l'issue de ce cours, l'étudiant se sera familiarisé avec :

— Le concept de modélisation de problèmes ;

— Des méthodes de résolution de problèmes classiques de la recherche opérationnelle.

Ce support de cours est structuré en quatre chapitres avec trois chapitres portant sur la théorie des graphes et un chapitre portant sur la programmation linéaire, comme suit :

Chapitre 1 - concept de graphes : Ce chapitre introduit principalement la terminologie propre à la théorie des graphes

Chapitre 2 - algorithmes de base en théorie des graphes : Ce chapitre traite deux problèmes classiques de la théorie des graphes. À savoir, le problème des plus courts chemins et le problème du flot maximal. Un algorithme y est étudié pour chacun des problèmes : algorithme de Dijkstra pour le premier et algorithme de Ford-Fulkerson pour le second.

Chapitre 3 - notions de base des arbres : Ce chapitre aborde un cas particulier de graphes qui sont les arbres. Y figure notamment le concept de l'arbre couvrant minimal et l'algorithme de Kruskal.

Chapitre 4 - introduction à la programmation linéaire : Ce chapitre traite le concept de programmation linéaire. Y est traitée la méthode de la résolution graphique ainsi que la méthode du simplexe. Une brève introduction au principe de la dualité y figure également.

On retrouvera une série d'exercices applicatifs résolus à la fin de chaque chapitre.

CHAPITRE 1

LE CONCEPT DE GRAPHES

1.1 Introduction

Comme pour les probabilités, beaucoup de travaux de la théorie des graphes ont été motivés par le jeu.

Les graphes sont constitués de points et de liens entre ces points. Les points peuvent représenter différentes notions selon le problème étudié, et les liens représentent des relations entre ces notions.

Un graphe sera dit **non orienté** si la relation représentée est réciproque. Dans le cas contraire, il sera dit **orienté**.

Les graphes sont pratiques pour représenter des problèmes de manière visualisable (graphique) : coloration de carte, représentation de molécules, problème des sept ponts de Königsberg (problème original), etc.

Ils permettent de construire un modèle théorique d'un problème et de le résoudre en exploitant des propriétés et des algorithmes déjà disponibles.

La plupart des problèmes traités par la théorie des graphes peuvent être décrits comme suit :

- Problèmes d'existence : généralement formulé comme suit :
  - Existe-il ... ?
  - Est-il possible de ... ?
- Problèmes de construction : généralement formulé comme suit :
  - Si ... existe, comment pouvons-nous le construire ?
- Problèmes d'énumération : généralement formulé comme suit :
  - Combien y a-t-il de ... , et comment pouvons-nous les lister tous ?
- Problèmes d'optimisation : généralement formulé comme suit :

— S'il y a plusieurs ..., quel(le) est le/la meilleur(e) ?

Ainsi, le problème des sept ponts de Königsberg formulé par Euler, était un problème d'existence. La question était : existe-il une promenade dans les rues de Königsburg permettant, à partir d'un point de départ, de passer une et une seule fois par chaque pont, et de revenir à son point de départ (voir la figure 1.1).

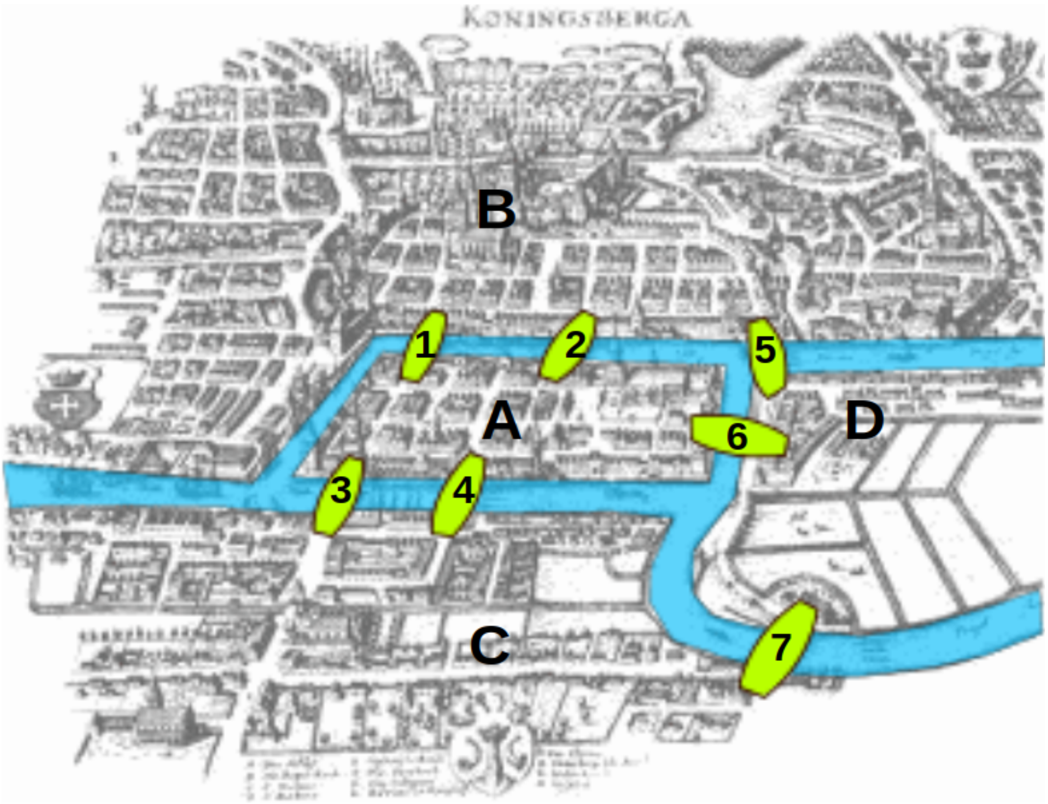


FIGURE 1.1 – Problème des sept ponts de Königsberg.

Ce problème peut être modélisé par un graphe non orienté où les points seraient les surfaces terrestres (zones hors de l'eau) et les liens seraient les ponts. La représentation graphique de ce graphe G_1 est donné à la figure 1.2.

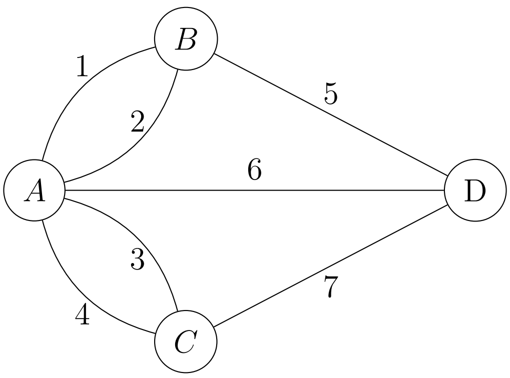


FIGURE 1.2 – Graphe G_1 modélisant le problème des sept ponts de Königsberg.

On remarque bien la simplification apportée par une telle représentation.

Il n'est tout de même pas encore évident de répondre à la question de l'existence ou non de la promenade pour l'instant. Nous y reviendrons vers la fin de ce chapitre.

Ce problème est à l'origine de la théorie des graphes.

1.2 Graphes non orientés

Un graphe non orienté $G = (X, U)$ est déterminé par deux ensembles X et U, avec :
- X : ensemble des points appelés **sommets**,
- U : ensemble des liens non orientés entre les paires non ordonnées de sommets appelés **arêtes**.

Exemple Le problème des sept ponts de Königsberg a été modélisé précédemment par un graphe non orienté $G_1 = (X_1, U_1)$ où :
- l'ensemble des sommets $X_1 = \{A, B, C, D\}$ représente l'ensemble des surfaces terrestres,
- l'ensemble des arêtes $U_1 = \{1, 2, 3, 4, 5, 6, 7\}$ représente l'ensemble des sept ponts.

Si $N = |X|$, on dit que N est l'**ordre** du graphe.

Exemple Le graphe G_1 est d'ordre 4.

Une arête u qui relie les deux sommets x et y est une paire non ordonnée de sommets et elle peut s'écrire donc $u = (x, y)$. On dit que :
- x et y sont les deux **extrémités** de u,
- u est **incidente** à x et à y,
- x et y sont **adjacents**.

Deux arêtes sont dites **parallèles** si elles possèdent les mêmes extrémités.

Exemple L'arête 6 du graphe G_1 précédent peut également être écrite sous la forme (A, B). A et B sont donc les extrémités de cette arête qui est elle même incidente à ces deux sommets. A et B sont adjacents.

Dans ce même graphe, les arêtes 1 et 2 sont parallèles car elles possèdent les mêmes extrémités A et B.

Remarque L'ensemble des sommets adjacents à un sommet x est noté par $\Gamma(x)$.

Exemple Dans le graphe précédant G_1, nous avons : $\Gamma(A) = \{B, C, D\}, \Gamma(B) = \{A, D\}, \Gamma(C) = \{A, D\}, \Gamma(D) = \{A, B, C\}$.

Une arête dont les deux extrémités sont identiques est appelée **boucle**.

Exemple La figure 1.3 représente un graphe G_2 comprenant un seul sommet et une boucle sur ce sommet.

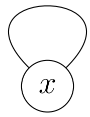


FIGURE 1.3 – Graphe G_2 comprenant un sommet et une boucle.

Le **degré** d'un sommet x est le nombre d'arêtes incidentes à x, avec les boucles comptées deux fois. On le note $d(x)$.

Exemple Dans le graphe G_1, $d(A) = 5$. Dans le graphe G_2, $d(x) = 2$.

Un graphe non orienté **simple** est un graphe sans boucle et sans arêtes parallèles.

Exemples

— Le graphe précédant G_1 est un graphe qui est non simple, car il existe deux arêtes parallèles entre les sommets A et B, et aussi deux arêtes parallèles entre les sommets A et C.

— Le graphe précédant G_2 est également un graphe qui est non simple, car il comporte une boucle.

— Le graphe représentant le réseau social Facebook où les sommets sont les comptes des utilisateurs et les arêtes sont les liens d'amitié entre les comptes est un graphe simple car un compte ne peut pas être ami avec lui-même, et deux comptes ne peuvent pas être amis plus d'une seule fois.

Un graphe non orienté **complet** est un graphe simple où il existe une arête entre chaque paire de sommets.

Un graphe non orienté complet d'ordre N est noté par K_N.

Exemple La figure 1.4 représente le graphe non orienté complet G_3 d'ordre 4, c.à.d. K_4.

Une **chaîne** est une séquence finie et alternée de sommets et d'arêtes, débutant et finissant par deux sommets appelés respectivement **extrémité initiale** et **extrémité terminale**, telle que chaque arête est incidente avec les sommets qui l'encadrent dans la séquence.

La **longueur** d'une chaîne est égale au nombre d'arêtes qui la composent, en comptant les

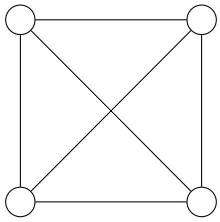


FIGURE 1.4 – Graphe G_3= K_4.

répétitions [1].

Une **chaîne élémentaire** est une chaîne qui ne parcourt pas plus d'une fois le même sommet.

Une **chaîne simple** est une chaîne qui ne parcourt pas plus d'une fois la même arête.

Remarques

— L'écriture complète d'une chaîne se fait en listant de manière ordonnée les sommets et les arêtes qui la composent.

— Une chaîne peut également être décrite uniquement par les arêtes qui la composent.

— De plus, dans le cas de graphes sans arêtes parallèles, une chaîne peut aussi être décrite uniquement par les sommets qui la composent.

Exemple Dans le graphe précédant G_1, nous avons par exemple la chaîne $(A, 1, B, 5, D, 7, C)$, qui est de longueur 3. Cette chaîne est simple et élémentaire.

Son extrémité initiale est le sommet A et son extrémité terminale est le sommet C.

Cette même chaîne peut aussi être décrite par ses arêtes uniquement comme suit : $(1, 5, 7)$.

Par contre, puisque G_1 n'est pas simple, cette chaîne ne peut pas être décrite uniquement par ses sommets, car en écrivant (A, B, D), on ne saurait pas s'il s'agit de la chaîne $(A, 1, B, 5, D, 7, C)$ ou bien $(A, 2, B, 5, D, 7, C)$.

Un **cycle** est une chaîne simple dont les extrémités sont identiques.

La **longueur** d'un cycle est égale au nombre d'arêtes qui le composent.

Exemple Dans le graphe G_1, nous avons par exemple le cycle $(A, 1, B, 5, D, 7, C, 4, A)$ qui est de longueur 4.

1.3 Graphes orientés

Les mêmes notions que pour les graphes non orientés sont reprises et adaptées au cas orienté dans cette section.

Un graphe orienté $G = (X, U)$ est déterminé par les deux ensembles X et U, avec :

- X : ensemble de points appelés **sommets**,
- U : ensemble de liens orientés entre des couples ordonnés de sommets appelés **arcs**.

$N = |X|$ est l'**ordre** du graphe.

Un arc u qui relie le sommet x au sommet y est un couple ordonné qui peut donc s'écrire $u = (x, y)$. On dit que :

- x est **l'extrémité initiale** de u,
- y est **l'extrémité terminale** de u,
- y est **successeur** de x et x est **prédécesseur** de y,
- u est **incident** à x et à y.

Deux arcs ayant la même extrémité initiale et la même extrémité terminale sont dits **parallèles**.

Un arc dont les deux extrémités sont identiques est appelé **boucle**.

Remarque Un arc est une arête à laquelle on précise un sens/une orientation.

Exemple Soit le graphe orienté G_4= (X_4, U_4) où l'ensemble des sommets est $X_4 = (1, 2, 3)$, l'ensemble des arcs est $U_4 = (a, b, c, d, e)$, et la représentation graphique est donnée à la figure 1.5.

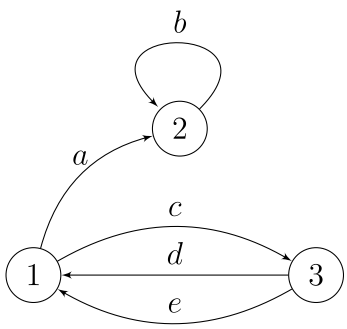


FIGURE 1.5 – Graphe orienté G_4.

G_4 est d'ordre 3 étant donné qu'il comporte 3 sommets.

L'arc $a = (1, 2)$ possède le sommet 1 comme extrémité initiale et le sommet 2 comme extrémité terminale. Le sommet 2 est donc successeur du sommet 1 et le sommet 1 est prédécesseur du sommet 2.

Les deux arcs d et e sont des arcs parallèles puisqu'ils ont la même extrémité initiale 3 et la même extrémité terminale 1. Les arcs c et d ne le sont pas.

L'arc b est une boucle.

Remarque L'ensemble des successeurs d'un sommet x est noté par $\Gamma(x)$ ou bien $\Gamma^+(x)$ alors que l'ensemble des ses prédécesseurs est noté par $\Gamma^-(x)$.

Exemple Dans le graphe G_4, nous avons :
- $\Gamma^+(1) = \{2, 3\}$ et $\Gamma^-(1) = \{3\}$
- $\Gamma^+(2) = \{2\}$ et $\Gamma^-(2) = \{1, 2\}$
- $\Gamma^+(3) = \{1\}$ et $\Gamma^-(3) = \{1\}$

Dans un graphe orienté :
- Le **demi-degré extérieur** d'un sommet x est le nombre d'arcs ayant x comme extrémité initiale. Il est noté $d^+(x)$,
- Le **demi-degré intérieur** d'un sommet x est le nombre d'arcs ayant x comme extrémité terminale. Il est noté $d^-(x)$,
- Le **degré** d'un sommet x est le nombre d'arcs incidents à x, avec les boucles comptées deux fois. Il est noté $d(x)$. Nous avons $d(x) = d^+(x) + d^-(x)$.

Exemple Dans le graphe précédant G_4, $d^+(2) = 1, d^-(2) = 2, d(2) = 3$.

Un **graphe orienté simple** est un graphe sans boucle et sans arcs parallèles.

Un **graphe orienté complet** est un graphe orienté simple où il existe un arc entre chaque couple ordonné de sommets.

Exemple
- Le graphe précédant G_4 est un graphe orienté non simple car il comporte une boucle, de plus, il y a deux arcs parallèles d et e. Si les arcs b et e étaient retirés, alors le graphe obtenu serait simple.
- Le graphe G_5 représenté à la figure 1.6 est un graphe orienté complet d'ordre 4.

Un **chemin** est une chaîne où tous les arcs sont parcourus dans le bon sens.

La **longueur** d'un chemin est égale au nombre d'arcs qui le composent, en comptant les répétitions.

Un **chemin élémentaire** est un chemin qui ne parcourt pas plus d'une fois le même sommet.

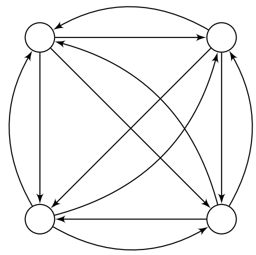


FIGURE 1.6 – Graphe orienté complet G_5.

Un **chemin simple** est un chemin qui ne parcourt pas plus d'une fois le même arc.

Exemples Dans le graphe G_4 :

— Nous avons le chemin simple $(1, c, 3, d, 1, a, 2, b, 2)$. Sa longueur est égale à 4. Ce chemin est non élémentaire car le sommet 2 est parcouru deux fois.
— $(2, a, 1, c, 3)$ est une chaîne mais pas un chemin car l'arc a est parcouru dans le mauvais sens.
— $(1, c, 3, d, 1, c, 3)$ est un chemin non simple car l'arc c est parcouru 2 fois.

Un **circuit** est un chemin dont les deux extrémités sont identiques.

La **longueur** d'un circuit est égale au nombre d'arcs qui le composent, en comptant les répétitions.

Exemples Dans le graphe G_4 :

— $(1, c, 3, d, 1)$ est un circuit de longueur 2.
— $(2, b, 2, b, 2)$ est également un circuit de longueur 2.

1.4 Lemme des poignées de main

Lemme : Dans un graphe $G = (X, U)$ nous avons toujours $\sum_{x \in X} d(x) = 2 \times |U|$.

Démonstration. Chaque arête/arc $u \in U$ est compté(e) deux fois dans la somme $\sum_{x \in X} d(x)$: une fois par extrémité. □

Exemple Cette propriété est valide pour tout graphe. À titre d'exemple, dans le graphe G_1, la somme des degrés de tous les sommets est $\sum_{x \in X} d(x) = d(A) + d(B) + d(C) + d(D) = 5 + 3 + 3 + 3 = 14$ et le nombre d'arêtes du graphe est $|U| = 7$. La propriété est donc bien vérifiée.

Théorème : Dans un graphe, il y a toujours un nombre pair de sommet à degré impair.

Démonstration. À réaliser lors du TD. □

Exemple Dans le graphe G_1 il y a exactement 4 sommets de degré impair. Il y a donc bien un nombre pair de sommets à degré impair.

1.5 Sous-graphe, graphe partiel et graphe complémentaire

Un **sous-graphe** du graphe $G = (X, U)$ est un graphe $G' = (X', U')$ tel que $X' \subseteq X$ et $U' \subseteq U$.

Un **graphe partiel** du graphe $G = (X, U)$ est un graphe $G' = (X, U')$ tel que $U' \subseteq U$.

Le **graphe complémentaire** du graphe simple $G = (X, U)$ est un graphe simple $\overline{G} = (X, \overline{U})$ tel que $(x, y) \in \overline{U}$ si et seulement si $(x, y) \notin U$.

Exemples Soient les graphes G_6, G_7, G_8 et G_9 de la figure 1.7. Nous avons :

— Le graphe G_7 est un graphe partiel du graphe G_6. Il est également sous-graphe du graphe G_6.

— Le graphe G_8 est sous-graphe du graphe G_6 ainsi que sous-graphe du graphe G_7.

— Le graphe G_9 est le graphe complémentaire du graphe G_6.

1.6 Connexité et forte connexité

Un graphe non orienté $G = (X, U)$ est dit **connexe** s'il existe au moins une chaîne entre chaque paire de sommets distincts $x, y \in X$.

Une **composante connexe** est un sous-graphe connexe maximal [a].

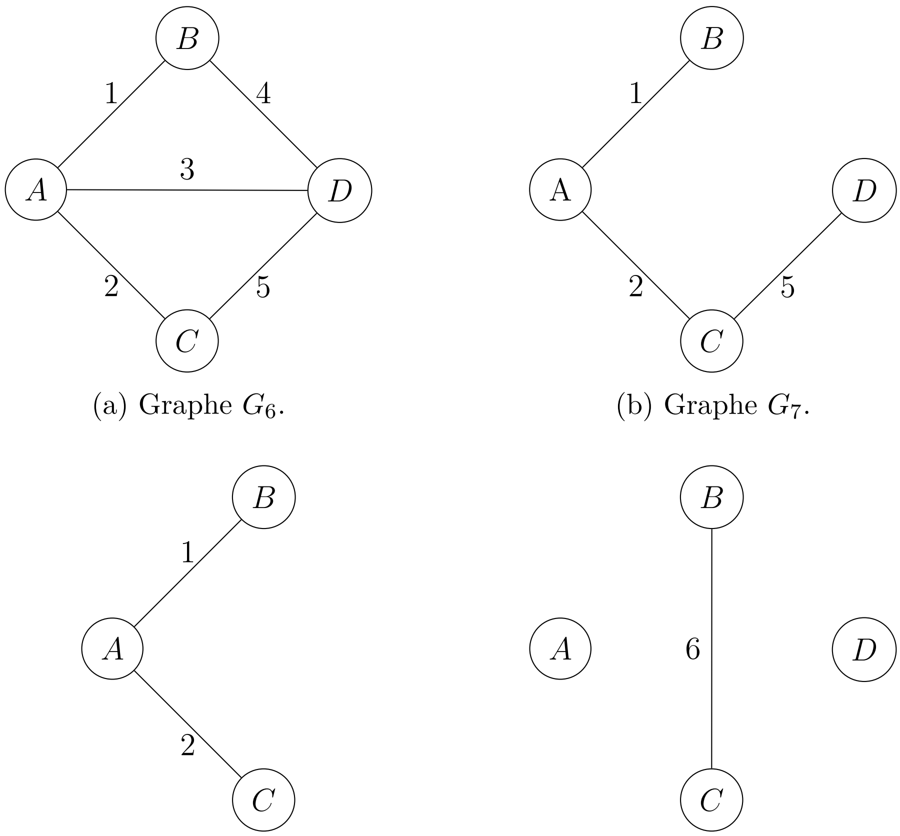
Le **nombre de connexité** d'un graphe est le nombre de ses composantes connexes. Il est noté p.

a. Un sous-graphe est maximal par rapport à la connexité s'il n'est pas inclus dans un autre sous-graphe connexe.

Exemples

— Parmi les graphes de la figure 1.7, G_6, G_7 et G_8 sont des graphes connexes. Ils comportent donc une seule composante connexe et leur nombre de connexité $p = 1$.

— Le graphe G_9 est quant à lui non connexe, et il comporte 3 composantes connexes composées respectivement des sommets $\{A\}, \{B, C\}$ et $\{D\}$. Son nombre de connexité $p = 3$.



(a) Graphe G_6. (b) Graphe G_7.

(c) Graphe G_8. (d) Graphe G_9.

FIGURE 1.7 – Exemples de sous-graphe, graphe partiel et graphe complémentaire.

Un graphe orienté est dit **fortement connexe** si pour toute paire de sommets $x, y \in X$, il existe au moins un chemin de x vers y et au moins un chemin de y vers x.

Une **composante fortement connexe** est un sous-graphe fortement connexe maximal.

Exemples Le graphe orienté G_4 vu précédemment est non fortement connexe car il n'existe par exemple, aucun chemin du sommet 2 au sommet 3. G_4 comporte deux composantes fortement connexe $\{1, 3\}$ et $\{2\}$.

La modification de ce graphe en rajoutant un arc du sommet 2 au sommet 3 produirait le graphe G_{10} donné à la figure 1.8 qui est un graphe fortement connexe.

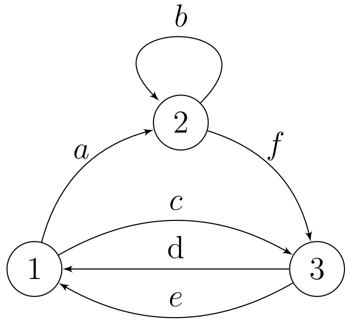


FIGURE 1.8 – Graphe fortement connexe G_{10}.

1.7 Graphe eulérien

Une **chaîne eulérienne** est une chaîne qui parcourt toutes les arêtes d'un graphe une et une seule fois.

Un **cycle eulérien** est une chaîne eulérienne qui forme un cycle.

Exemple Le graphe G_{11}, ci-dessous, contient la chaîne eulérienne : $(a, b, d, i, e, f, h, g, c)$. Cette chaîne forme un cycle, c'est donc également un cycle eulérien.

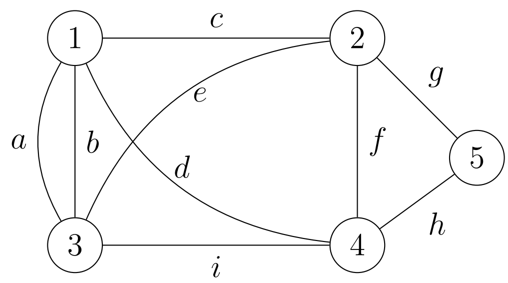


FIGURE 1.9 – Graphe G_{11} avec cycle eulérien.

Un **graphe semi-eulérien** est un graphe qui contient une chaîne eulérienne.

Un **graphe eulérien** est un graphe qui contient un cycle eulérien.

Exemple Le graphe précédent G_{11} contient un cycle eulérien. Ce graphe est donc un graphe eulérien. Il est également semi-eulérien.

Théorème : Un graphe connexe est semi-eulérien si et seulement s'il contient exactement 0 ou 2 sommets de degré impair. Il est eulérien si et seulement s'il contient 0 sommet de degré impair.

Exemple Si on vérifie le degré de tous les sommets du graphe eulérien précédent G_{11}, on peut remarquer que tous ses sommets sont bien de degré pair.

Si on supprime l'arête c, on obtient un nouveau graphe avec exactement 2 sommets à degré impair. Ce nouveau graphe serait donc non eulérien mais il serait semi-eulérien.

Si on revient au problème introductif de notre cours, qui est le problème des ponts de Königsberg (voir la section 1.1), et qui consiste à déterminer s'il est possible de trouver un promenade qui parcourt tous les ponts de la ville une et une seule fois et qui revient au point de départ, on peut constater qu'il s'agit finalement de déterminer si le graphe correspondant, le graphe G_1, est eulérien, ou non.

On peut constater que G_1 possède des sommets à degré impair (ses 4 sommets sont tous à degré impair). Ce graphe n'est donc pas eulérien. On peut donc en déduire que la promenade recherchée n'existe pas.

1.8 Matrice d'adjacence

Pour un graphe $G = (X, U)$ d'ordre N et sans arêtes/arcs parallèles, une **matrice d'adjacence** A est une matrice *booléenne carrée* de taille $N \times N$ et qui est construite de la manière suivante :

$$\forall i, j \in 1 \ldots N, a_{ij} = \begin{cases} 1 \text{ si } (x_i, x_j) \in U \\ 0 \text{ sinon} \end{cases}$$

Exemples Soit le graphe non orienté G_{12} représenté graphiquement à la figure 1.10.

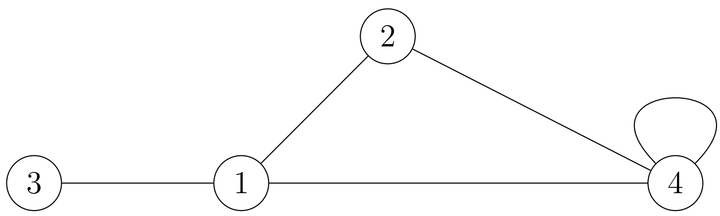


FIGURE 1.10 – Graphe G_{12}.

La matrice d'adjacence de G_{12} est donnée comme suit :

$$A = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Soit maintenant le graphe orienté G_{13} obtenu en retirant l'arc d du graphe G_4 afin d'éliminer les arcs parallèles et dont la représentation graphique est donnée à la figure 1.11 :

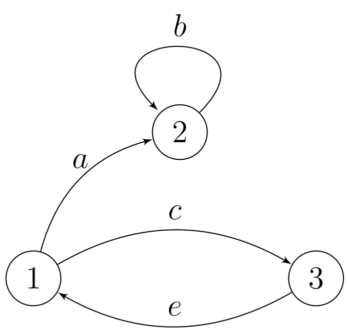


FIGURE 1.11 – Graphe G_{13} sans arcs parallèles.

Sa matrice d'adjacence est la suivante :

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \end{array}$$

Remarques

- Pour les graphes non orientés :
  - A est symétrique
  - $\forall i \in 1 \ldots N : \quad 2a_{ii} + \sum_{\substack{j=1\ldots N \\ j \neq i}} a_{ij} = 2a_{ii} + \sum_{\substack{j=1\ldots N \\ j \neq i}} a_{ji} = d(x_i)$
  - $\sum_{i=1\ldots N} a_{ii}$ = nombre de boucles dans le graphe
- Pour les graphes orientés :
  - A n'est pas nécessairement symétrique
  - $\forall i \in 1 \ldots N, \sum_{j=1\ldots N} a_{ij} = d^+(x_i)$
  - $\forall i \in 1 \ldots N, \sum_{j=1\ldots N} a_{ji} = d^-(x_i)$
  - $\forall i \in 1 \ldots N, \sum_{j=1\ldots N} (a_{ij} + a_{ji}) = d(x_i)$
  - $\sum_{i=1\ldots N} a_{ii}$ = nombre de boucles dans G

Exemple En ce qui concerne le graphe orienté G_{13}, on remarque que pour sa matrice d'adjacence, la somme des éléments de la ligne du sommet 1 donne bien son demi-degré extérieur ($\sum_{j=1\ldots N} a_{1j} = d^+(1) = 2$). De même que la somme des éléments de la colonne du sommet 1 donne bien son demi-degré intérieur ($\sum_{j=1\ldots N} a_{j1} = d^-(1) = 1$). Le degré de ce sommet est la somme des deux valeurs précédentes.

Le nombre de boucles de ce graphe est égal à 1 et il correspond bien à la somme des éléments de la diagonale de A.

1.9 Exercices

Exercice 1

1. Quel type de relations sont modélisées par des graphes non orientés ?

2. Un graphe non orienté est déterminé par deux ensembles. Quels sont-ils ?

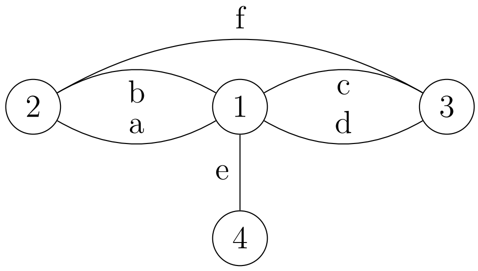
3. Parmi les notions suivantes, quelles sont celles qui concernent uniquement les graphes orientés :
   1. Arc
   2. Cycle
   3. Circuit

   4. Chaîne
   5. Connexité
   6. Graphe réduit
   7. Forte connexité

4. Répondre par vrai ou faux :
   1. Un graphe simple d'ordre 10 peut contenir 9 arcs.
   2. Un cycle est toujours un circuit.
   3. Une matrice d'adjacence de graphe est toujours symétrique.
   4. Un graphe fortement connexe d'ordre 5 peut contenir un sommet à degré égal à 0.
5. Une arête peut relier au minimum combien de sommets dans un graphe ?
6. Existe-il un graphe non orienté simple dont les sommets ont pour degrés 1, 2, 2, 2, 5 ? Justifier.
7. Un graphe complet est-il simple ?
8. Quel est le nombre d'arêtes du graphe complémentaire d'un graphe complet d'ordre 5 ?
9. Quelle est la longueur maximale possible d'une chaîne élémentaire dans un graphe d'ordre 5 ?
10. Quelle est la longueur maximale possible d'une chaîne non-élémentaire dans un graphe d'ordre 5 ?
11. Quel est le nombre de connexité d'un graphe connexe ?
12. Quel est le nombre minimal de sommets adjacents à un sommet quelconque dans un graphe connexe d'ordre $N \geq 2$?
13. Quels sont les graphes complets d'ordre N qui admettent un cycle eulérien ?
14. Donner, si c'est possible, un exemple de graphe qui est eulérien mais non connexe.
15. La diagonale de la matrice d'adjacence d'un graphe G contient exactement 4 éléments non nuls. Que peut-on déduire ?
16. Donner le nombre d'arcs dans un graphe orienté G en fonction des éléments de sa matrice d'adjacence A.

Exercice 2

Soit le graphe $G = (X, U)$ représenté graphiquement ci-dessous :



1. Donner :

   a) Les ensembles X et U.

   b) L'ordre de G.

   c) Le degré de chaque sommet dans X.

2. Que faudrait-il modifier dans G afin d'obtenir un graphe simple ?

Exercice 3

1. Tracer tous les graphes non orientés simples possibles dont l'ensemble des sommets est $\{a, b, c\}$. Organiser les graphes de manière à ce que chaque graphe soit représenté à coté de son graphe complémentaire .

2. Tracer un graphe orienté fortement connexe à 5 sommets et 5 arcs.

3. Tracer un graphe orienté sans circuit à 5 sommets et 10 arcs.

Exercice 4

Soit le graphe orienté $G = (X, U)$ avec $X = \{1, 2, 3, 4\}$ et $\Gamma(1) = \{2\}, \Gamma(2) = \{2\}, \Gamma(3) = \{1\}, \Gamma(4) = \{\}$.

1. Tracer le graphe G.

2. Donner le degré de chaque sommet dans G.

Exercice 5

Un graphe non orienté simple possède 15 arêtes, 3 sommets de degré 4 et tous les autres sommets de degré 3. Quel est l'ordre de ce graphe ?

Exercice 6

Est-il possible de tracer des graphes non orientés simples dont les sommets ont les degrés suivants (justifier) ?

1. 5,2,1
2. 2,2,2
3. 1,1,1

Exercice 7

Une ligue de football comprend 7 clubs. Pour des raisons de temps, on décide que chaque club ne jouera que 3 matches chacun. Est-ce possible ? Justifier.

Quel est le plus petit nombre de matches que doit jouer chaque club pour que cela soit possible ? Justifier.

Exercice 8

1. Combien y a-t-il d'arêtes dans un graphe non orienté complet à 4 sommets (K_4) ?
2. Combien y a-t-il de graphes non orientés simples possibles ayant 4 sommets ?
3. Combien y a-t-il de graphes non orientés possibles ayant 4 sommets ?
4. Combien y a-t-il d'arcs dans un graphe orienté complet à 4 sommets ?
5. Combien y a-t-il de graphes orientés simples possibles ayant 4 sommets ?
6. Combien y a-t-il de graphes orientés possibles ayant 4 sommets ?

Exercice 9

Considérons le réseau social Facebook. En supposant que celui-ci comporte N utilisateurs :

1. Quel est le nombre maximal de relations de type "Amitié" dans ce réseau ?
2. Quel est le nombre maximal de relations de type "Abonnement" dans ce réseau ?
3. L'ensemble des étudiants l'école et leurs relations d'amitié sur Facebook représentent quelle notion par rapport au réseau Facebook entier ?

Exercice 10

Un petit pays composé de 9 villes souhaite mettre en place un réseau de transport moderne. Dans un premier temps, il est proposé de mettre en place une ligne de bus directe entre chaque ville et toutes les autres.

1. Combien cela fait-il de lignes ?

Le budget du pays étant trop faible, toutes les lignes ne peuvent être mises en place. Après négociation, chaque ville arrive à obtenir exactement 5 lignes directes qui partent de cette ville.

2. Proposer un schéma de réseau de bus possible pour ce pays.

Exercice 11

Montrer que dans un graphe simple d'ordre N, il y a toujours au moins 2 sommets de même degré.

Exercice 12

On souhaite convertir de l'argent d'une devise vers une autre. Le tableau suivant donne la fonction f qui exprime les taux de change possibles entre 4 devises. Par exemple : étant donné que $f(D_1, D_2) = 0,01$, la conversion de 400 unités de la devise D_1 donnerait $400 \times 0,01 = 4$ unités en devise D_2. Toutes les conversions ne sont pas possibles : pour deux monnaies D_i et D_j, on peut parfois convertir de l'argent de D_i en D_j, parfois non.

	D_1	D_2	D_3	D_4
D_1	-	0,01	-	-
D_2	100	-	1,11	0,85
D_3	91	0,9	-	0,7
D_4	-	-	1,43	-

On considère un graphe de change qui modélise les conversions possibles.

1. Quel est le type de ce graphe ?

2. Tracer ce graphe.

Une séquence de change est la conversion d'une monnaie D_i en monnaie D_j en passant par des monnaies intermédiaires.

3. À quoi correspond cette séquence dans le graphe ?

4. Quel est la valeur du taux de change d'une telle séquence ?

5. Sous quelle condition (exprimée sur G) quelqu'un peut-il devenir infiniment riche en changeant de l'argent ? Donner un exemple d'après le graphe obtenu, si un tel exemple existe.

Exercice 13

Un réseau maillé sans fil est constitué d'appareils (ordinateurs, téléphones, etc.) qui peuvent communiquer de manière directe s'ils sont assez proches. Dans ces réseaux, les communications peuvent également se faire de manière indirecte en multi-saut.

On suppose que, dans un certain réseau maillé sans fil, tout appareil est assez proche d'au moins la moitié des appareils du réseau. Prouver que toute paire d'appareils peut alors communiquer dans ce réseau.

Exercice 14

Soit le graphe connexe G où le degré de chaque sommet est pair. Prouver qu'après suppression de n'importe quelle arête de G, on obtient un nouveau graphe G' qui est nécessairement connexe.

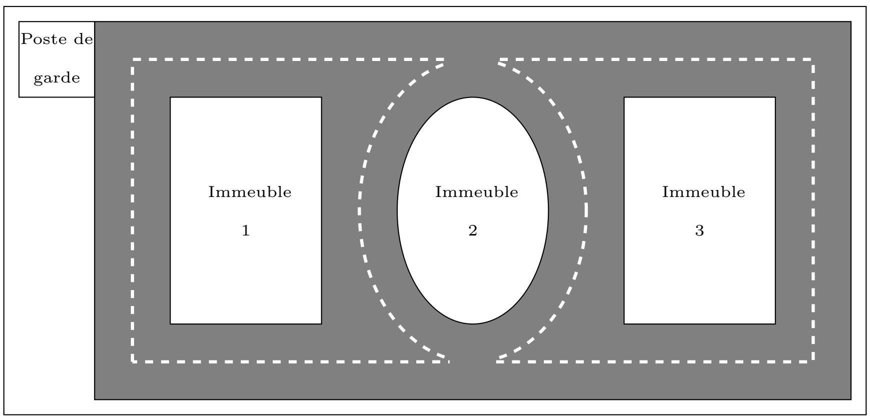
Exercice 15

Un facteur désire faire sa tournée en passant une seule fois par chaque rue à revenir à son point de départ.

1. Est-ce possible dans le deux cas suivants ?
2. Lorsque la réponse à la précédente question est oui, proposer une tournée possible.

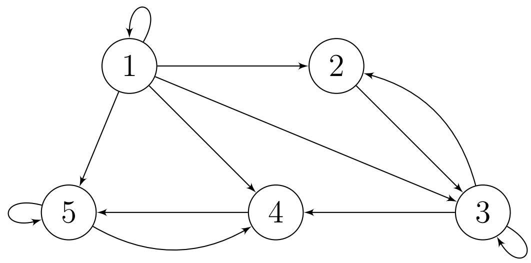
Exercice 16

Un agent de sécurité doit faire sa ronde en parcourant toutes les rues d'un quartier résidentiel possédant 3 immeubles (voir figure ci-après). Il souhaite assurer cela en marchant le moins possible. La meilleure solution serait qu'il suive un itinéraire, qui commencerait et qui se terminerait au poste de garde, et où il ne parcourrait pas une même rue plus d'une fois. Est-ce qu'un tel itinéraire existe ?



Exercice 17

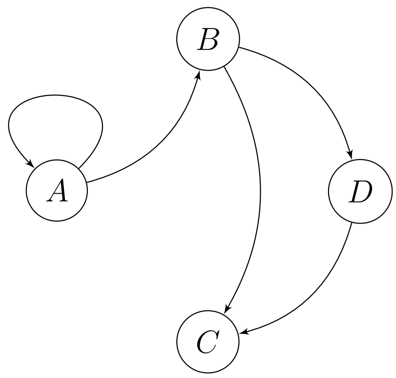
Soit le graphe orienté $G = (X, U)$ suivant :



Donner la matrice d'adjacence du graphe G.

Exercice 18

Soit le graphe orienté G tracé ci-dessous :



1. Donner le demi-degré sortant de chaque sommet de G.
2. Donner la matrice d'adjacence de G.
3. Donner un circuit de longueur égale à 3 dans G.

Exercice 19

Soit un graphe non orienté G ayant la matrice A suivante comme matrice d'adjacence :

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

1. Tracer G.

2. Donner le nombre de connexité de G, c.à.d. p.

3. Donner une modification minimale à apporter à A afin d'obtenir un nouveau graphe G_1 qui ne soit pas simple.

4. Donner une modification minimale à apporter à A afin d'obtenir un nouveau graphe G_2 qui a un nombre de connexité supérieur à p ?

5. Donner la matrice d'adjacence de $\overline{G}$, le graphe complémentaire de G.

Exercice 20

Soit A la matrice d'adjacence d'un graphe non orienté $G = (X, U)$:

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

Donner :

1. L'ordre de G.

2. Le degré de chaque sommet dans X.

1.10 Solutions des exercices

Solution de l'exercice 1

1. Les relations dans les graphes non orientés modélisent des relations réciproques.

2. Un graphe non orienté est déterminé par un ensemble de sommets et un ensemble d'arêtes.

3. Les notions qui concernent uniquement les graphes orientés sont :
   — Arc
   — Circuit
   — Graphe réduit
   — Forte connexité

4. 1. Vrai.
   2. Faux.
   3. Faux.
   4. Faux.

5. Une arête relie au minimum un sommet à lui même (cas d'un boucle).

6. Non, il n'existe pas un graphe non orienté simple dont les sommets ont pour degrés 1, 2, 2, 2, 5, car le degré maximum d'un sommet dans un graphe non orienté simple d'ordre 5 est égal à 4, on ne peut donc pas avoir un sommet de degré 5.

7. Oui, un graphe complet est simple par définition.

8. Le nombre d'arêtes du graphe complémentaire d'un graphe complet d'ordre 5 est égal à 0.

9. La longueur maximale possible d'une chaîne élémentaire dans un graphe d'ordre 5 est 4.

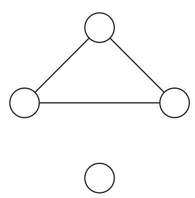
10. La longueur maximale possible d'une chaîne non-élémentaire dans un graphe d'ordre 5 est l'infini (il n'y a pas de limite).

11. Le nombre de connexité d'un graphe connexe est 1.

12. Dans un graphe connexe d'ordre $N \geq 2$, un sommet est adjacent à au moins un autre sommet (le minimum est donc égal à 1).

13. Les graphes complets d'ordre N qui admettent un cycle eulérien sont ceux où N est impair.

14. Graphe eulérien mais non connexe :



15. La diagonale de la matrice d'adjacence d'un graphe G contient exactement 4 éléments non nuls. On peut en déduire que le graphe contient 4 boucles.

16. $\sum_{i,j=1 \ldots N} a_{ij}$, avec N ordre du graphe.

Solution de l'exercice 2

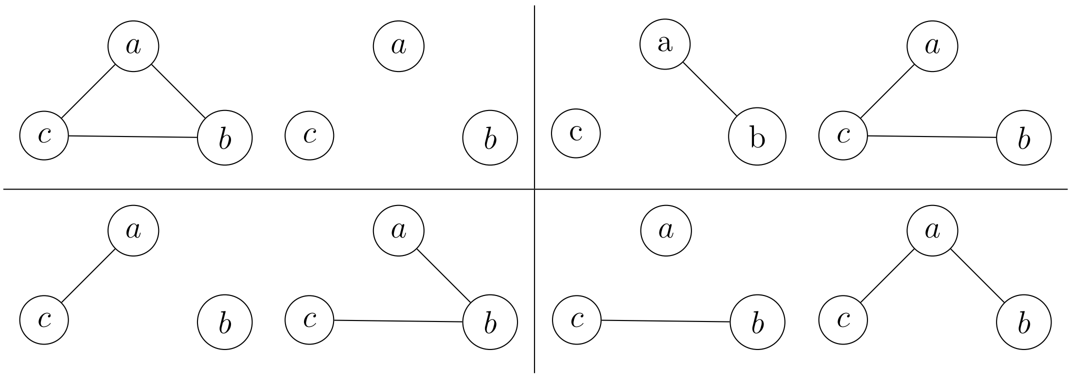
1. a) $X = \{1, 2, 3, 4\}$, $U = \{a, b, c, d, e, f\}$.

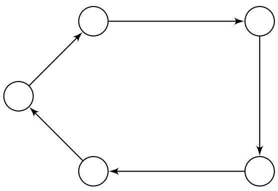
b) L'ordre de G est égal à 4.

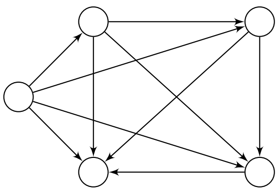
c) $d(1) = 5, d(2) = 3, d(3) = 3, d(4) = 1$.

2. Afin d'obtenir un graphe simple, il faudrait supprimer les arêtes parallèles, par exemple : a et d.

Solution de l'exercice 3

1. 

2. 

3. 

Solution de l'exercice 4

1. Le diagramme du graphe :

2. Les degrés des sommets sont : $d(1) = 2$, $d(2) = 3$, $d(3) = 1$, $d(4) = 0$.

Solution de l'exercice 5

Soit le graphe non orienté simple $G = (X, U)$ d'ordre N qui possède 15 arêtes, 3 sommets de degré 4 et tous les autres sommets de degré 3.

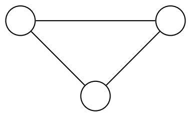
Nous avons $\sum_{x \in X} d(x) = 2|U|$

$\Rightarrow 3 \times 4 + (N - 3) \times 3 = 2 \times 15$

$\Rightarrow N = 9$

Solution de l'exercice 6

1. Non, il n'est pas possible de tracer un graphe simple dont les degrés des sommets sont 5,2,1 car dans un graphe simple d'ordre N, le degré maximal d'un sommet est égal à $N - 1$. Nous ne pouvons donc pas avoir un graphe simple d'ordre 3 avec un sommet d'ordre 5.
2. Oui, on peut tracer un graphe simple dont les sommets ont les degrés 2,2,2. La preuve est le diagramme donné ci-dessous :

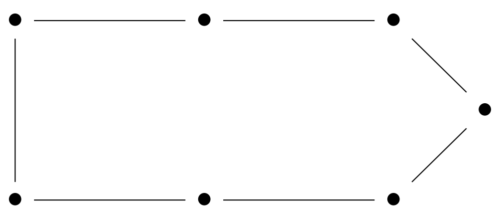


3. Non, il n'est pas possible de tracer un graphe simple dont les degrés des sommets sont 1,1,1 car on aurait un graphe à 3 sommets à degré impair, or, d'après le théorème des poignées de mains, le nombre de sommets à degré impair dans un graphe est toujours pair.

Solution de l'exercice 7

Le problème consiste à tenter de construire un graphe non orienté d'ordre 7, où le degré de chaque sommet serait égal à 3. Nous aurions donc un nombre impair de sommets à degré impair, ce qui est impossible car tout graphe comporte un nombre pair de sommets à degré impair.

Jouer 1 seul match est impossible pour la même raison que précédemment. Le plus petit nombre de matchs que doit jouer chaque club est donc 2. La preuve est l'existence d'un tel graphe :



Solution de l'exercice 8

1. Dans le graphe K_4, le degré de chaque sommet x est égal à $d(x) = 3$. La somme des degrés est égale à $3 \times 4 = 12$. En appliquant le lemme des poignées de mains, $\sum_{x \in X} d(x) = 2 \times |U|$, on a $|U| = 12/2 = 6$. Il y a donc 6 arêtes.

2. Dans graphe non orienté simple d'ordre 4, chacune des 6 arêtes possibles, existe ou non. Il y a donc $2^6 = 64$ graphes possibles.

3. Il y a une infinité de graphes non orientés d'ordre 4.

4. Dans un graphe orienté complet d'ordre 4, le degré de chaque sommet est égal à 6 (demi degré extérieur $d^+(x) = 3$ et demi degré intérieur $d^-(x) = 3$). La somme des degrés est égale à $6 \times 4 = 24$. En appliquant le lemme des poignées de mains, $\sum_{x \in X} d(x) = 2 \times |U|$, on a $|U| = 24/2 = 12$. Il y a donc 12 arcs dans le graphe.

5. Dans graphe orienté simple d'ordre 4, chacun des 12 arcs possibles, existe ou non. Il y a donc $2^{12} = 4096$ graphes possibles.

6. Il y a une infinité de graphes orientés d'ordre 4.

Solution de l'exercice 9

1. La relation "Amitié" peut être modélisée par les arêtes d'un graphe non orienté simple d'ordre N. Un tel graphe avec le nombre maximal d'arêtes est un graphe complet K_N. En appliquant le même principe que pour l'exercice précédant, son nombre d'arêtes est $((N-1) \times N)/2$.

2. La relation "Abonnement" peut être modélisée par les arcs d'un graphe orienté simple d'ordre N. Un tel graphe avec le nombre maximal d'arcs est un graphe complet. En appliquant le même principe que pour l'exercice précédant, son nombre d'arcs est $(((N-1) \times 2) \times N)/2 = (N-1) \times N$.

3. L'ensemble des comptes des étudiants représente un sous ensemble de l'ensemble de tous les comptes sur Facebook. Idem pour l'ensemble des arêtes d'amitiés. Ceci représente donc un sous-graphe par rapport au réseau Facebook entier.

Solution de l'exercice 10

1. Le problème peut être modélisé par un graphe non orienté complet d'ordre 9. Le nombre de lignes correspond au nombre d'arêtes et est donc égal à $9 \times (9-1)/2 = 36$.

2. Il tel schéma n'existe pas car il impliquerait un graphe d'ordre 9 (impair) où le degré de chaque sommet serait égal à 5 (impair), or, dans un graphe, il y toujours un nombre pair de sommets à degré impair.

Solution de l'exercice 11

On veut montrer que dans un graphe simple d'ordre N, il y a toujours au moins 2 sommets de même degré.

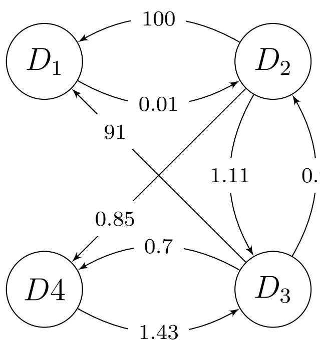
On raisonne par l'absurde. On suppose donc qu'il existe un graphe simple d'ordre N où tous les sommets ont un degré différent les uns des autres. Les degrés des sommets ordonnés de manière croissante seraient : $0, 1, 2, \ldots, N-1$. On a alors un sommet de degré 0 (n'est relié avec aucun autre sommet), et un sommet de degré $N-1$ (est relié avec tous les autres sommets). Contradiction !

On en conclut que dans un graphe simple, il y au moins 2 sommets de même degré.

Solution de l'exercice 12

1. La graphe est un graphe orienté valué.

2. Le graphe :



3. Une séquence de change correspond à un chemin qui relie D_i à D_j en parcourant les différentes devises intermédiaires.

4. La valeur du taux de change d'une telle séquence correspond au produit des valeurs associées aux arcs parcourus sur le chemin. Appelons cette valeur v.

5. La condition sous laquelle quelqu'un peut devenir infiniment riche en changeant de l'argent est l'existence d'un circuit dont la valeur v est supérieure à 1. Exemple : (D_2, D_4, D_3, D_2) dont le v est égal à $0,85 \times 1,43 \times 0{,}9 = 1,09395$.

Solution de l'exercice 13

Le problème peut être modélisé par un graphe non orienté simple $G = (X, U)$ d'ordre N avec :

— X : ensemble des appareils

— U : arêtes qui représentent la relation de communication directe entre les appareils.

On nous donne : $\forall x \in X, d(x) \geq N/2$.

On doit prouver que G est connexe, c.à.d. $\forall (x, y) \in X^2$, il existe une chaîne reliant x et y.

On raisonne par l'absurde, on suppose que G n'est pas connexe, c.à.d. $\exists (x, y) \in X^2$ tel qu'il n'existe pas de chaîne reliant x et $y \Rightarrow x$ et y font partie de deux composantes connexes différentes C_x et C_y, d'ordre N_x et N_y respectivement.

Étant donné que $d(x) \geq N/2$ et $d(y) \geq N/2$, on obtient $N_x \geq N/2+1$ et $N_y \geq N/2+1 \Rightarrow N_x + N_y \geq N+2 > N$. Contradiction ! (l'union des deux composantes connexes contiendrait plus de sommets que le graphe lui même).

G est donc connexe.

Solution de l'exercice 14

On raisonne par l'absurde. On suppose que la suppression d'une arête (i,j) donne un graphe G' non connexe. Cela impliquerait que G' possède 2 composantes connexes : C_i qui contient le sommet i et C_j qui contient le sommet j. Nous aurions alors un seul sommet à degré impair dans le sous-graphe C_i (même chose pour C_j), ce qui est absurde. G' est donc forcément connexe.

Solution de l'exercice 15

1. Afin que le facteur puisse effectuer sa tournée en passant une seule fois par chaque rue et revenir à son point de départ, il faudrait que le graphe correspondant soit eulérien. Pour cela, il suffit de vérifier, selon le théorème d'Euler, si chaque sommet est de degré pair (étant donné que les deux graphes sont connexes). Ceci n'est pas vérifié pour le graphe (a) mais l'est pour le graphe (b).

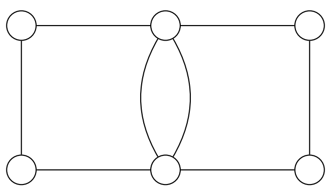
2. Un exemple de tournée possible dans le graphe (b) : (1,3,5,4,3,2,4,1) qui est un cycle eulérien.

Solution de l'exercice 16

La situation peut être modélisée par un graphe non orienté $G = (X, U)$ avec :

X **:** Ensemble des intersections ;

U **:** Ensemble des ruelles.



Le problème consiste à déterminer si G est eulérien. Selon le théorème d'Euler, la réponse est oui étant donné que chaque sommet dans ce graphe connexe est de degré pair. Un itinéraire, qui commencerait et qui se terminerait au poste de garde, et où l'agent ne parcourrait pas une même rue plus d'une fois existe.

Solution de l'exercice 17

La matrice d'adjacence du graphe G est :

$$A = \begin{array}{cc} & \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

Solution de l'exercice 18

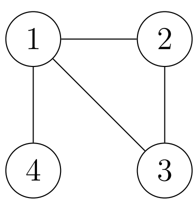
1. Les demi-degrés sortants des sommets de G sont : $d^+(A) = 2$, $d^+(B) = 2$, $d^+(C) = 0$, $d^+(D) = 1$.

2. La matrice d'adjacence de G est :

$$M = \begin{array}{cc} & \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{array}{c} A \\ B \\ C \\ D \end{array} & \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

3. Un circuit de longueur 3 dans G : (A,A,A,A).

Solution de l'exercice 19

1. 

2. Le nombre de connexité de G est $p = 1$.

3. $A_1 = \begin{array}{cc} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$

4. $A_2 = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$

5. $\overline{A} = \begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$

Solution de l'exercice 20

1. L'ordre de G est 5.

2. Le degré de chaque sommet dans X : $d(1) = 3, d(2) = 2, d(3) = 4, d(4) = 3, d(5) = 2$.

CHAPITRE 2

ALGORITHMES DE BASE EN THÉORIE DES GRAPHES

2.1 Introduction

L'un des intérêts principaux de la théorie des graphes est l'existence d'une multitude d'algorithmes pour la résolution de problèmes divers. Ces problèmes peuvent trouver leur application dans de nombreux secteurs tels que celui des réseaux électriques, de la télécommunication, du transport, etc.

Nous nous concentrerons dans ce chapitre sur deux problèmes et sur un algorithme pour chacun.

Le premier problème sera le problème des plus courts chemins et le second sera le problème du flot maximum. Pour ces deux problèmes, des solutions instinctives, dites gloutonnes existent, et consistent à régler le problème par tâtonnement. Ces solutions peuvent toutefois être non efficaces en matière de calcul, voir non optimales. Des algorithmes adaptés ont été proposés pour répondre à ces besoins.

Pour le problème des plus courts chemins, la notion de longueur modélisée peut représenter différentes notions : distance physique, durée de temps, coût budgétaire, etc. L'algorithme étudié sera l'algorithme de Dijkstra [2].

Pour le problème du flot maximum, la question à laquelle on cherche à répondre est : quelle est la meilleure manière de transporter la plus grande quantité possible de flot entre deux sommets dans un graphe, le flot pouvant être de natures différentes : flot hydraulique, flot de données, flot de marchandises, etc. L'algorithme étudié sera l'algorithme de Ford-Fulkerson [3].

2.2 Réseau

Un **réseau non orienté** est un graphe non orienté où on associe des valeurs numériques aux arêtes et/ou aux sommets. De la même manière,un **réseau orienté** est un graphe orienté où on associe des valeurs numériques aux arcs et/ou aux sommets [4].

Les valeurs ainsi associées peuvent modéliser diverses notions selon le problème traité.

Pour ce chapitre, nous nous concentrerons sur les réseaux orientés. Le terme réseau désignera implicitement un réseau orienté.

2.3 Problème des plus courts chemins

2.3.1 Graphe valué

Un **graphe valué** $G = (X, U)$ est un réseau où on associe une valeur numérique notée par le terme **longueur**, $l(u)$, pour chaque arc $u \in U$.

Remarque La longueur d'un arc peut représenter une distance physique entre les objets modélisés par les sommets du graphe, comme elle peut représenter d'autres notions selon le problème étudié. Aussi, dans certaines situations, les longueurs des arcs peuvent être négatives.

Exemple Le graphe G_{14} donné à la figure 2.1 est un exemple de graphe valué d'ordre 4 où les valeurs numériques inscrites au niveau des arcs sont des longueurs et non pas de simples étiquettes comme au chapitre précédent.

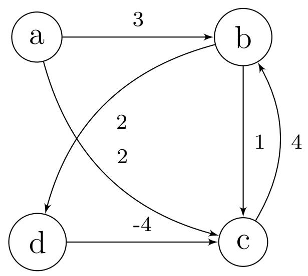


FIGURE 2.1 – Graphe valué G_{14}.

La modélisation d'un réseau routier peut se faire par un graphe valué où les sommets sont les intersections, les arcs sont les routes et les longueurs des arcs sont les longueurs physiques des routes. Une autre modélisation pourrait affecter un coût au passage par les arcs du graphe. Ces coûts pouvant être positifs comme négatifs (gain ou bien perte d'argent).

La **longueur d'un chemin** dans un graphe valué est la somme des longueurs des arcs qui le constituent.

Exemple Dans le graphe précédent G_{14}, le chemin (a,b,c) a une longueur égale à $3 + 1 = 4$ et le chemin (d,c,b,c,b) a une longueur égale à $-4 + 4 + 1 + 4 + 1 = 6$.

Remarque Pour tout sommet dans un graphe valué, on considère l'existence d'un chemin de longueur 0 reliant ce sommet à lui même.

2.3.2 Problème des plus courts chemins

Le **problème des plus courts chemins** consiste à trouver les chemins ayant les plus petites longueurs dans un graphe valué.

Il existe plusieurs variantes du problème et différents algorithmes proposés pour chaque variante :

- Recherche du plus court chemin entre deux sommets dans un graphe ;
- Recherche des plus courts chemins d'un sommet vers tous les autres sommets d'un graphe ;
- Recherche des plus courts chemins entre chaque couple de sommets dans un graphe ;
- Recherche des plus courts chemins dans un graphe avec ou sans arcs à longueurs négatives ;
- Recherche des plus courts chemins dans un graphe avec ou sans circuits à longueurs négatives ;
- etc.

Nous nous intéresserons dans ce cours à l'algorithme de Dijkstra.

2.3.3 L'algorithme de Dijkstra

L'algorithme de Dijkstra s'applique à un graphe $G = (X, U)$ valué par des valeurs positives $l(u)$ pour tout $u \in U$. Il offre en sortie les longueurs des plus courts chemins d'un sommet $a \in X$ donné, dit "source", vers tous les autres sommets du graphe.

Le principe de l'algorithme est d'affecter une valeur $dist(x)$ à chaque sommet $x \in X$. Cette valeur représentera, tout au long de l'algorithme, la longueur du plus court chemin actuel du sommet a au sommet x. $dist(x)$ est initialisée à 0 pour a et à $+\infty$ pour les autres sommets. L'objectif étant de diminuer la valeur des $dist(x)$ autant que possible au cours de l'exécution de l'algorithme.

Le pseudo-code de l'algorithme de Dijkstra est donné ci-après.

Sont utilisées pour cet algorithme les variables suivantes :

- $\overline{S}$: ensemble des sommets non marqués, i.e. sommets non encore marqués.
- $dist(x)$: distance actuelle pour tout sommet $x \in X$, i.e. longueur du plus court chemin trouvé jusqu'à présent du sommet a au sommet x.
- $pred(x)$: meilleur prédécesseur actuel pour tout sommet $x \in X$, i.e. prédécesseur du sommet x dans le plus court chemin trouvé jusqu'à présent du sommet a au sommet x.

Détails de l'algorithme ligne par ligne :

- Lignes 1 à 7 : Lignes d'initialisation :
    - Ligne 1 : $\overline{S}$ est initialisé avec l'ensemble X. Au départ, aucun sommet n'est marqué.

Algorithm 1 Algorithme de Dijkstra

Entrées :

$G = (X, U)$: graphe valué par des longueurs positives $l(u)$ pour tout $u \in U$, avec un sommet source $a \in X$.

Résultats :

$dist(x)$: la longueur du plus court chemin du sommet a vers le sommet x, pour tout $x \in X$.
$pred(x)$: le prédécesseur du sommet x dans le plus court chemin du sommet a vers le sommet x, pour tout $x \in X$.

▷ Initialisation des variables de l'algorithme

1: $\overline{S} = X$
2: $dist(a) = 0$
3: $pred(a) = a$
4: **pour** $x \in X - \{a\}$ **faire**
5:     $dist(x) = +\infty$
6:     $pred(x) = NULL$
7: **fin pour**

▷ Itérations de l'algorithme

8: **tant que** $\overline{S} \neq \emptyset$ **faire**
9:     $x \leftarrow$ sommet dans $\overline{S}$ ayant la plus petite valeur $dist$
10:     $\overline{S} \leftarrow \overline{S} - \{x\}$
11:     **pour** chaque sommet $y \in \overline{S}$ successeur de x dans G **faire**
12:         **si** $dist(y) > dist(x) + l(x, y)$ **alors**
13:             $dist(y) \leftarrow dist(x) + l(x, y)$
14:             $pred(y) \leftarrow x$
15:         **fin si**
16:     **fin pour**
17: **fin tant que**

  - Lignes 2 et 3 : $dist(a)$ et $pred(a)$ sont initialisés aux valeurs 0 et a respectivement, étant donné que le plus court chemin du sommet a au sommet a possède une longueur nulle.
  - Lignes 4 à 7 : La distance de chaque sommet $x \in X - \{a\}$ est initialisée à la plus grande valeur possible, i.e. $dist(x) \leftarrow +\infty$. Le meilleur prédécesseur de chaque sommet $x \in X - \{a\}$ est initialisé à $NULL$.
- Lignes 8 à 17 : Boucle qui va se répéter tant qu'il y aura des sommets non marqués :
  - Ligne 9 : Le sommet x non marqué et qui possède la plus petite valeur $dist()$ est sélectionné.
  - Ligne 10 : Le sommet x est marqué à présent et il est retiré de l'ensemble $\overline{S}$.
  - Lignes 11 à 16 : Boucle qui va se répéter pour chaque sommet y non marqué et successeur du sommet x dans G :
    - Ligne 12 à 15 : Si la distance actuelle du sommet y est supérieure à la distance du sommet x qu'on additionne à $l(x, y)$, alors x est un meilleur prédécesseur que le prédécesseur actuel de y. Il y a alors mise à jour de $dist(y)$ et de $pred(y)$.

L'ensemble $\overline{S}$ vide signifie que tous les sommets sont marqués et l'algorithme touche à sa fin. Les variables $dist(x)$ contiendront alors les longueurs des plus courts chemins du sommet a vers les sommets $x \in X$, et les variables $pred(x)$ contiendront les prédécesseurs des sommets $x \in X$ dans les plus courts chemins.

Remarques

- L'algorithme de Dijkstra s'exécute en N itérations, N étant l'ordre du graphe G.
- L'algorithme de Dijkstra ne retourne pas comme résultats les plus courts chemins vers chaque sommet. Ces derniers peuvent toutefois être construits en se basant sur le contenu des variables $pred(x)$.
- Un sommet $x \in X$ associé à une valeur $dist(x) = +\infty$ à la fin de l'algorithme est un sommet pour lequel il n'existe aucun chemin à partir du sommet a. La valeur $pred(x)$ sera dans ce cas toujours égale à $NULL$ à la fin de l'algorithme.

Exemple Soit le graphe valué G_{15} donné à la figure 2.2.

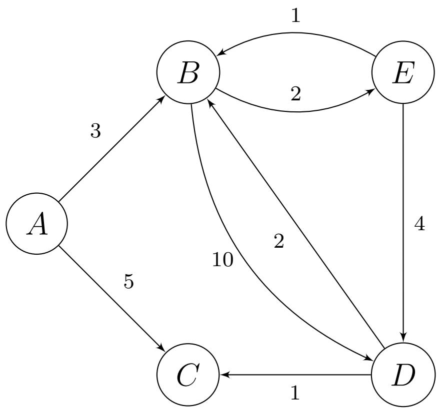


FIGURE 2.2 – Graphe valué G_{15}.

La trace d'exécution de l'algorithme de Dijkstra sur G_{15} pour trouver les longueurs des plus courts chemins du sommet A vers tous les autres sommets du graphe est comme suit :

x	$dist(x)$	$\overline{S}$	$dist_{pred}$				
			A	B	C	D	E
-	-	$\{A, B, C, D, E\}$	0_A	$+\infty_{NULL}$	$+\infty_{NULL}$	$+\infty_{NULL}$	$+\infty_{NULL}$
A	0	$\{B, C, D, E\}$	0_A	3_A	5_A	$+\infty_{NULL}$	$+\infty_{NULL}$
B	3	$\{C, D, E\}$	0_A	3_A	5_A	13_B	5_B
C	5	$\{D, E\}$	0_A	3_A	5_A	13_B	5_B
E	5	$\{D\}$	0_A	3_A	5_A	9_E	5_B
D	9	$\{\}$	0_A	3_A	5_A	9_E	5_B

Ainsi, l'exécution de l'algorithme produit en résultats les longueurs des plus courts chemins du sommet A vers tous les sommets du graphe. Ces longueurs apparaissent à la dernière ligne du tableau. À titre d'exemple, la longueur du plus court chemin de sommet A vers le sommet C est égale à 5.

L'algorithme produit également le prédécesseur de chaque sommet dans le plus court chemin du sommet A vers ce sommet. Par exemple, le prédécesseur du sommet E dans le plus court chemin à partir du sommet A est le sommet B.

Les plus courts chemins ne sont quant à eux pas directement produits par l'algorithme comme résultat. Ils peuvent toutefois être déduits en se basant sur les prédécesseurs des sommets dans les plus courts chemins. À titre d'exemple, le meilleur prédécesseur du sommet E est le sommet B, celui du sommet B est le sommet A. Et donc le plus court chemin du sommet A vers le sommet E est le chemin (A, B, E).

Nous avons donc :

- Le plus court chemin du sommet A vers le sommet A est le chemin (A) et sa longueur est 0.
- Le plus court chemin du sommet A vers le sommet B est le chemin (A, B) et sa longueur est 3.
- Le plus court chemin du sommet A vers le sommet C est le chemin (A, C) et sa longueur est 5.
- Le plus court chemin du sommet A vers le sommet D est le chemin (A, B, E, D) et sa longueur est 9.
- Le plus court chemin du sommet A vers le sommet E est le chemin (A, B, E) et sa longueur est 5.

2.4 Le problème de flot maximum

2.4.1 Réseau de flot

Un **réseau de flot** $G = (X, U)$ est un réseau avec deux sommets particuliers :

- $s \in X$: appelé **source** et qui est sans arc entrant ;
- $t \in X$: appelé **puits** et qui est sans arc sortant ;

et où on associe deux valeurs positives $c(u)$ et $f(u)$ appelées respectivement **capacité** et **quantité de flot** à chaque arc $u \in U$.

Les quantités de flot $f(u)$ doivent respecter les contraintes suivantes :

- *Contrainte de capacité :* pour chaque arc $u \in U$, la quantité de flot ne doit pas dépasser sa capacité ($f(u) \leq c(u)$).
- *Contrainte de conservation de flot :* pour chaque sommet $x \in X - \{s, t\}$, la somme des quantités entrantes vers ce sommet doit être égale à la somme des quantités sortantes de ce sommet ($\sum_{(y,x) \in U} f(y, x) = \sum_{(x,y) \in U} f(x, y)$).

Exemples La figure 2.3 représente un exemple de réseau de flot, G_{16}, avec 4 sommets dont le sommet source s et le sommet puits t, et où chaque arc est associé à deux valeurs : une capacité et une quantité de flot, cette dernière étant précisée entre parenthèses. À titre d'exemple, la capacité de l'arc (A, t) est égale à 4, i.e. $c(A, t) = 4$, et sa quantité de flot est égale à 1, i.e. $f(A, t) = 1$.

Nous avons donc une quantité de flot égale à 4 qui sort de s, arrivée à A, elle se divise en deux, avec 3 unités qui vont vers B puis vers t et 1 unité qui va directement vers t.

Ces quantités respectent bien évidemment les deux contraintes sur les quantités de flots. Pour chaque arc, la quantité de flot ne dépasse pas sa capacité. Et pour chacun des sommets A et B, la somme des quantités entrantes et égale à la somme des quantités sortantes.

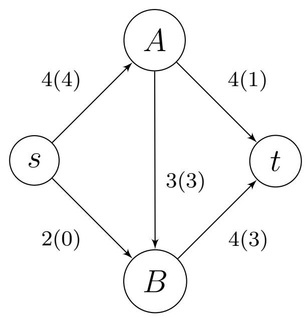


FIGURE 2.3 – Réseau de flot G_{16}.

Un réseau de flot peut modéliser différents problèmes, et la notion de capacité peut modéliser différents concepts dépendamment du problème étudié. À titre d'exemple, la capacité dans un réseau hydraulique peut représenter le débit maximum d'une conduite d'eau (d'un tuyau), c.à.d. la quantité maximum d'eau pouvant être transportée par la conduite par unité de temps. Les quantités de flot pourraient être les débits d'eau réels dans les conduites. Ces débits réels ne doivent bien sûr pas dépasser le débit maximum des conduites, et tout ce qui arrivera à un branchement devra en sortir.

Un arc dans un réseau est dit **saturé** si $f(u) = c(u)$.

Exemple Dans l'exemple précédent, les arcs (s, B) et (A, B) sont des arcs saturés. Leurs quantités de flot sont égales à leurs capacités respectives, ce qui signifie que ces arcs ne pourront plus transporter de flot supplémentaire.

Le **flot d'un réseau** est l'ensemble de toutes les quantités de flot associées aux arcs du réseau.

Exemple Dans l'exemple précédent, l'ensemble des quantités de flot associées aux 5 arcs du réseau représente un flot.

La **valeur de flot** d'un réseau est égale à la somme des quantités entrantes vers son puits t (qui est elle même égale à la somme des quantités sortantes de sa source s).

Exemple La valeur de flot de l'exemple précédent est égale à $f(A, t) + f(B, t) = f(s, A) + f(s, B) = 4$.

Le **flot maximum** d'un réseau est le flot avec la plus grande valeur de flot possible pour ce réseau.

Exemple Dans la figure 2.4, nous avons un flot maximum qui parcourt le réseau. En effet, plus aucune quantité supplémentaire ne peut être injectée au réseau vu la saturation des deux arcs (s, A) et (s, B). La valeur du flot maximum pour ce réseau est donc égale à 6.

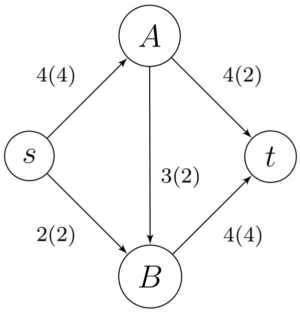


FIGURE 2.4 – Flot maximum.

2.4.2 Problème du flot maximum

Le **problème du flot maximum** consiste à trouver le flot maximum dans un réseau.

Considérons tout d'abord une première méthode donnée par l'algorithme 2. Cette méthode consiste à initialiser les quantités de flots de tous les arcs du réseau à 0, puis, à essayer d'augmenter la valeur du flot en recherchant au fur et à mesure des chemins de s vers t pour lesquels le flot n'a encore saturé aucun arc. Nous verrons par la suite que le flot ainsi obtenu n'est pas nécessairement maximum.

Algorithm 2 Algorithme glouton

Entrées :
$G = (X, U)$: réseau de flot à traiter, avec une source s, un puits t, et des capacité $c(u)$ pour tout $u \in U$.

Résultats :
f : un flot dans le réseau G.

▷ Initialisation des variables de l'algorithme

$term \leftarrow faux$
pour $u \in U$ **faire**
  $f(u) = 0$
fin pour

▷ Itérations de l'algorithme

répéter
  chercher dans G un chemin P de s vers t tel que $f(u) < c(u)$ pour chaque $u \in P$
  si un tel chemin n'existe pas **alors**
    $term \leftarrow vrai$
  sinon
    $\Delta \leftarrow \min_{u \in P}(c(u) - f(u))$
    pour $u \in P$ **faire**
      $f(u) \leftarrow f(u) + \Delta$
    fin pour
  fin si
jusqu'à ce que $term = vrai$

Sont utilisées pour cet algorithme les variables suivantes :

— $term$: variable booléenne initialisée à $faux$ et qui sera mise à $vrai$ une fois que tous les chemins dans le réseau contiendront au moins un arc saturé.

— $f(u)$: quantité de flot de l'arc $u \in U$.

— Δ : variable qui prend la plus petite valeur de capacité restante sur un chemin de s vers t, en sachant que la capacité restante sur un arc u correspond à la valeur $c(u) - f(u)$.

Détails de l'algorithme ligne par ligne :

— Lignes 1 à 4 : Lignes d'initialisation :

  — Ligne 1 : La variable booléenne $term$ est initialisée à $faux$.

  — Lignes 2 à 4 : Les quantités de flot sur tous les arcs sont initialisées à 0.

— Lignes 5 à 15 : Boucle qui va se répéter tant qu'il y aura des chemins sans arcs saturés dans le réseau :

  — Ligne 6 : Chercher dans le réseau G un chemin P de s vers t qui contient au moins un arc non saturé.

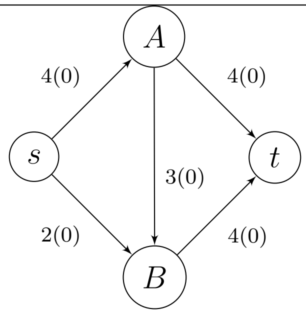
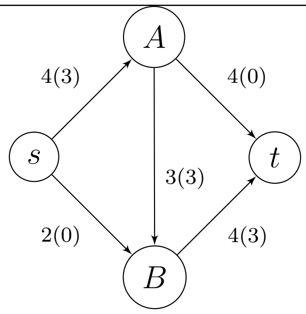
  — Lignes 7 et 8 : Si un tel chemin n'existe pas, l'algorithme touche à sa fin.

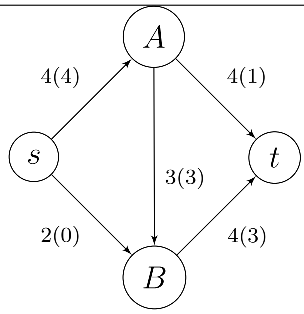
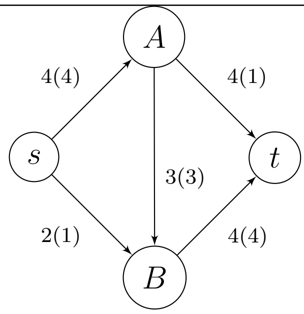
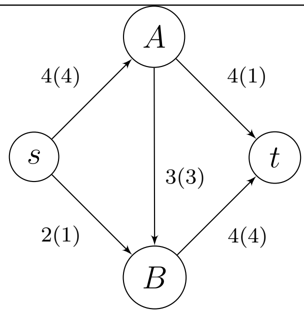
  — Ligne 9 à 14 : Dans le cas contraire, une certaine quantité de flot sera ajoutée sur ce chemin de manière à respecter la contrainte de capacité de tous ses arcs :

    — Ligne 10 : Δ reçoit la capacité restante minimale sur tous les arcs du chemin P.

    — Lignes 11 à 13 : Une quantité de flot supplémentaire égale à Δ est ajoutée à chaque arc du chemin P.

Exemple Exécutons cet algorithme sur notre précédent réseau.

$term$	P	Δ	flot
$faux$	-	-	
$faux$	(s, A, B, t)	$\min(4, 3, 4) = 3$	

$faux$	(s, A, t)	$\min(1, 4) = 1$	
$faux$	(s, B, t)	$\min(2, 1) = 1$	
$vrai$	-	-	

La valeur du flot ainsi trouvé est égale à 5, or nous avons déjà trouvé un flot avec une valeur supérieure. Ceci signifie que le flot trouvé n'est pas maximum.

Le choix des chemins lors de l'exécution de l'algorithme a produit un résultat non optimal car une fois une quantité de flot fixée sur un arc, il n'y a aucun moyen de faire de retour arrière pour cette quantité afin de la réorienter autrement.

En effet, en permettant la réorientation de flot nous pourrions :

— Ajouter 1 unité à $f(s, B)$

— Diminuer 1 unité à la $f(A, B)$ et ajouter cette unité à $f(A, t)$, ce qui correspond à une réorientation

— Laisser $f(B, t)$ à 4, ce qui correspond à la somme des 2 unités arrivant de s et des 2 unités arrivant de A.

La valeur du flot ainsi obtenu serait égale à 6. Et le flot est le même flot que celui donné à la figure 2.4.

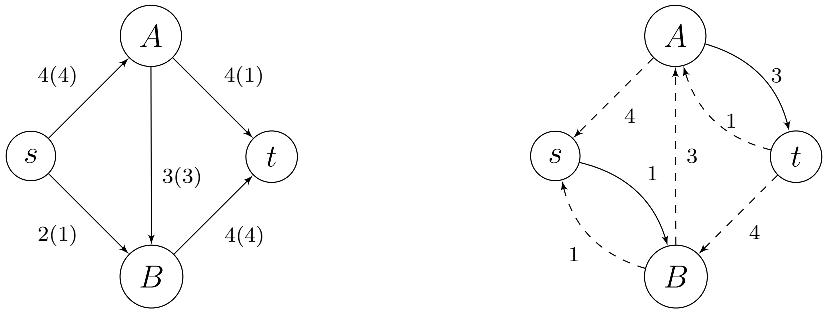
Ce concept de réorientation de flot est possible en introduisant la notion de réseau résiduel à l'algorithme de recherche du flot maximum.

Le **réseau résiduel** $G_f = (X, U_f)$ d'un réseau $G = (X, U)$ indique ce qu'il est encore possible d'ajouter comme flot à ce réseau. G_f comporte les mêmes sommets que G, et pour chaque arc $u = (x, y) \in U$ il existe :

— Un arc (x, y) dans U_f à **capacité résiduelle** égale à $c(u) - f(u)$ ssi $c(u) - f(u) > 0$. Cet arc sera dit **arc direct**.

— un arc (y, x) dans U_f à capacité résiduelle égale à $f(u)$ ssi $f(u) > 0$. Cet arc sera dit **arc inverse**.

Exemple Reprenons le réseau précédent dans son état final après avoir appliqué l'algorithme glouton. Le réseau et son réseau résiduel sont donnés à la figure 2.5. Dans le graphe résiduel, les arcs directs sont représentés par des lignes continues alors que les arcs inverses sont représentés par des lignes pointillées. Ainsi, la capacité résiduelle de l'arc direct (A, t), par exemple, est égale à 3 et la capacité résiduelle de l'arc inverse (t, A) est égale à 1.



(a) Réseau de flot. (b) Réseau résiduel.

FIGURE 2.5 – Réseau de flot et son réseau résiduel.

Le fait qu'il existe dans le réseau résiduel au moins un chemin de s vers t, (s, B, A, t), signifie qu'il est possible de trouver un flot avec une valeur supérieure à celui donné à la figure 2.5a.

Un **chemin augmentant** est un chemin élémentaire de s vers t dans le réseau résiduel [4].

Au lieu de chercher des chemins sans arcs saturés dans G, l'algorithme de Ford-Fulkerson donné à l'algorithme 3 consiste à chercher des chemins augmentants dans son réseau résiduel. Si un tel chemin existe, il s'agira alors pour chaque arc (x, y) du chemin augmentant :

— d'ajouter du flot à l'arc (x, y) dans G si (x, y) est un arc direct,

— de diminuer du flot à l'arc (y, x) dans G si (x, y) est un arc inverse, ce qui correspond à une réorientation de flot.

L'absence d'un tel chemin indique l'optimalité du flot trouvé jusqu'à présent. La preuve de cette optimalité ne sera pas traitée dans ce cours.

Sont utilisées pour cet algorithme les mêmes variables que pour l'algorithme glouton précédent :

— *term* : variable booléenne initialisée à $faux$ et qui sera mise à $vrai$ une fois qu'il n'y aura plus de chemins augmentants dans le réseau résiduel.

— $f(u)$: quantité de flot de l'arc $u \in U$ dans le réseau G.

Algorithm 3 Algorithme de Ford-Fulkerson

Entrées :

$G = (X, U)$: réseau de flot à traiter, avec une source s, un puits t, et des capacités $c(u)$ pour $u \in U$.

Résultats :

f : un flot maximum dans le réseau G.

▷ Initialisation des variables de l'algorithme

1: $term \leftarrow faux$
2: **pour** $u \in U$ **faire**
3: $\quad f(u) = 0$
4: **fin pour**
5: Construire G_f, le réseau résiduel de G avec le flot f

▷ Itérations de l'algorithme

6: **répéter**
7: $\quad$ chercher dans G_f un chemin augmentant P de s vers t
8: $\quad$ **si** un tel chemin n'existe pas **alors**
9: $\quad\quad term \leftarrow vrai$
10: $\quad$ **sinon**
11: $\quad\quad \Delta \leftarrow \min_{u \in P}$(capacité résiduelle de u dans G_f)
12: $\quad\quad$ **pour** $u = (x, y) \in P$ **faire**
13: $\quad\quad\quad$ **si** u est un arc direct dans G **alors**
14: $\quad\quad\quad\quad f(x, y) \leftarrow f(x, y) + \Delta$
15: $\quad\quad\quad$ **sinon**
16: $\quad\quad\quad\quad f(y, x) \leftarrow f(y, x) - \Delta$
17: $\quad\quad\quad$ **fin si**
18: $\quad\quad$ **fin pour**
19: $\quad\quad$ Re-construire G_f, le réseau résiduel de G avec le nouveau flot f
20: $\quad$ **fin si**
21: **jusqu'à ce que** $term = vrai$

— Δ : variable qui prend la plus petite valeur de capacité résiduelle sur un chemin augmentant.

Détails de l'algorithme ligne par ligne :

— Lignes 1 à 5 : Lignes d'initialisation :

  — Ligne 1 : La variable booléenne *term* est initialisée à $faux$.

  — Lignes 2 à 4 : Les quantités de flot sur tous les arcs de G sont initialisées à 0.

  — Ligne 5 : Le réseau résiduel G_f est construit.

— Lignes 6 à 21 : Boucle qui va se répéter tant qu'il y aura des chemins augmentants :

  — Ligne 7 : Chercher dans le réseau résiduel G_f un chemin augmentant P de s vers t.

  — Lignes 8 et 9 : Si un tel chemin n'existe pas, l'algorithme touche à sa fin.

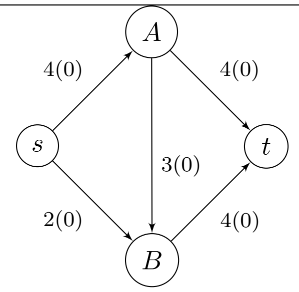
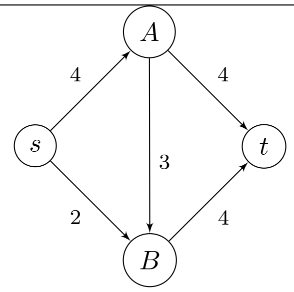
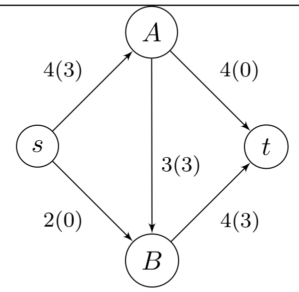
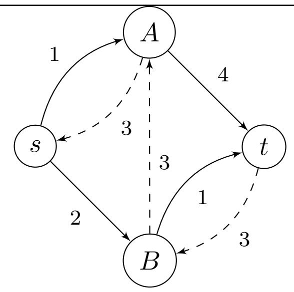
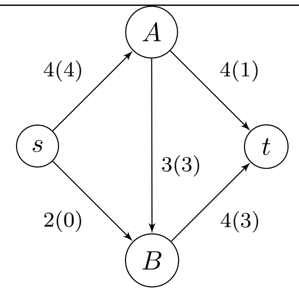
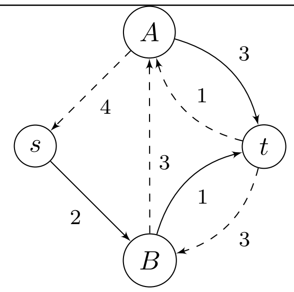
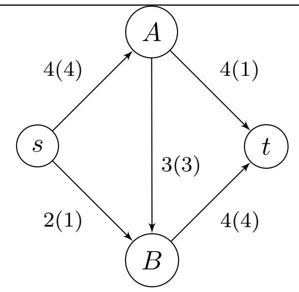
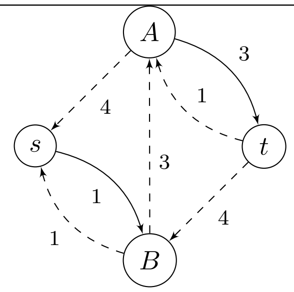
  — Ligne 11 à 19 : Dans le cas contraire, une certaine quantité de flot sera ajoutée au réseau de manière à respecter la contrainte de capacité de tous ses arcs :

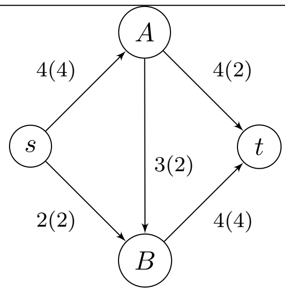
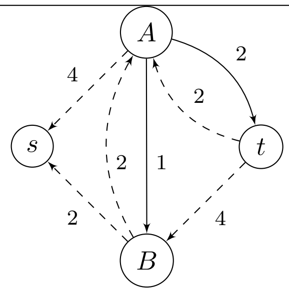
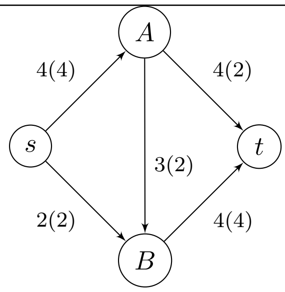
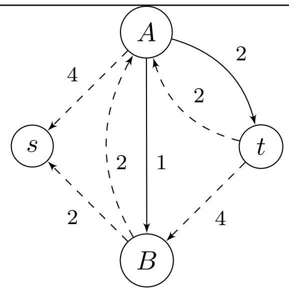
    — Ligne 11 : Δ reçoit la capacité résiduelle minimale sur tous les arcs du chemin P.

    — Lignes 12 à 18 : Pour chaque arc $u = (x, y)$ dans P :

  - — Lignes 13 à 14 : Une quantité de flot supplémentaire égale à Δ est ajoutée à chaque $f(x, y)$ si u est un arc direct.
  - — Lignes 15 à 16 : Une quantité de flot égale à Δ est soustraite de chaque $f(y, x)$ si u est un arc inverse.
- — Le réseau résiduel est reconstruit avec le nouveau flot obtenu.

Exemple Exécutons l'algorithme de Ford-Fulkerson sur notre réseau.

term	P	Δ	flot	G_f
$faux$	-	-		
$faux$	(s, A, B, t)	$\min(4, 3, 4) = 3$		
$faux$	(s, A, t)	$\min(1, 4) = 1$		
$faux$	(s, B, t)	$\min(2, 1) = 1$		

$faux$	(s, B, A, t)	$\min(1, 3, 3) = 1$		
$vrai$	-	-		

Le flot ainsi obtenu à la dernière ligne du tableau est à présent maximum et la solution est donc optimale.

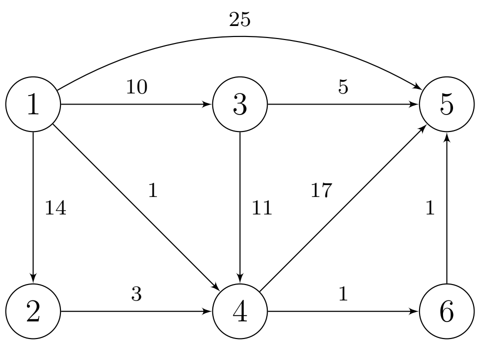
2.5 Exercices

Exercice 21

1. Un graphe valué peut-il comporter des longueurs négatives sur les arcs.
2. Que contient la variable $dist(x)$ de l'algorithme de Dijkstra comme information pour tout sommet $x \in X$ dans un graphe $G = (X, U)$, tout au long de l'exécution de l'algorithme ?
3. Un réseau de flot possède une source et un puits. Quelles sont leurs particularités ?

Exercice 22

Soit le graphe orienté valué G tracé ci-dessous :



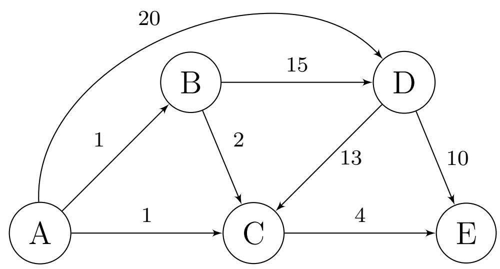
1. Quel est l'algorithme à appliquer pour obtenir les longueurs des plus courts chemins du sommet 1 vers tous les autres sommets du graphe G ?

2. Donner la trace d'exécution de cet algorithme sur ce graphe.

3. Énumérer ces plus courts chemins.

Exercice 23

Soit le graphe valué $G = (X, U)$ tracé ci-dessous :



1. Appliquer l'algorithme de Dijkstra pour trouver les longueurs des plus courts chemins allant du sommet A vers tous les autres sommets de G.

2. Énumérer ces plus courts chemins.

Exercice 24

Une compagnie aérienne possède un plan de vol qui relie les villes A, B, C, D et E selon le tableau suivant. Les valeurs données correspondent à la durée des vols disponibles :

	A	B	C	D	E
A		1h30	2h		2h15
B	1h40				3h
C	2h20			2h55	
D			3h20		1h05
E	2h25	3h10		1h10	

1. Quels sont les trajets les plus rapides pour aller de la ville B à chacune des autres villes proposées ?

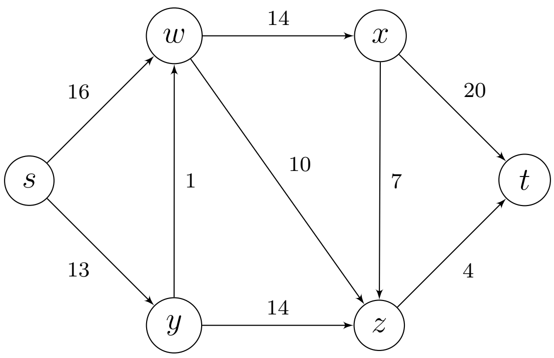
2. Comment modifier l'algorithme utilisé pour répondre à la question 1, afin de pouvoir prendre en considération les temps d'escale entre les vols ?

Exercice 25

Un logiciel de traitement de texte LaTeXutilise une procédure d'optimisation pour décomposer les paragraphes en lignes de manière à obtenir un résultat élégant. Supposons qu'un paragraphe contienne n mots, et que chaque mot soit numéroté avec une valeur i allant de 1 à n selon son emplacement dans le paragraphe. Soit c_{ij} une valeur calculée par le logiciel et qui qui dénote l'élégance d'une ligne si celle-ci commence par le mot i et se termine par le mot $j-1$. En considérant que les valeurs c_{ij} sont connues, formuler le problème de décomposition des mots du paragraphe en lignes de manière à maximiser l'élégance totale du paragraphe par un problème du plus court chemin.

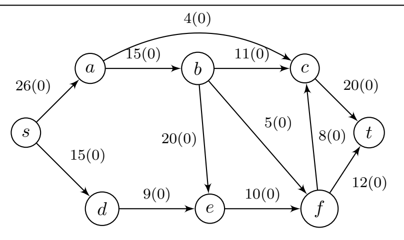
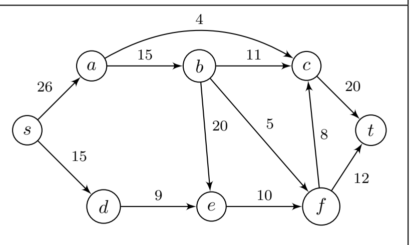
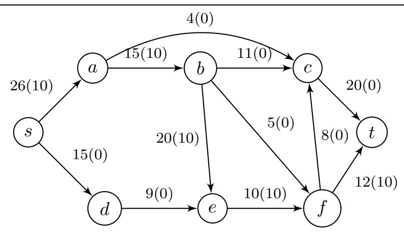
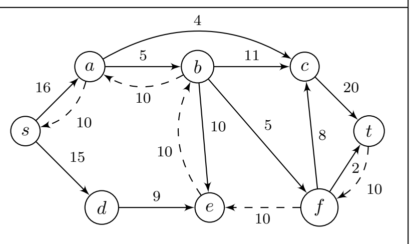
Exercice 26

Trouver le flot maximum du réseau de flot ci-dessous à l'aide de l'algorithme de Ford-Fulkerson.



Exercice 27

Compléter la trace d'exécution de l'algorithme de Ford-Fulkerson suivante :

$term$	P	Δ	flot	G_f
$faux$	-	-		
$faux$	$(s,a,b,e,$ $f,t)$	min(26,15,20, 10,12)=10		

Exercice 28

Chacun des 5 chercheurs d'un laboratoire doit utiliser un ordinateur pour effectuer des calculs pendant une ou plusieurs périodes d'une heure. Chaque chercheur i a donné l'ensemble h_i des créneaux horaires durant lesquelles il est disponible pour effectuer ses calculs, ainsi que le nombre r_i de créneaux dont il a besoin :

$$\begin{array}{ll} h_1 = \{1, 2, 4, 5\} & r_1 = 3 \\ h_2 = \{2, 3, 5, 6\} & r_2 = 2 \\ h_3 = \{1, 3, 4\} & r_3 = 2 \\ h_4 = \{2, 3, 4, 5\} & r_4 = 3 \\ h_5 = \{4, 5, 6\} & r_5 = 1 \end{array}$$

On cherche à affecter les créneaux horaires aux chercheurs de manière à satisfaire tout le monde, sachant que l'ordinateur ne peux accueillir plus de deux personnes à la fois.

1. Modéliser ce problème en terme de graphe.

2. Résoudre le problème en utilisant l'algorithme adapté.

2.6 Solutions des exercices

Solution de l'exercice 21

1. Oui.

2. La variable $dist(x)$ contient à tout moment durant l'exécution de l'algorithme de Dijkstra la longueur du plus court chemin trouvé jusqu'à ce moment de la source a vers le sommet x.

3. Soit s la source d'un réseau de flot et p son puits. Nous avons $d^-(s) = 0$ et $d^+(p) = 0$.

Solution de l'exercice 22

1. L'algorithme permettant d'obtenir les plus courts chemins du sommet 1 vers tous les autres sommets du graphe G est l'algorithme de Dijkstra.

2. Trace d'exécution de l'algorithme de Dijkstra sur G avec le sommet 1 comme sommet source :

x	$dist(x)$	$\overline{S}$	$dist_{pred}$					
			1	2	3	4	5	6
-	-	{1,2,3,4,5,6}	0_1	∞_{NULL}	∞_{NULL}	∞_{NULL}	∞_{NULL}	∞_{NULL}
1	0	{2,3,4,5,6}	0_1	14_1	10_1	1_1	25_1	∞_{NULL}
4	1	{2,3,5,6}	0_1	14_1	10_1	1_1	18_4	2_4
6	2	{2,3,5}	0_1	14_1	10_1	1_1	3_6	2_4
5	3	{2,3}	0_1	14_1	10_1	1_1	3_6	2_4
3	10	{2}	0_1	14_1	10_1	1_1	3_6	2_4
2	14	{}	0_1	14_1	10_1	1_1	3_6	2_4

3. Les plus courts chemins sont :
   - — vers le sommet 2 : (1, 2) ;
   - — vers le sommet 3 : (1, 3) ;
   - — vers le sommet 4 : (1, 4) ;
   - — vers le sommet 5 : (1, 4, 6, 5) ;
   - — vers le sommet 6 : (1, 4, 6).

Solution de l'exercice 23

1. Trace d'exécution de l'algorithme de Dijkstra sur G avec A comme sommet source :

x	$dist(x)$	$\overline{S}$	$dist_{pred}$				
			A	B	C	D	E
-	-	$\{A, B, C, D, E\}$	0_A	∞_{NULL}	∞_{NULL}	∞_{NULL}	∞_{NULL}
A	0	{B,C,D,E}	0_A	1_A	1_A	20_A	∞_{NULL}
B	1	{C,D,E}	0_A	1_A	1_A	16_B	∞_{NULL}
C	1	{D,E}	0_A	1_A	1_A	16_B	5_C
E	5	{D}	0_A	1_A	1_A	16_B	5_C
D	16	{}	0_A	1_A	1_A	16_B	5_C

   Les longueurs des plus courts chemins du sommet A vers les sommets B, C, D et E sont 1, 1, 16 et 5 respectivement.

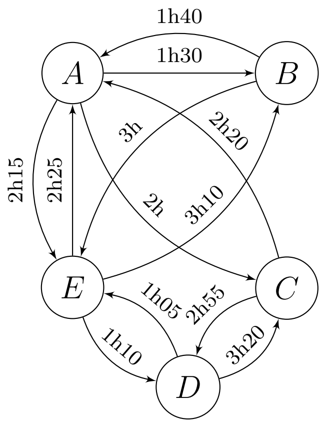
2. Les plus courts chemins sont :
   - — vers le sommet B : (A, B).
   - — vers le sommet C : (A, C).
   - — vers le sommet D : (A, B, D).
   - — vers le sommet E : (A, C, E).

Solution de l'exercice 24

1. Le problème peut être modélisé par un graphe orienté valué $G = (X, U)$, où :

— X : ensemble des villes $\{A, B, C, D, E\}$;

— U : ensemble des vols. Les valeurs associées aux arcs sont les durées des vols.



Trace de l'exécution de l'algorithme de Dijkstra :

x	$dist(x)$	$\overline{S}$	$dist_{pred}$				
			A	B	C	D	E
-	-	$\{A, B, C, D, E\}$	$+\infty_{NULL}$	0_B	$+\infty_{NULL}$	$+\infty_{NULL}$	$+\infty_{NULL}$
B	0	$\{A, C, D, E\}$	$1h40_B$	0_B	$+\infty_{NULL}$	$+\infty_{NULL}$	$3h_B$
A	1h40	$\{C, D, E\}$	$1h40_B$	0_B	$3h40_A$	$+\infty_{NULL}$	$3h_B$
E	3h	$\{C, D\}$	$1h40_B$	0_B	$3h40_A$	$4h10_E$	$3h_B$
C	3h40	$\{D\}$	$1h40_B$	0_B	$3h40_A$	$4h10_E$	$3h_B$
D	4h10	$\{D\}$	$1h40_B$	0_B	$3h40_A$	$4h10_E$	$3h_B$

Les trajets les plus rapides à partir de la ville B sont :

— vers A : (B, A) et sa durée est de 1h40

— vers C : (B, A, C) et sa durée est de 3h40

— vers D : (B, E, D) et sa durée est de 4h10

— vers E : (B, E) et sa durée est de 3h

2. Considérons que la durée d'une escale entre un vol de i vers j et un autre vol de j vers k est noté par ESC_{ijk}. Afin de prendre en considération le temps d'escale dans la durée totale du trajet, on pourrait modifier l'algorithme de Dijkstra comme suit :

   Remplacer les lignes 12 et 13 par :

   si $(dist(y) > dist(x) + l(x, y) + ESC_{pred(x)xy})$ **alors**

   $dist(y) \leftarrow dist(x) + l(x, y) + ESC_{pred(x)xy}$

   fin si

Solution de l'exercice 25

Le problème peut être modélisé par un graphe orienté valué $G = (X, U)$ avec :

— X : ensemble des sommets numérotés de 1 à $n+1$,

— U : ensemble des arcs tels qu'un arc $(i, j) \in U$ ssi $i < j$. Chaque arc est associé à la valeur $-c_{ij}$.

La résolution du problème revient à trouver le plus court chemin du sommet 1 au sommet $n+1$, ainsi les arcs retenus dans le plus court chemin nous permettront d'obtenir les meilleures lignes pour avoir le paragraphe le plus élégant possible.

Exemple : Si on a 53 mots et qu'on obtient (1,14, 28, 49, 54) comme plus court chemin :

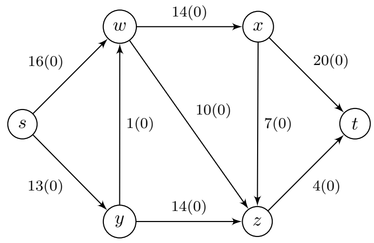
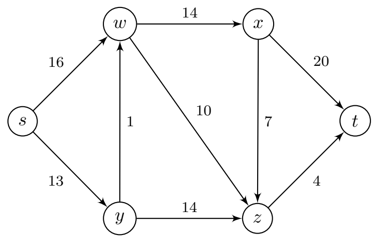
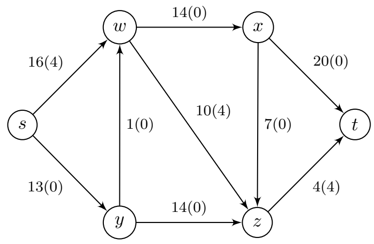
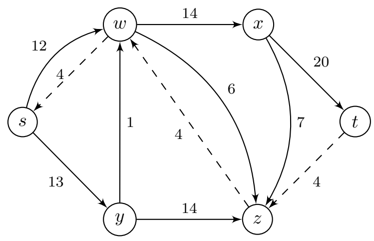
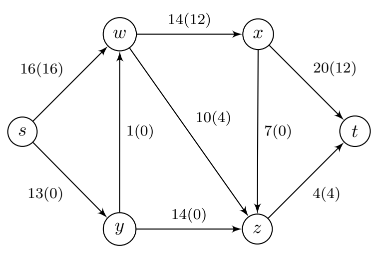
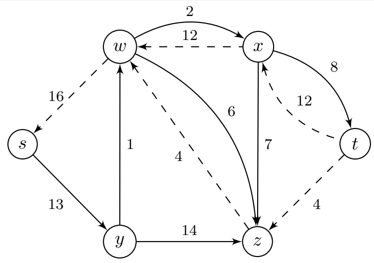
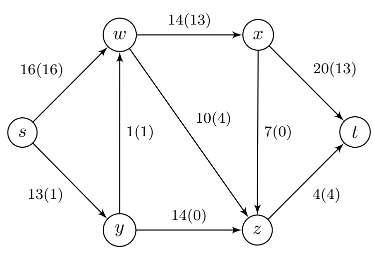
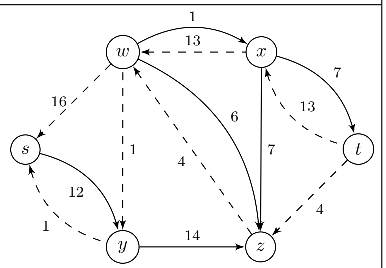
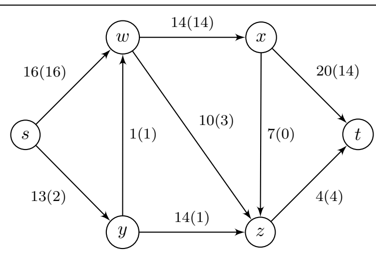
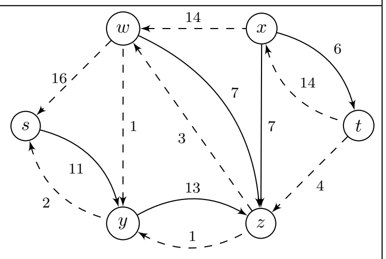
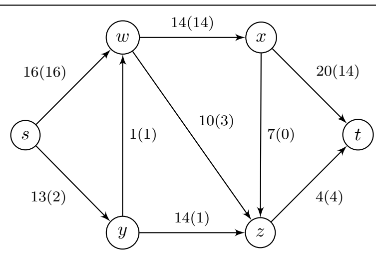
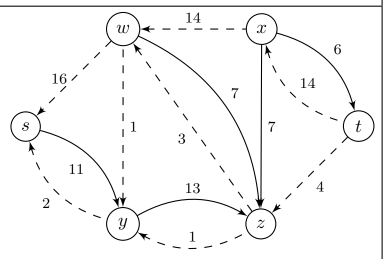
— La première ligne contiendra les mots 1 à 13 ;

— La deuxième ligne contiendra les mots 14 à 27 ;

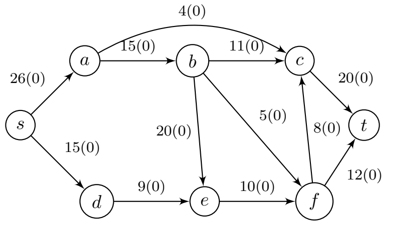
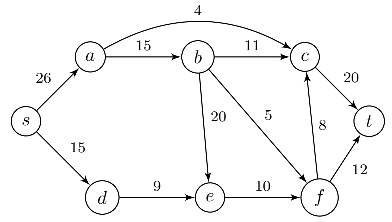
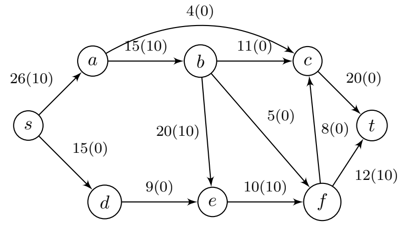
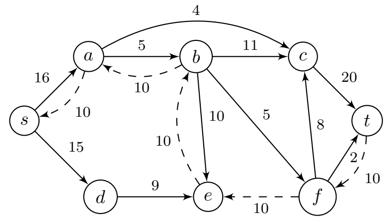
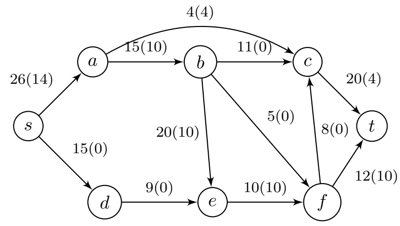
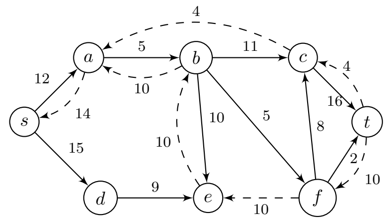
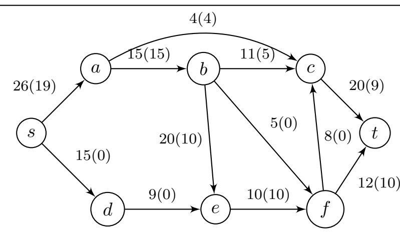
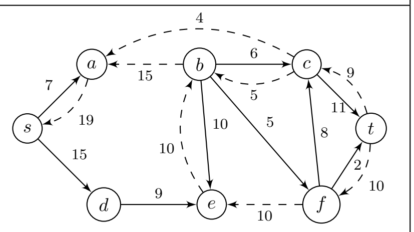
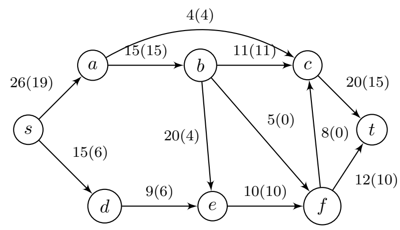
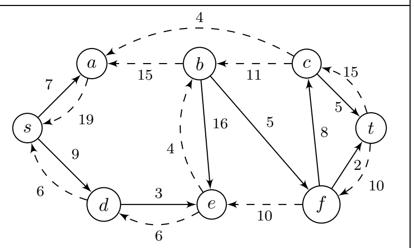
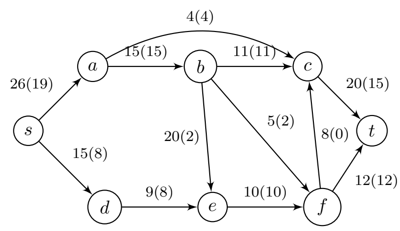
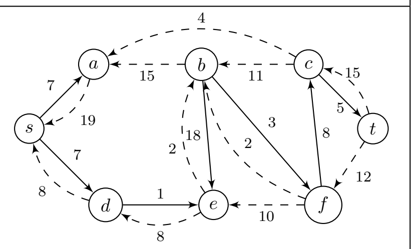
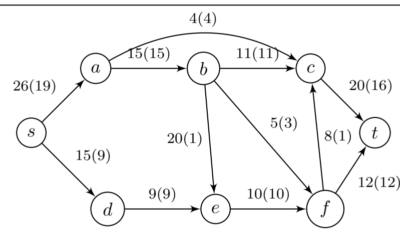
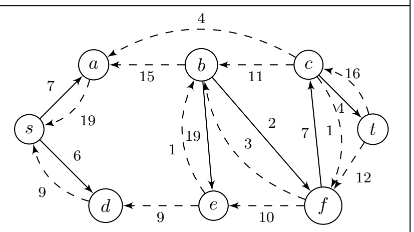
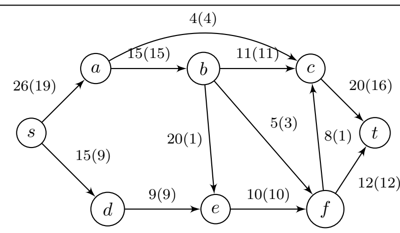
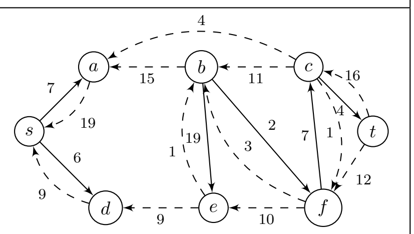
— La troisième ligne contiendra les mots 28 à 48 ;

— La quatrième ligne contiendra les mots 49 à 53 ;

Solution de l'exercice 26

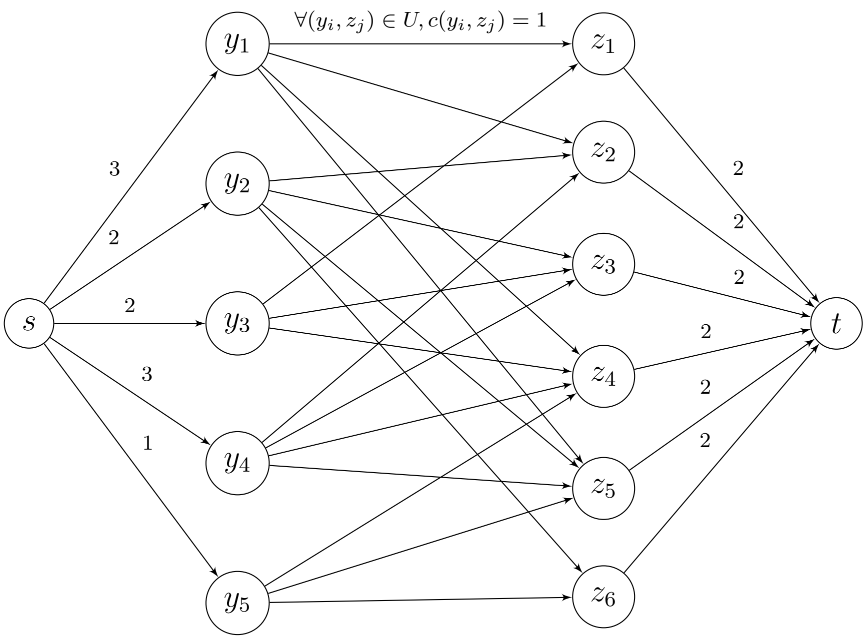
$term$	P	Δ	flot	G_f
$faux$	-	-		
$faux$	(s,w,z,t)	$\min(16,10,4) = 4$		
$faux$	(s,w,x,t)	$\min(12,14,20) = 12$		
$faux$	(s,y,w,x,t)	$\min(13,1,2,8) = 1$		
$faux$	(s,y,z,w,x,t)	$\min(12,14,4,1,7) = 1$		
$vrai$	-	-		

Solution de l'exercice 27

$term$	P	Δ	flot	G_f
$faux$	-	-		
$faux$	(s, a, b, e, f, t)	min(26,15,20, 10,12)=10		
$faux$	(s, a, c, t)	min(16,4,20)=4		
$faux$	(s, a, b, c, t)	min(12,5,11,16) =5		
$faux$	(s, d, e, b, c, t)	min(15,9,10,6, 11)=6		
$faux$	(s, d, e, b, f, t)	min(9,3,4,5,2) =2		
$faux$	(s, d, e, b, f, c, t)	min(7,1,2,3,8,5) =1		
$vrai$	-	-		

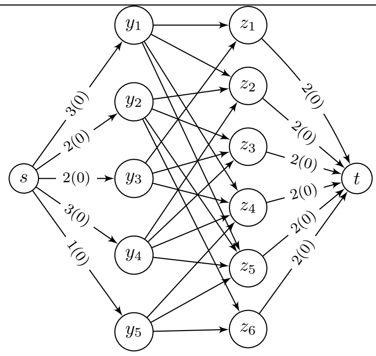
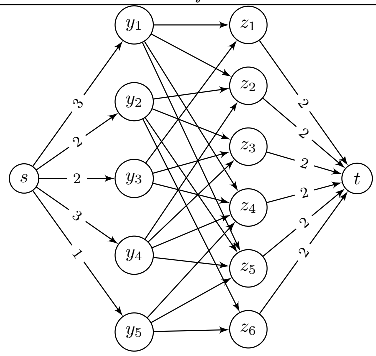
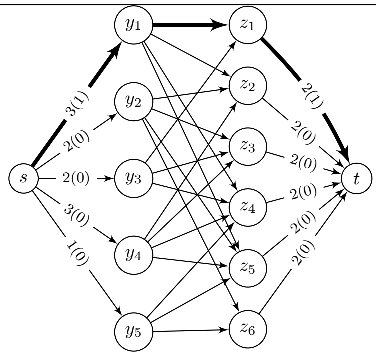
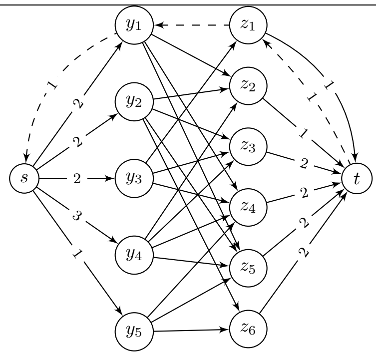
Solution de l'exercice 28

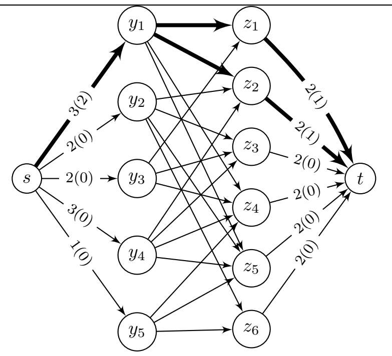
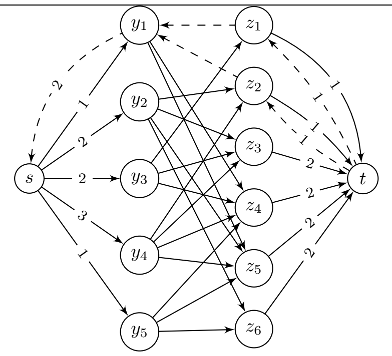
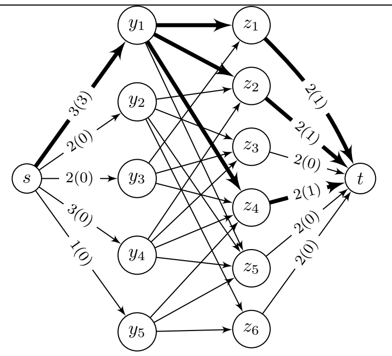
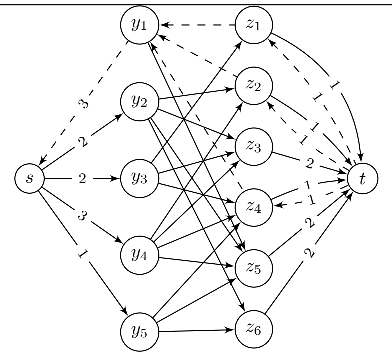
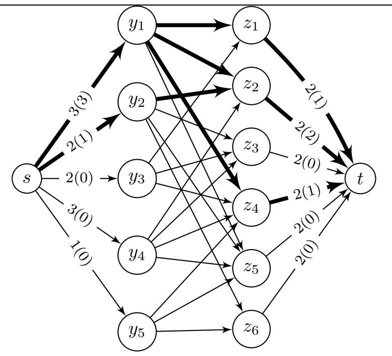
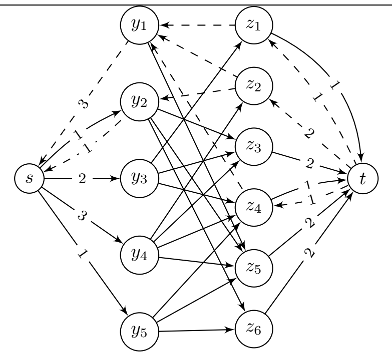
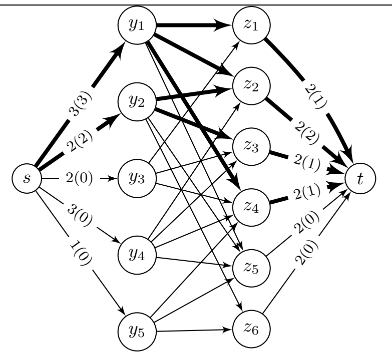
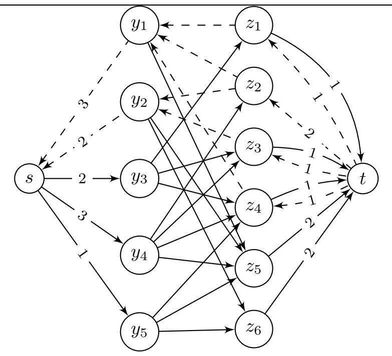
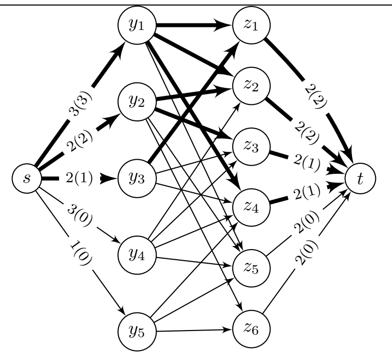
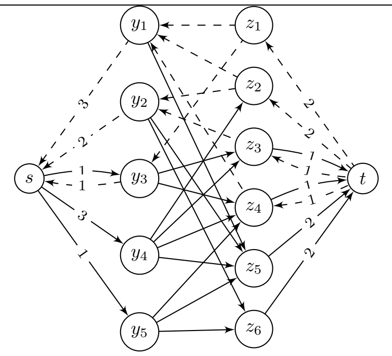
1. Le problème consiste à trouver un flot maximal dans un réseau de flot $G = (X, U)$ avec :
   - L'ensemble des sommets $X = \{s, t\} \cup Y \cup Z$ construit comme suit :
     - s : sommet source ;
     - t : sommet puits ;
     - Y : ensemble des 5 chercheurs $\{y_1, \dots, y_5\}$;
     - Z : ensemble des 6 créneaux horaires $\{z_1, \dots, z_6\}$;
   - L'ensemble des arcs U construit comme suit :
     - (s, y_i) pour tout $i \in \{1, \dots, 5\}$ avec une capacité égale à r_i (nombre de créneaux horaires dont a besoin le chercheur i) ;
     - (y_i, z_j) pour tout $i \in \{1, \dots, 5\}$ et pour tout $j \in \{1, \dots, 6\}$ si $j \in h_i$ (le chercheur i est libre durant le créneau j). Capacité de l'arc égale à 1 ;
     - (z_j, t) pour tout $j \in \{1, \dots, 6\}$ avec une capacité égale à 2 (nombre maximal de chercheurs pouvant être accueillis par un même ordinateur durant le même créneau horaire).

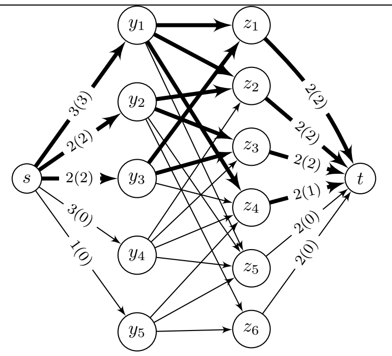
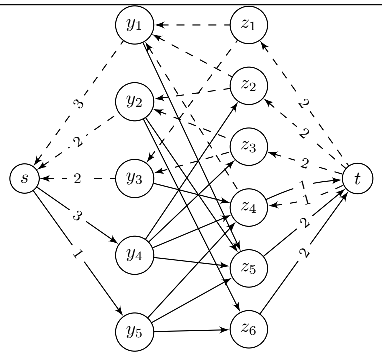
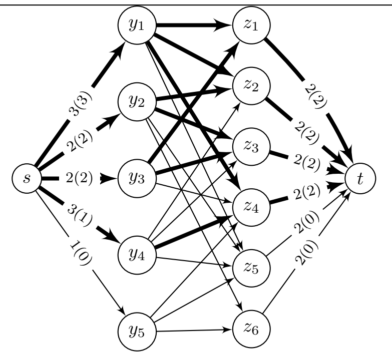
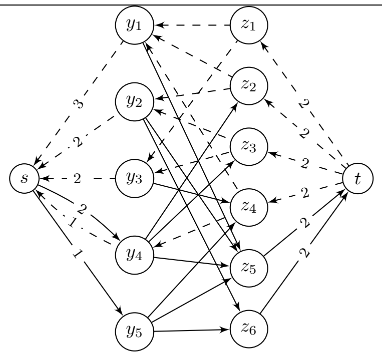
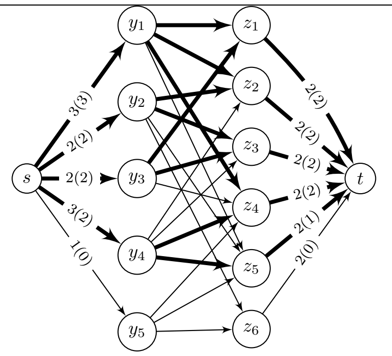
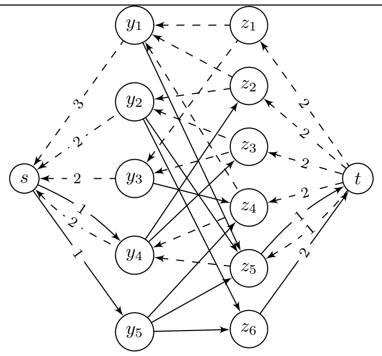
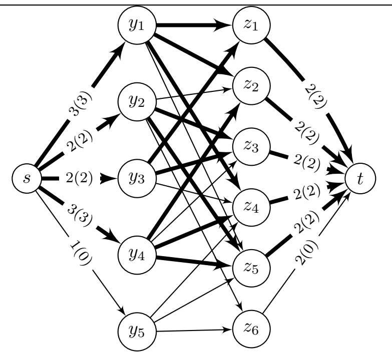
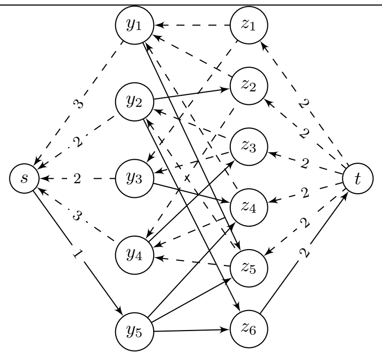
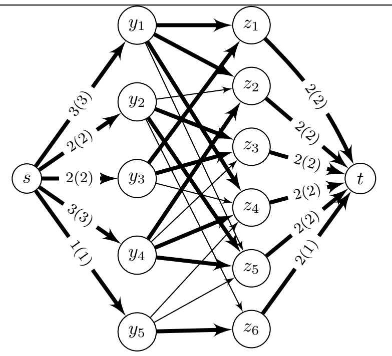
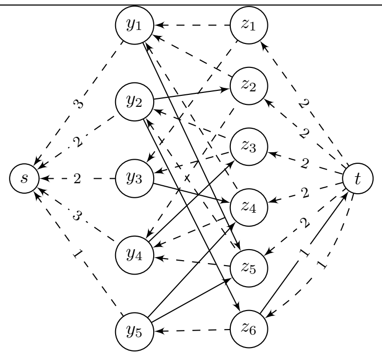


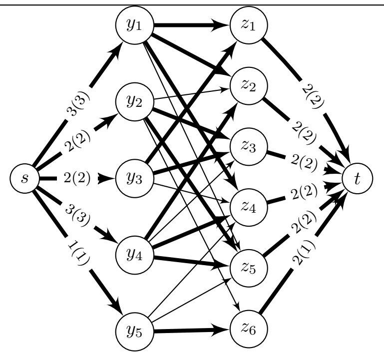
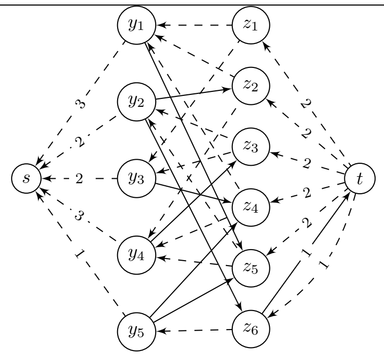
2. Afin de résoudre le problème, on applique l'algorithme de Ford-Fulkerson pour trouver le flot maximum dans ce réseau de flot.

   Par soucis de lisibilité, est considéré dans la trace d'exécution de l'algorithme ce qui suit :
   - Les arcs épais sont des arcs associés à des quantités de flot non nulles.
   - Dans le réseau de flot de chaque itération, les capacité et quantités de flot ne sont pas indiquées de manière explicite sur les arcs (y_i, z_i). La capacités de chacun de ces arcs est égale à 1 et lorsqu'il transporte une quantité de flot, i.e. lorsque l'arc sera épais, cette quantité de flot sera toujours égale à 1.
   - Dans le réseau résiduel de chaque itération, la capacité résiduelle de chaque arc direct (y_i, z_i) ou bien inverse (z_i, y_i), lorsqu'ils existent, est égale à 1.

$term$	P	Δ	flot	G_f
$faux$	-	-		
$faux$	(s, y_1, z_1, t)	min(3,1,2)=1		

$faux$	(s, y_1, z_2, t)	min(2,1,2)=1		
$faux$	(s, y_1, z_4, t)	min(1,1,2)=1		
$faux$	(s, y_2, z_2, t)	min(2,1,1)=1		
$faux$	(s, y_2, z_3, t)	min(2,1,1)=1		
$faux$	(s, y_3, z_1, t)	min(2,1,1)=1		

$faux$	(s, y_3, z_3, t)	min(1,1,1)=1		
$faux$	(s, y_4, z_4, t)	min(3,1,1)=1		
$faux$	(s, y_4, z_5, t)	min(2,1,2)=1		
$faux$	$(s, y_4, z_2, y_2, z_5, t)$	min(1,1,1,1,1) =1		
$faux$	(s, y_5, z_6, t)	min(1,1,2)=1		

vrai	-	-		

Ainsi, les chercheurs pourront être programmés comme suit :

— Chercheur 1 : aux créneaux 1, 2 et 4 ;

— Chercheur 2 : aux créneaux 3 et 5 ;

— Chercheur 3 : aux créneaux 1 et 3 ;

— Chercheur 4 : aux créneaux 2, 4 et 5 ;

— Chercheur 5 : au créneau 6.

CHAPITRE 3

NOTIONS DE BASE DES ARBRES

3.1 Introduction

Les arbres sont un type particulier de graphes qui se prêtent bien à la représentation de bon nombre de formats de données : relations hiérarchiques, arbres généalogiques, etc. Ils ont également plusieurs applications : routage de données dans un réseau, analyses d'algorithmes, analyse d'expressions algébriques, etc.

Les arbres peuvent être orientés ou non. Seront discutés pour ce cours uniquement ces derniers.

Nous nous intéresserons à leur définition et à quelques unes de leurs propriétés, puis au problème de l'arbre couvrant d'un graphe. L'algorithme de Kruskal pour l'identification d'un arbre couvrant minimal d'un graphe sera présenté.

3.2 Définitions

Un **arbre** est un graphe connexe sans cycle.

Une **forêt** est une collection d'un ou de plusieurs arbres.

Une **feuille** dans un arbre est un sommet de degré 1. [1]

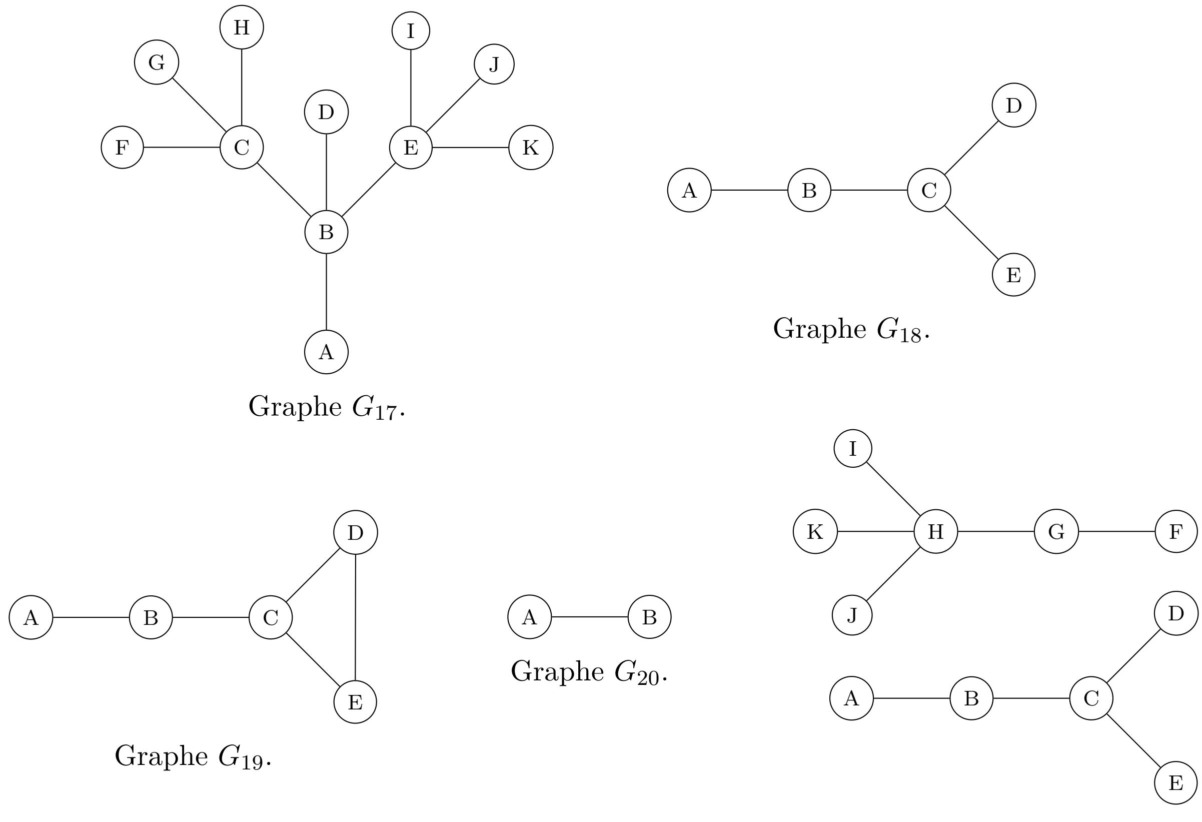
Exemples Dans les différents graphes donnés à la figure 3.1, certains sont des arbres alors que d'autres non.

Ainsi, le graphe G_{17} est bien connexe et sans cycle, il représente donc bien un arbre. Celui-ci comporte 8 feuilles : les sommets $\{$A, D, F, G, H, I, J, K$\}$.

Les graphes G_{18} et G_{20} sont également des arbres.

Le graphe G_{19} n'est pas un arbre car il contient le cycle (C,D,E,C).

Enfin, le graphe G_{21} est constitué de deux arbres. Il représente donc une forêt.



Graphe G_{17}. Graphe G_{18}. Graphe G_{19}. Graphe G_{20}. Graphe G_{21}.

FIGURE 3.1 – Arbre ou non ?

Exemple Un exemple d'utilisation de la notion d'arbre est la représentation graphique des matchs d'un tournois de sport à élimination directe - coupe. Un tel arbre est donné à la figure 3.2.

Dans cet arbre d'ordre 7, chaque sommet représente un match. Le premier niveau -le plus à gauche- représente les quarts de finale et il comporte 4 matchs, chaque match étant constitué de deux équipes. À l'issue de chaque match, le vainqueur appartiendra au sommet du niveau qui suit sur la chaîne qui relie ce sommet au sommet de la finale, pour constituer les sommets des demi-finales. Et ainsi de suite jusqu'à arriver à la finale.

Les mentions "quart de finale", "demi finale" et "finale" n'appartiennent bien évidemment pas au graphe.

3.3 Propriétés

> Tout arbre A d'ordre N, possède $N - 1$ arêtes.

Démonstration. La propriété peut être prouvée par récurrence :
Pour $N = 1$, l'arbre sera un arbre à un seul sommet et $1 - 1 = 0$ arête.

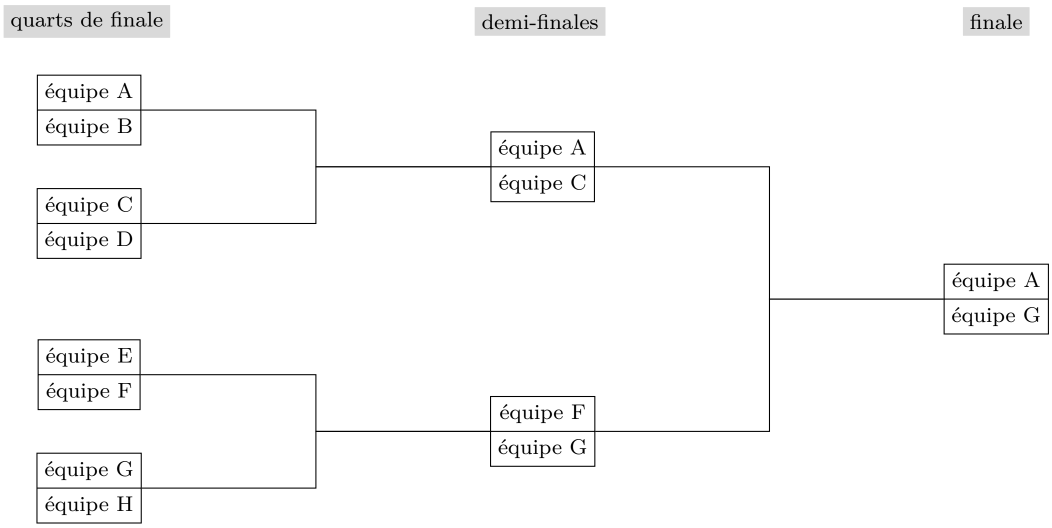


FIGURE 3.2 – Arbre G_{22} représentant les matchs d'un tournois à élimination directe.

Supposons que la propriété est vraie pour n'importe quel arbre d'ordre i avec $1 < i < N$, et montrons qu'elle l'est aussi pour un arbre A d'ordre N.
En supprimant n'importe quelle arête de A, nous obtenons deux arbres A_1 et A_2, le premier d'ordre $N_1 < N$ et le deuxième d'ordre $N_2 < N$, (avec $N_1 + N_2 = N$). Ces deux arbres ont respectivement $N_1 - 1$ et $N_2 - 1$ arêtes.
On en déduit que A possède $N_1 - 1 + N_2 - 1 + 1 = N - 1$ arêtes. □

Toute forêt d'ordre N contenant k composantes connexes, possède $N - k$ arêtes.

Démonstration. Une forêt contenant k arbres respectivement d'ordre $N_1, N_2, \ldots, N_k$, possède $N_1 - 1 + N_2 - 1 + \ldots + N_k - 1 = N - k$ arêtes. □

Un graphe d'ordre N est un arbre si et seulement s'il est connexe et possède $N - 1$ arêtes.

Démonstration. Une démonstration doit être faite pour chacun des deux sens de l'équivalence.
Premier sens de l'équivalence : si un graphe d'ordre N est un arbre, alors il est connexe par définition, et nous avons déjà vu qu'il contient $N - 1$ arêtes.
Deuxième sens de l'équivalence : si un graphe est connexe, qu'il contient $N - 1$ arêtes et qu'il n'est pas un arbre, cela veut dire qu'il contient des cycles. En supprimant autant d'arêtes que nécessaire pour éliminer tous les cycles du graphe, nous obtiendrons un arbre avec moins de $N - 1$ arêtes, ce qui est contradictoire avec la première propriété vue. □

Un graphe d'ordre N est un arbre si et seulement s'il est sans cycle et possède $N-1$ arêtes.

Démonstration. Une démonstration doit être faite pour chacun des deux sens de l'équivalence.
Premier sens de l'équivalence : si un graphe d'ordre N est un arbre alors il est sans cycle par

définition et nous avons déjà vu qu'il contient $N-1$ arêtes.
Deuxième sens de l'équivalence : si un graphe G d'ordre N est sans cycle, qu'il contient $N-1$ arêtes et qu'il n'est pas un arbre, cela veut dire qu'il n'est pas connexe.
Si G n'est pas connexe, alors il contient $k>1$ composantes connexes sans cycle, c.à.d, k arbres formant une forêt à $N-k$ arêtes, ce qui contradictoire avec l'hypothèse de départ. □

Tout arbre d'ordre $N \geq 2$, possède au moins 2 feuilles.

Démonstration. La propriété peut être prouvée par l'absurde :
Supposons qu'on a un arbre $A=(X,U)$ d'ordre N avec moins de 2 feuilles, c.à.d. 0 ou 1 feuille.
En utilisant le lemme des poignées de mains, nous obtenons que $\sum_{x\in X} d(x) = 2N-2$.
Si A possédait 0 feuille, alors $\forall x \in X, d(x) \geq 2$, et donc $\sum_{x\in X} d(x) \geq 2N$, ce qui est contradictoire avec ce qui précède.
Si A possédait 1 feuille, alors $\exists x' \in X, d(x') = 1$ et $\forall x \in X - \{x'\} : d(x) \geq 2$, et donc $\sum_{x\in X} d(x) \geq 2N-1$, ce qui est aussi contradictoire avec ce qui précède. □

Exemple Les propriétés données ci-dessus peuvent facilement être vérifiées sur les exemples précédents.

3.4 Arbre couvrant

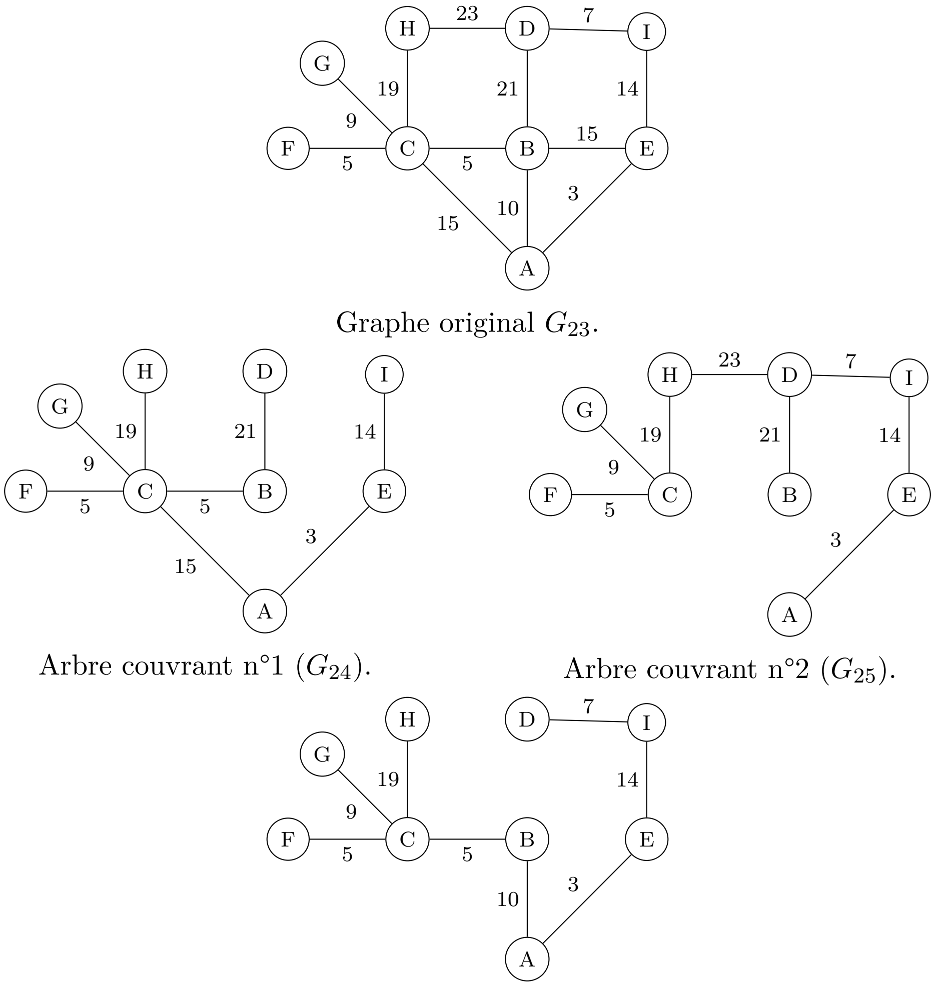
Un **arbre couvrant** d'un graphe connexe G est un arbre partiel de G.

Exemple La figure 3.3 montre l'exemple d'un graphe G_{23} avec trois exemples d'arbres couvrants G_{24}, G_{25} et G_{26}.

Soit G un graphe valué connexe. Un **arbre couvrant minimal** de G est un arbre couvrant dont la somme des longueurs des arêtes est inférieure ou égale à la somme des longueurs des arêtes de n'importe quel autre arbre couvrant de G.

Exemples G_{24} est l'arbre couvrant minimal du graphe G_{23}. La somme des valeurs associées à ses arêtes qui est égale à 72 est inférieure à la somme des valeurs associées aux arêtes de n'importe quel autre arbre couvrant.

Un exemple montrant l'intérêt du problème de recherche de l'arbre couvrant minimal serait le suivant : on désire relier par un réseau de télécommunication filaire toutes les maisons d'un nouveau quartier en utilisant le minimum possible de longueur de câble. Une solution optimale serait un arbre couvrant minimal à partir du graphe complet dont les sommets seraient les maisons et les arêtes auraient pour longueur la distance entre les maisons.



Graphe original G_{23}.

Arbre couvrant n°1 (G_{24}).

Arbre couvrant n°2 (G_{25}).

Arbre couvrant n°3 (G_{26}).

FIGURE 3.3 – Arbres couvrants

3.4.1 L'algorithme de Kruskal

L'algorithme de Kruskal est l'un des algorithmes les plus populaires pour l'identification de l'arbre couvrant minimal d'un graphe non orienté connexe valué. Son principe est simple : sélectionner les arêtes par poids croissant si elles ne forment pas de cycles avec les arêtes précédemment sélectionnées.

Son pseudo-code est donné par l'algorithme 4.

Sont utilisées pour cet algorithme les variables suivantes :

— $\overline{T}$: ensemble des arêtes non marquées ;

— U' : ensemble des arêtes de l'arbre couvrant minimal ;

— u : arête sélectionnée à chaque itération.

Détails de l'algorithme ligne par ligne :

— Lignes 1 à 2 : Lignes d'initialisation :

  — Ligne 1 : $\overline{T}$ est initialisé avec l'ensemble U. Au départ, aucune arête n'est marquée.

  — Lignes 2 : U', l'ensemble des arêtes de l'arbre couvrant minimal à construire est vide au départ.

Algorithm 4 Algorithme de Kruskal

Entrées :

$G = (X, U)$: graphe non orienté connexe valué par des valeurs $l(u)$ pour tout $u \in U$ et d'ordre N.

Résultats :

$A = (X, U')$: arbre couvrant minimal de G.

▷ Initialisation

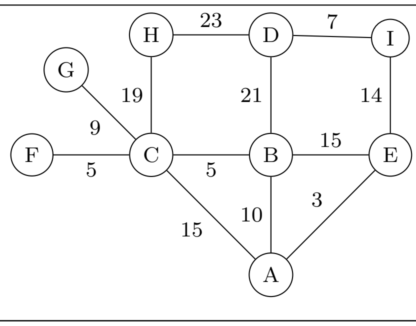
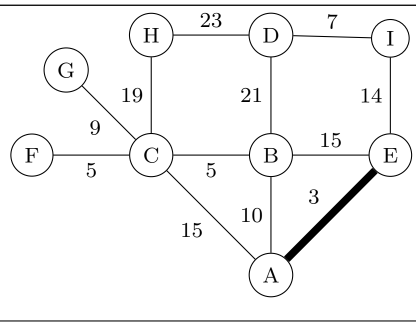
1: $\overline{T} = U$
2: $U' = \emptyset$

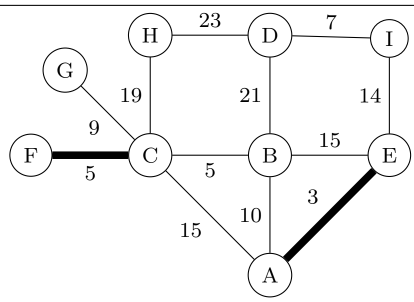
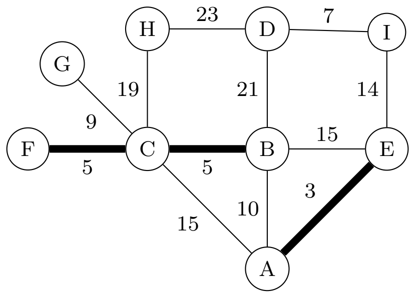
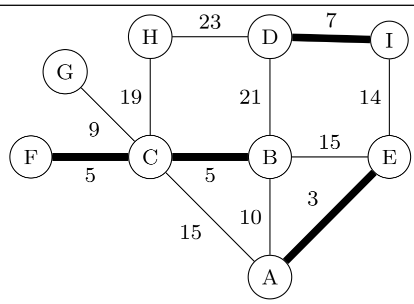
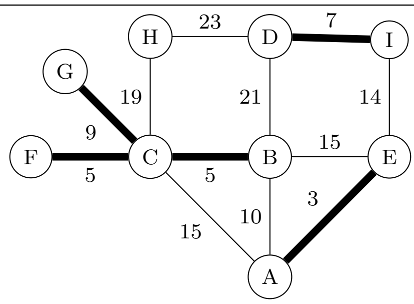
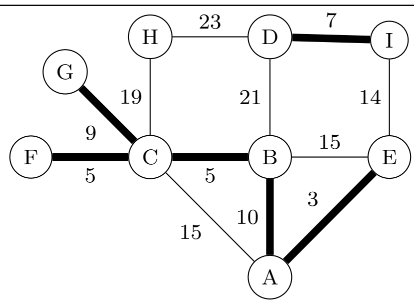
▷ Itérations de l'algorithme

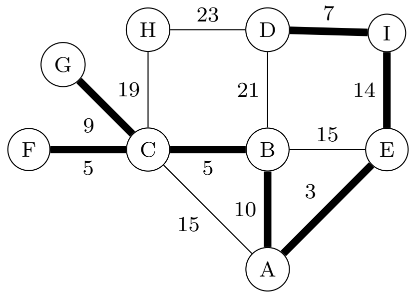
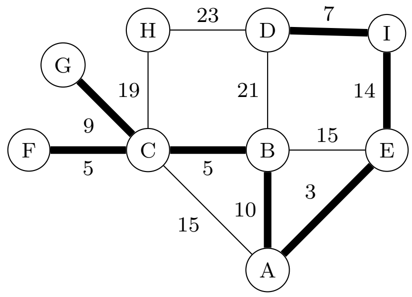
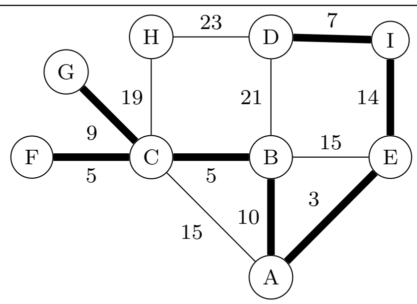
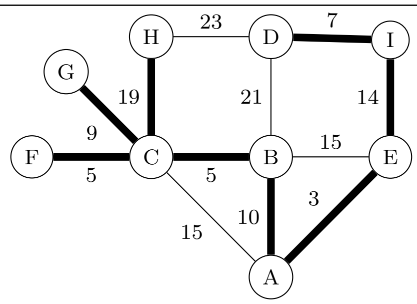
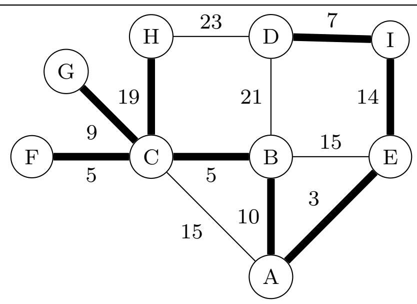
3: **tant que** $|U'| < N - 1$ **faire**
4: $\quad u =$ arête de $\overline{T}$ ayant la plus petite valeur $l()$
5: $\quad$ **si** $U' \cup \{u\}$ ne contient pas de cycle **alors**
6: $\quad\quad U' = U' \cup \{u\}$
7: $\quad$ **fin si**
8: $\quad \overline{T} = \overline{T} - \{u\}$
9: **fin tant que**

— Lignes 3 à 9 : Boucle qui va se répéter tant que A n'est pas un arbre couvrant :
  — Ligne 4 : Une arête u non marquée et qui possède la plus petite valeur $l()$ est sélectionnée.
  — Lignes 5-7 : u est ajoutée à U' si cela ne provoque pas de cycle.
  — Lignes 8 : Dans tous les cas, u est marquée.

Exemple Ci-dessous, la trace de l'exécution de l'algorithme de Kruskal sur le graphe G_{23} d'ordre $N = 9$. Une arête ajoutée à l'ensemble U' est précisée en trait gras à chaque itération.

$\|U'\| < 8$	u	$l(u)$	cycle	A
oui	-	-	-	
oui	(A, E)	3	non	

<table>
<tr><td>oui</td><td>(F,C)</td><td>5</td><td>non</td><td>
</td></tr>
<tr><td>oui</td><td>(C,B)</td><td>5</td><td>non</td><td>
</td></tr>
<tr><td>oui</td><td>(D,I)</td><td>7</td><td>non</td><td>
</td></tr>
<tr><td>oui</td><td>(C,G)</td><td>9</td><td>non</td><td>
</td></tr>
<tr><td>oui</td><td>(B,A)</td><td>10</td><td>non</td><td>
</td></tr>
</table>

oui	(I, E)	14	non	
oui	(A, C)	15	oui	
oui	(B, E)	15	oui	
oui	(C, H)	19	non	
non	-	-	-	

3.5 Exercices

Exercice 29

Tracer ce qui suit si cela est possible. Dans le cas contraire, justifier l'impossibilité :

1. Un arbre d'ordre 10 avec exactement 12 arêtes.
2. Un arbre d'ordre 12 avec exactement 10 arêtes.
3. Une forêt d'ordre 13 avec exactement 13 arêtes.
4. Une forêt d'ordre 13 avec exactement 12 arêtes.
5. Une forêt d'ordre 13 avec exactement 11 arêtes.

Exercice 30

Quelle est la somme des degrés d'un arbre d'ordre n ?

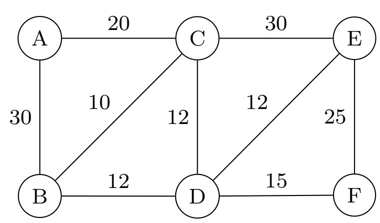
Exercice 31

Déterminer si chacune des séquences suivantes peut représenter les degrés d'un arbre :

1. $(4, 1, 1, 1, 1)$
2. $(3, 3, 2, 1, 1)$
3. $(2, 2, 1, 1)$
4. $(4, 4, 3, 3, 3, 2, 2, 1, 1, 1, 1, 1, 1, 1)$

Exercice 32

Trouver un arbre couvrant minimal au graphe suivant :



3.6 Solutions des exercices

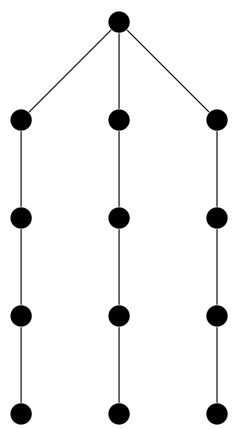
Solution de l'exercice 29

1. Dans un arbre d'ordre n, le nombre d'arêtes est égal à $n-1$. Il n'est donc pas possible de tracer un arbre d'ordre 10 avec 12 arêtes car $12 \neq 10-1=9$.

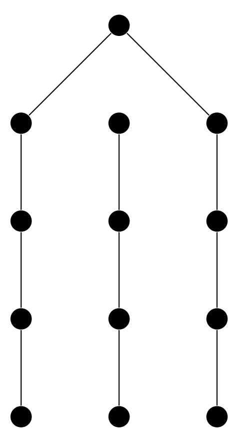
2. Pour la même raison précédente, il n'est pas possible de tracer un arbre d'ordre 12 avec 10 arêtes car $10 \neq 12-1=11$.

3. Dans une forêt d'ordre n à k composantes connexes, le nombre d'arêtes est égal à $n-k$ (avec $k \geq 1$ donc). Il n'est donc pas possible de tracer une forêt d'ordre 13 avec 13 arêtes car $\nexists k \geq 1/13 = 13-k$.

4. Une telle forêt peut être tracée et correspond à un seul arbre, i.e. une seule composante connexe (12=13-1).



5. Une telle forêt peut être tracée et correspond à deux arbres, i.e. deux composantes connexes (11=13-2).

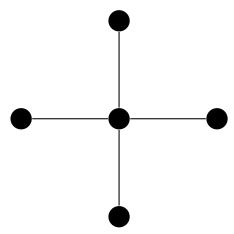


Solution de l'exercice 30

Pour un arbre $A=(X,U)$, nous avons : $|U|=n-1$ et $\sum_{x\in X} d(x) = 2|U|$. Ceci implique que la somme des degré de n'importe quel arbre d'ordre n est égal à $2n-2$.

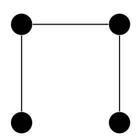
Solution de l'exercice 31

1. Oui, on peut tracer un arbre avec cette séquence de degrés :



2. La somme des 5 degrés est égale à 3+3+2+1+1=10, ce qui impliquerait l'existence de 10/2=5 arêtes. Or, dans un arbre d'ordre 5, le nombre d'arêtes est égal à 4. Cette séquence ne peut donc pas pas représenter un arbre.

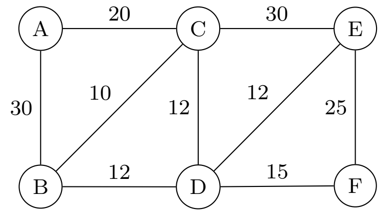
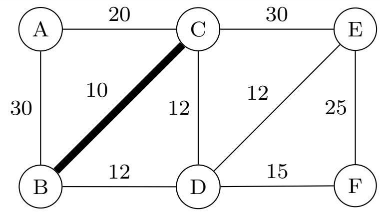
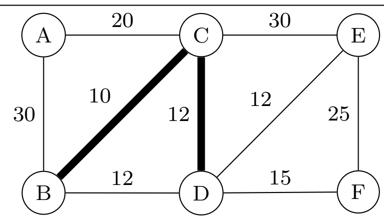
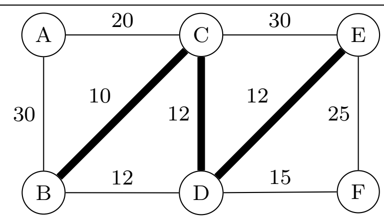
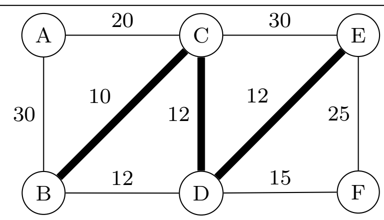
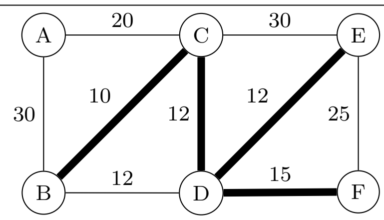
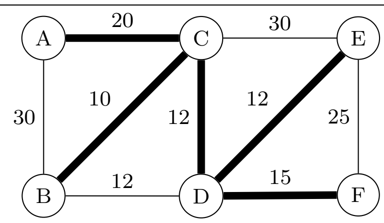
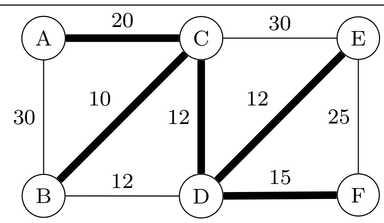
3. Oui, on peut tracer un arbre avec cette séquence de degrés :



4. Même chose que la séquence 2. La somme des 14 degrés est égale à 28, ce qui impliquerait l'existence de 28/2=14 arêtes. Or, dans un arbre d'ordre 14, le nombre d'arêtes est égal à 13. Cette séquence ne peut donc pas pas représenter un arbre.

Solution de l'exercice 32

Un arbre couvrant minimal peut être trouvé en appliquant l'algorithme de Kruskal sur le graphe donné :

$\lvert U' \rvert < 5$	u	$l(u)$	cycle	A
oui	-	-	-	
oui	(B,C)	10	non	
oui	(C,D)	12	non	
oui	(D,E)	12	non	
oui	(B,D)	12	oui	
oui	(D,F)	15	non	
oui	(A,C)	20	non	
non	-	-	-	

CHAPITRE 4

INTRODUCTION À LA PROGRAMMATION LINÉAIRE

4.1 Introduction

Tout comme la théorie des graphes, la programmation linéaire est un outil de la recherche opérationnelle qui permet la résolution de problèmes et aide à la prise de décision.

La programmation linéaire permet la résolution de problèmes qui ont généralement la forme suivante : allouer des ressources limitées à plusieurs activités concurrentes, de la meilleure manière possible [5]. La programmation linéaire trouve des applications dans divers domaines : production industrielle, planification de l'agriculture, etc.

La programmation linéaire utilise un modèle mathématique pour la modélisation du problème concerné. Le terme linéaire indique que toutes les fonctions d'un tel modèle doivent être sous forme linéaire. Le terme programmation indique que l'outil aide à réaliser de la planification.

4.2 Programme linéaire

Un programme linéaire (PL) est constitué de :

— **Variables de décision** : ce qu'on doit décider. Il s'agit de valeurs réelles.

— Une **fonction objectif** : ce qu'on doit optimiser. Il s'agit d'une fonction linéaire à minimiser ou bien à maximiser, appliquée sur les variables de décision.

— **Contraintes** : ce qu'on doit respecter. Elles doivent également être linéaires en fonction des variables de décision. Elles peuvent être sous forme d'équation ($=$) ou d'inégalité ($\leq$ ou $\geq$).

Problème Un artisan chocolatier décide de confectionner deux sortes de tablettes de chocolat, des tablettes classiques et des tablettes de luxe. En allant inspecter ses réserves, il constate qu'il lui reste 18 kg de cacao, 8 kg de noisettes et 14 kg de lait. Une tablette classique nécessite 20 g de cacao, 20 g de noisettes, et 40 g de lait. Une tablette de luxe nécessite 60 g de cacao, 20 g de noisettes, et 20 g de lait. Il fera un profit de 20 DA en vendant une tablette classique, et un profit de 30 DA en vendant une tablette de luxe. Combien doit-il fabriquer de tablettes classiques et de luxe pour obtenir le plus grand bénéfice possible ?

Modélisation Pour ce problème, il y a deux variables sur lesquelles une décision doit être prise par l'artisan, i.e. nous avons deux variables de décision :

— x_1 : nombre de tablettes classiques à produire ;

— x_2 : nombre de tablettes de luxe à produire.

L'objectif est de maximiser le bénéfice total. Il s'agit donc d'un problème de maximisation, et la fonction objectif à optimiser est :

$$z = 20x_1 + 30x_2 \tag{4.1}$$

En ce qui concerne les contraintes du problème, nous en avons une pour chaque ressource, c'est-à-dire, pour chaque ingrédient dans les réserves de l'artisan.

Nous avons 18 kg de cacao. La fabrication d'une tablette classique en nécessite 20 g. La fabrication de x_1 tablettes classiques, nécessite donc $20 \times x_1$ g. La fabrication d'une tablette de luxe nécessite 60 g de cacao. La fabrication de x_2 tablettes classiques, nécessite donc $60 \times x_2$ g. La quantité totale de cacao à utiliser ne devant pas dépasser la quantité en réserve, nous obtenons la contrainte dite **contrainte fonctionnelle** :

$$20x_1 + 60x_2 \leq 18000 \tag{4.2}$$

En appliquant le même raisonnement sur les noisettes et le lait, nous obtenons les deux nouvelles contraintes fonctionnelles :

$$20x_1 + 20x_2 \leq 8000 \tag{4.3}$$

$$40x_1 + 20x_2 \leq 14000 \tag{4.4}$$

De plus, nous avons des **contraintes de positivité** sur les variables de décision : il est insensé qu'elles soient négatives. Nous ajoutons donc les contraintes :

$$x_1 \geq 0 \tag{4.5}$$

$$x_2 \geq 0 \tag{4.6}$$

En résumé, le programme linéaire complet modélisant notre problème est donné comme suit :

$$\begin{array}{rl} \text{maximiser} & z = 20x_1 + 30x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} 20x_1 + 60x_2 \leq 18000 \\ 20x_1 + 20x_2 \leq 8000 \\ 40x_1 + 20x_2 \leq 14000 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array} \tag{4.7}$$

Remarque Les variables de décision pour ce problème ne sont pas censées être réelles mais entières. À sa résolution en tant que programme linaire, nous avons deux cas :

— La solution optimale obtenue est entière => le résultat est optimal et exact pour le problème.

— La solution optimale obtenue n'est pas entière => le résultat obtenu permettra d'avoir une approximation de la solution entière optimale pour le problème.

4.3 Résolution graphique

La résolution graphique d'un programme linéaire peut être appliquée pour des problèmes avec un maximum de 3 variables de décision. La résolution est facile pour une ou deux variables (résolution sur une droite ou un plan). Elle est un peu moins évidente pour 3 variables de décision car il faudra passer à une géométrie à 3 dimensions.

Pour un problème à deux variables de décision, la résolution graphique d'un PL consiste à suivre les étapes suivantes :

— Considérer un plan et tracer ses deux axes, chaque axe représentant une variable de décision.

— Pour chaque contrainte, tracer la droite correpondante et éliminer la partie du plan qui ne respecte pas la contrainte.

— À cette étape, ce qui n'a pas été éliminé du plan correspond à la **région réalisable** du PL. Tous les points qu'ils contient sont des **solutions réalisables** du PL, et tous les autres points sont dits **solutions non réalisables**.

— Tracer la droite qui correspond à une première valeur pour la fonction objectif (par exemple, la droite $z = 0$).

— Deux cas se présentent :

  — S'il s'agit d'un problème de maximisation, identifier le sens de l'augmentation de la fonction objectif z : prendre une nouvelle valeur pour la fonction objectif et tracer sa droite (par exemple, la droite $z = 10$). Celle-ci sera parallèle à la droite $z = 0$ étant donné qu'elles ont la même pente. La position où se trouve la droite $z = 10$ par rapport à la droite $z = 0$ correspond au sens de l'augmentation de la fonction objectif. La(les) **solution(s) optimale(s)** correspondra/correspondront au(x) point(s) qui sera/seront compris dans la droite parallèle ayant été le plus poussée vers le sens de l'augmentation de la fonction objectif, tout en faisant partie de la région réalisable. La droite comprenant la/les solution(s) optimale(s) correspondra à une valeur de fonction objectif optimale et qui sera notée z^*.

  — S'il s'agit d'un problème de minimisation, suivre le raisonnement inverse.

Remarque :

— La résolution graphique de PLs à 1 ou 3 variables de décision suit le même principe.

— Un point se trouvant à l'intersection de limites de plusieurs contraintes est dit **sommet**. Si ce sommet appartient à la région réalisable, il est dit **sommet réalisable**.
— La solution optimale d'un problème linéaire est souvent un sommet. Le problème peut également avoir plusieurs solutions correspondant à un segment (où même encore une droite ou une demi-droite) dans le cas où la droite de la fonction objectif est parallèle à la droite de l'une des contraintes du PL.
— Un PL pour lequel la région réalisable est vide est dit **PL non réalisable**.
— Un PL de maximisation/minimisation pour lequel l'augmentation/diminution du z est illimitée est dit **PL non borné**.

Exemple Résolvons notre problème précédent graphiquement sur un plan à deux axes correspondant aux deux variables de décision x_1 et x_2.

Nous commençons tout d'abord par identifier la région réalisable. Pour ce faire, toutes les solutions, i.e. points du plan, ne respectant pas une ou plusieurs contraintes du PL sont éliminées.

En ce qui concerne la contrainte (4.2), i.e. celle du cacao, la droite $20x_1 + 60x_2 = 18000$ doit tout d'abord être tracée en s'aidant de deux points appartenant à cette droite. Par exemple les points (0,300) et (900,0). Étant donné qu'il s'agit d'une contrainte sous forme de ($\leq$), le demi-plan ne respectant pas cette contrainte doit être éliminé. Pour identifier ce demi-plan, on peut utiliser un point quelconque n'appartenant pas à la droite tracée. Si ce point respecte la contrainte, alors le demi-plan le contenant respecte la contrainte et l'autre demi-plan est éliminé. Si le point ne respecte pas la contrainte, alors c'est ce demi-plan qui est éliminé.

En choisissant le point (0,0) par exemple, la contrainte (4.2) est respectée car $20 \times 0 + 60 \times 0$ est bien inférieur ou égal à 18000. Le demi-plan ne contenant pas le point (0,0) est donc éliminé (voir la figure 4.1).

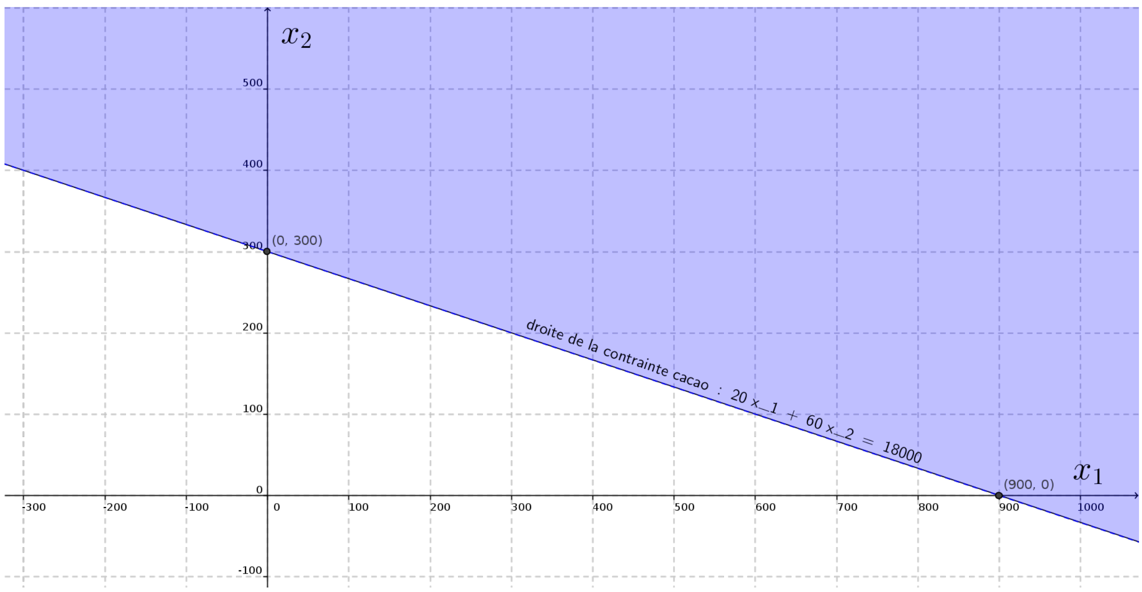


FIGURE 4.1 – Élimination du demi-plan ne respectant pas la contrainte (4.2).

De la même manière, toutes les autres solutions non réalisables, i.e. celles qui ne respectent pas la contrainte (4.3) ou la contrainte (4.4) ou une des contraintes de positivité, sont également éliminées (voir la figure 4.2).

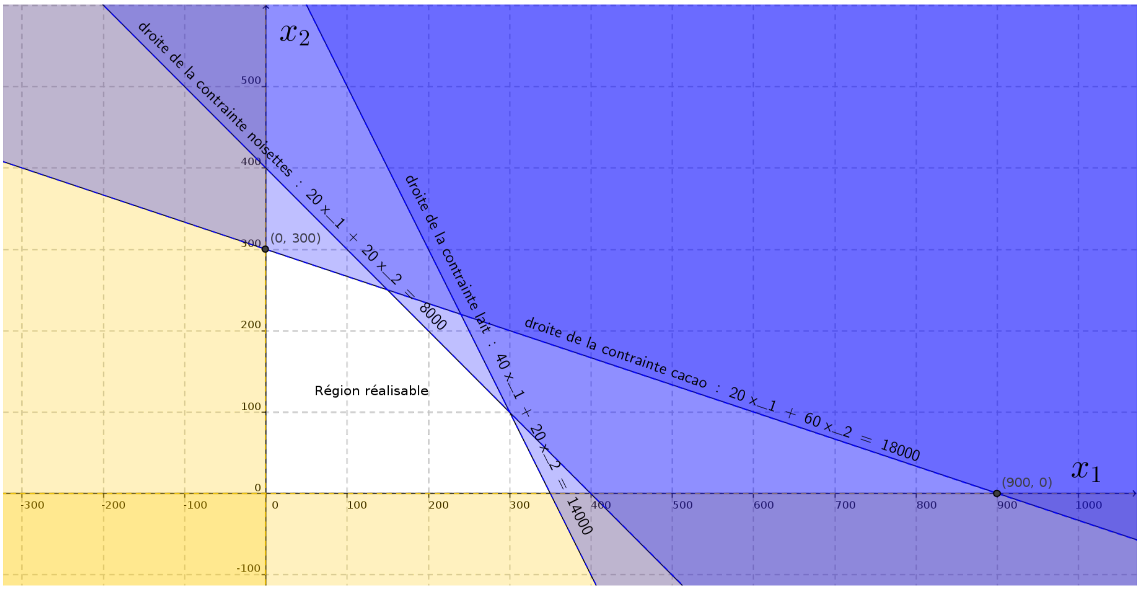


FIGURE 4.2 – Identification de la région réalisable.

La région réalisable correspond alors à la zone du plan qui est restée blanche avec le polygone qui l'entoure. Tous les points qui y apparaissent correspondent à des solutions réalisables qui respectent donc toutes les contraintes du PL.

Il s'agit à présent d'identifier, parmi ces solutions réalisables, celle(s) qui est/sont optimale(s) par rapport à la fonction objectif.

On commence par tracer la droite $z = 0$ (voir la figure 4.3).

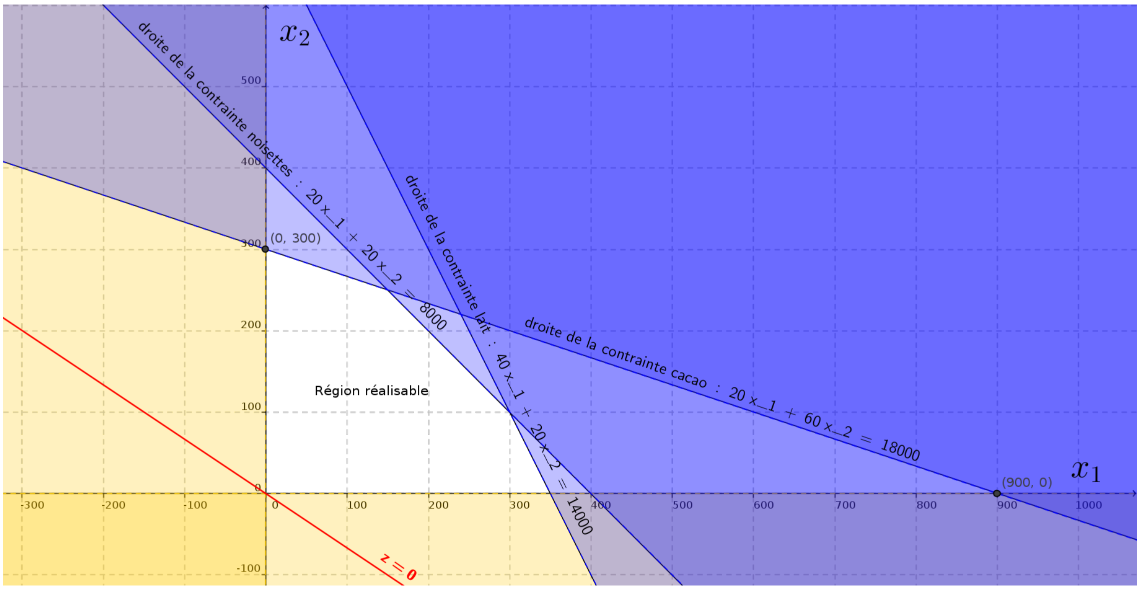


FIGURE 4.3 – Droite $z = 0$.

On identifie ensuite le sens de l'augmentation du z, étant donné qu'on a un problème de maximisation. On peut prendre par exemple $z = 3000$ et tracer sa droite (voir la figure 4.4). Cette droite est parallèle à la droite $z = 0$ étant donné qu'elles ont la même pente, et la position de cette nouvelle droite par rapport à la droite $z = 0$ indique le sens de l'augmentation du z.

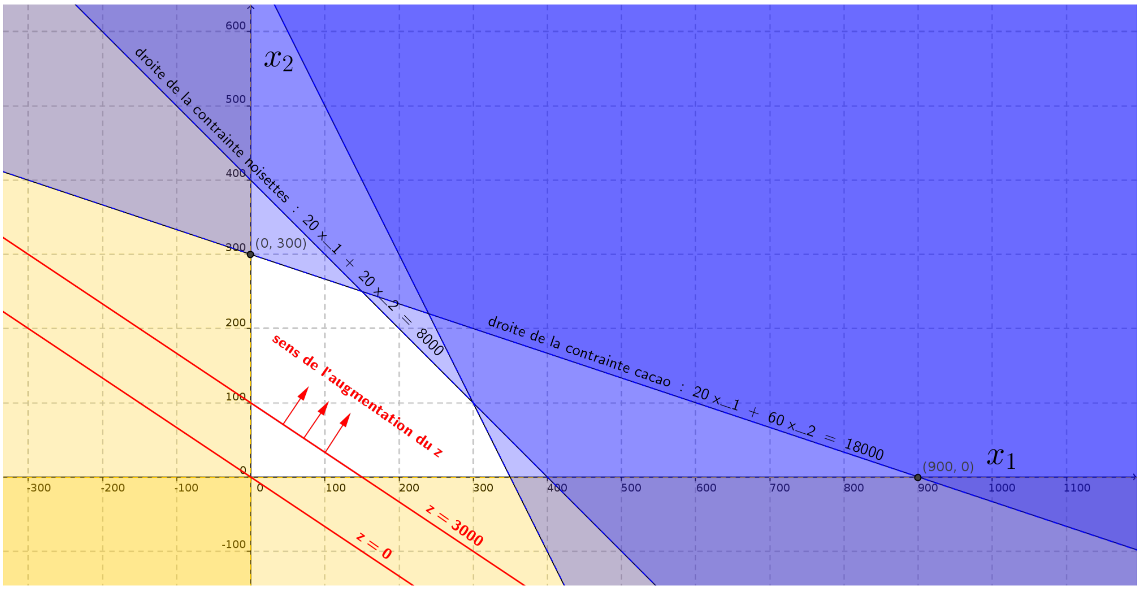


FIGURE 4.4 – Droite $z = 3000$ et sens de l'augmentation du z.

La position de la droite $z = 3000$ est plus haute que celle de la droite $z = 0$. Pour trouver le meilleur z, on identifie la droite parallèle aux deux précédentes droites, qui est la plus haute possible, tout en contenant au moins un point appartenant à la région réalisable.

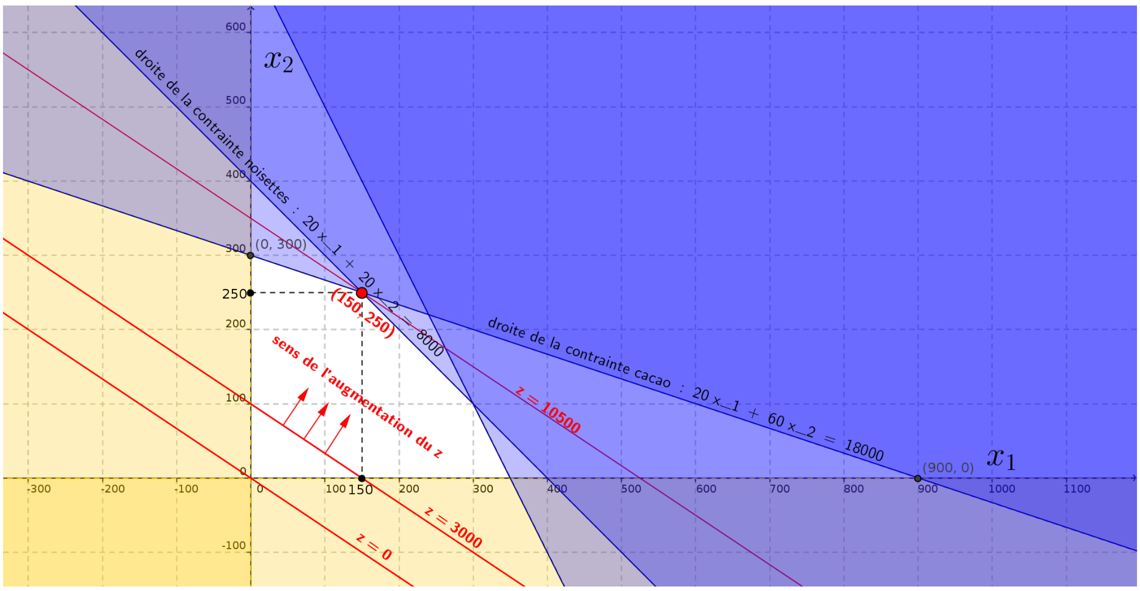


FIGURE 4.5 – Identification de la solution optimale.

Ceci correspond à la droite passant par le point (150, 250) (voir la figure 4.5). Ce dernier représente donc la solution optimale au PL. Afin d'obtenir la valeur optimale pour z, il suffit de remplacer les valeurs $x_1 = 150$ et $x_2 = 250$ dans la fonction objectif. Ceci nous donne une valeur de fonction objectif optimale $z^* = 20 \times 150 + 30 \times 250 = 10500$.

Enfin, pour en revenir à notre problème original, afin d'obtenir le meilleur profit qui soit et qui est égal à 10500 DA, l'artisan doit produire 150 tablettes classiques et 250 tablettes de luxe.

4.4 Forme standard

Un PL à n variables de décision et m contraintes est dit de forme standard pour notre cours s'il est écrit comme suit [5] :

$$\begin{array}{rl} \text{maximiser} & z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ \text{s.c.} & \left\{ \begin{array}{ll} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \leq b_2 \\ & \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \leq b_m \\ x_1, x_2, \ldots x_n \geq 0 & \end{array} \right. \end{array}$$

Exemple Le PL 4.7 précédent est écrit sous forme standard étant donné que :

— Il s'agit d'un problème de maximisation ;

— Toutes ses contraintes fonctionnelles sont de la forme ($\leq$) ;

— Il y une contrainte de positivité sur chacune de ses variables de décision.

4.4.1 Écriture matricielle de la forme standard

En considérant que :

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \mathbf{c} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

Un PL en forme standard peut être écrit sous forme matricielle comme suit :

$$\begin{array}{rl} \text{maximiser} & z = \mathbf{c}^\intercal \mathbf{x} \\ \text{s.c.} & \mathbf{Ax} \leq \mathbf{b} \text{ et } \mathbf{x} \geq 0 \end{array}$$

Exemple Pour notre PL précédent, nous avons :

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 20 & 60 \\ 20 & 20 \\ 40 & 20 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 18000 \\ 8000 \\ 14000 \end{bmatrix}, \mathbf{c} = \begin{bmatrix} 20 \\ 30 \end{bmatrix}.$$

Ce PL pourrait donc être écrit comme suit :

$$\begin{array}{rl} \text{maximiser} & z = \begin{bmatrix} 20 & 30 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{s.c.} & \begin{bmatrix} 20 & 60 \\ 20 & 20 \\ 40 & 20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 18000 \\ 8000 \\ 14000 \end{bmatrix} \text{ et } \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \end{array}$$

4.5 Méthode du Simplexe

La méthode du Simplexe est une méthode algébrique pour la résolution de programmes linéaires. Largement utilisé, des implémentations logicielles sophistiquées de la méthode sont disponibles pour la résolution de problèmes linéaires sur ordinateurs.

Pour notre cours, nous considérerons l'application du Simplexe sur des problèmes sous forme standard où tous les membres droits des contraintes fonctionnelles (les b_i) sont positifs.

4.5.1 Interprétation géométrique de la méthode du Simplexe

La méthode du Simplexe repose sur la visite de ses sommets réalisables d'un programme linéaire. Cependant, elle ne les visite pas tous et possède une condition d'arrêt dite **test d'optimalité**.

En considérant que deux sommets réalisables sont dits **adjacents** s'ils sont extrémités d'une même arête de la région réalisable, le test d'optimalité du Simplexe se base sur la propriété suivante :

> Soit un PL qui possède au moins une solution optimale. Si un sommet réalisable ne possède aucun sommet réalisable adjacent avec une meilleure valeur de la fonction objectif z, alors ce sommet est optimal.

Graphiquement, le fonctionnement du simplexe est le suivant :

1. considérer un sommet réalisable et la valeur de z qui lui correspond ;
2. chercher un sommet réalisable qui est adjacent au sommet précédent et qui correspond à une meilleure valeur pour z ;
3. si un tel sommet n'existe pas, alors, optimalité atteinte, arrêt ;
4. sinon, considérer le nouveau sommet et sa valeur de z et revenir à l'étape 2.

Exemple Notre précédent exemple possède 5 sommets réalisables : $(0, 0)$, $(0, 300)$, $(150, 250)$, $(300, 100)$ et $(350, 0)$. L'application de la méthode du Simplexe dessus correspond à ce qui suit :

- Commencer au sommet réalisable $(0, 0)$, il correspond à $z = 0$.
- Le sommet réalisable $(0, 0)$ possède deux sommets réalisables adjacents $(0, 300)$ et $(350, 0)$ avec de meilleures valeurs, respectivement, $z = 900$ et $z = 700$. Le sommet $(0,\ 0)$ n'est donc pas optimal. On sélectionne le sommet $(0, 300)$.
- Le sommet $(0, 300)$ possède un sommet réalisable adjacent $(150, 250)$ avec une meilleure valeur $z = 10500$. Le sommet $(0, 300)$ n'est donc pas optimal. On se place donc au niveau du sommet $(150, 250)$.
- Le sommet $(150, 250)$ ne possède aucun sommet réalisable adjacent avec une meilleure valeur pour z (les deux sommets réalisables adjacents $(0, 300)$ et $(300, 100)$ produisent tous les deux un $z = 900$). Le sommet $(150, 250)$ avec $z = 10500$ est donc optimal. On s'arrête.

4.5.2 Simplexe : résolution algébrique

La résolution algébrique d'un PL par le Simplexe est basée sur la résolution d'un système d'équations.

Forme canonique et variables d'écart

Avant de pouvoir appliquer le Simplexe sur un PL, il est nécessaire de le ré-écrire en forme dite **forme canonique** où toutes les contraintes fonctionnelles sont transformées en équations grâce à l'introduction de **variables d'écart**.

Ainsi, pour passer d'un PL en forme standard à m contraintes et n variables de décision à la forme canonique, une variable d'écart positive est ajoutée à chaque membre gauche des contraintes. Chaque variable d'écart quantifiera le surplus du membre droit par rapport au membre gauche de la contrainte fonctionnelle d'origine dans une solution donnée. Aussi, l'équation donnant la fonction objectif est ajoutée comme dernière contrainte avant les contraintes de positivité. On obtient alors la nouvelle forme du problème comme suit :

$$\begin{array}{ll}
\text{maximiser} & z \\
\text{s.c.} & \left\{ \begin{array}{ll}
a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + e_1 & = b_1 \\
a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + e_2 & = b_2 \\
 & \vdots \\
a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + e_m & = b_m \\
z - c_1x_1 - c_2x_2 - \ldots - c_nx_n & = 0 \\
x_1, x_2, \ldots x_n, e_1, e_2, \ldots e_m \geq 0 &
\end{array} \right.
\end{array}$$

Ce qui donne en forme matricielle :

$$\begin{aligned}
\text{maximiser} \quad & z \\
\text{s.c.} \quad & \mathbf{Ax} + \mathbf{e} = \mathbf{b} \\
& z - \mathbf{c}^\intercal \mathbf{x} = 0 \\
& \mathbf{x}, \mathbf{e} \geq 0
\end{aligned}$$

avec $\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_m \end{bmatrix}$.

On peut aussi écrire :

$$\begin{aligned}
\text{maximiser} \quad & z \\
\text{s.c.} \quad & \mathbf{Ax} + \mathbf{Ie} = \mathbf{b} \\
& z - \mathbf{c}^\intercal \mathbf{x} = 0 \\
& \mathbf{x}, \mathbf{e} \geq 0
\end{aligned}$$

avec **I** la matrice identité.

Remarque

- Une solution au PL d'origine auquel on ajoute les valeurs des variables d'écart de la forme canonique est dite **solution augmentée**.
- Une variable d'écart égale à 0 pour une certaine contrainte fonctionnelle dans une solution augmentée donnée signifie que cette dernière correspond à une solution qui se trouve au niveau de la limite de la contrainte concernée.
- Dans un PL en forme canonique à $n+m$ variables et m équations issues de contraintes fonctionnelles, le système d'équations possède n degrés de liberté. Ainsi, en fixant n variables à des valeurs arbitraires, les valeurs des m autres variables peuvent être obtenues en résolvant le système des m équations.

Exemple Reprenons le PL en forme standard construit précédemment :

$$\begin{array}{rl} \text{maximiser} & z = 20x_1 + 30x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} 20x_1 + 60x_2 \leq 18000 \\ 20x_1 + 20x_2 \leq 8000 \\ 40x_1 + 20x_2 \leq 14000 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

La ré-écriture du problème en forme canonique nécessite l'injection de 3 variables d'écart e_1, e_2 et e_3 :

$$\begin{array}{rl} \text{maximiser} & z \\ \text{s.c.} & \left\{ \begin{array}{l} 20x_1 + 60x_2 + e_1 = 18000 \\ 20x_1 + 20x_2 + e_2 = 8000 \\ 40x_1 + 20x_2 + e_3 = 14000 \\ z - 20x_1 - 30x_2 = 0 \\ x_1, x_2, e_1, e_2, e_3 \geq 0 \end{array} \right. \end{array} \tag{4.8}$$

Ainsi, si on fabrique $x_1 = 350$ tablettes classiques et $x_2 = 0$, la solution augmentée correspondante est $(x_1, x_2, e_1, e_2, e_3) = (350, 0, 11000, 1000, 0)$.

Pour la première contrainte fonctionnelle qui est celle du cacao, la valeur de la variable d'écart e_1 quantifie la quantité de cacao qui n'a pas été utilisée dans une solution donnée. La solution augmentée donnée plus haut avec $e_1 = 11000$ signifie qu'avec la fabrication de 350 tablettes classiques et 0 tablette de luxe il restera 11000 g de cacao en stock.

Solution de base

A chaque itération du Simplexe :

- n variables dites **variables hors base** sont fixées à 0 ;
- Les valeur des m autres variables peuvent être déduites. Ces variables constituent une **base** et sont dites **variables de base** ;
- Chaque solution augmentée ainsi constituée est une solution dite **solution de base** ;
- Graphiquement, cette solution correspond à un sommet ;
- Si la contrainte de positivité est vérifiée sur chaque variable de base de cette solution de base, cette dernière est dite **solution de base réalisable**. Elle correspond alors à un sommet réalisable. Dans le cas contraire, elle est dite **solution de base non réalisable**.
- Une **solution de base réalisable optimale** est une solution de base réalisable qui produit une valeur de la fonction objectif z qui est optimale.

Remarque À chaque itération du Simplexe, et selon la base considérée, le système d'équations est écrit de manière à ce que les variables de base et la fonction objectif soient écrites en fonction des variables hors base. Ainsi :

- La i-ème variable de base apparaitra uniquement au niveau de la i-ème équation et sera multiplié par le coefficient 1.
- La fonction objectif z apparaitra toujours au niveau de la dernière équation et sera multiplié par le coefficient 1.

Exemple En fixant les deux variables x_1 et x_2 à 0 comme variables hors base et en résolvant le système d'équations pour obtenir la valeur des variables de base e_1, e_2 et e_3 et celle de la fonction objectif z, comme ceci :

$$\begin{cases} x_1 = 0, x_2 = 0 \\ \quad 20x_1 + 60x_2 + \boldsymbol{e_1} = 18000 \\ \quad 20x_1 + 20x_2 + \boldsymbol{e_2} = 8000 \\ \quad 40x_1 + 20x_2 + \boldsymbol{e_3} = 14000 \\ z - 20x_1 - 30x_2 = 0 \end{cases} \tag{4.9}$$

nous obtenons :

$$\begin{cases} x_1 = 0, x_2 = 0 \\ e_1 = 18000 \\ e_2 = 8000 \\ e_3 = 14000 \\ z = 0 \end{cases} \tag{4.10}$$

Ce qui correspond à la solution de base $(x_1, x_2, e_1, e_2, e_3) = (0, 0, 18000, 8000, 14000)$ et à $z = 0$.

Toute les variables de base sont positives, il s'agit donc d'une solution de base réalisable, qui correspond au sommet réalisable $(0, 0)$.

De la même manière, en fixant les deux variables x_1 et e_1 à 0 comme variables hors base et en résolvant le système d'équations pour obtenir la valeur des variables de base x_2, e_2 et e_3 et celle de la fonction objectif z, comme ceci :

$$\begin{cases} x_1 = 0, e_1 = 0 \\ \frac{1}{3}x_1 + \boldsymbol{x_2} + \frac{1}{60}e_1 = 300 \\ \frac{40}{3}x_1 - \frac{1}{3}e_1 + \boldsymbol{e_2} = 2000 \\ \frac{100}{3}x_1 - \frac{1}{3}e_1 + \boldsymbol{e_3} = 8000 \\ z - 10x_1 + \frac{1}{2}e_1 = 9000 \end{cases} \tag{4.11}$$

nous obtenons :

$$\begin{cases} x_1 = 0, e_1 = 0 \\ x_2 = 300 \\ e_2 = 2000 \\ e_3 = 8000 \\ z = 9000 \end{cases} \tag{4.12}$$

Ce qui correspond à la solution de base $(x_1, x_2, e_1, e_2, e_3) = (0, 300, 0, 2000, 8000)$ et à un $z = 9000$.

Toutes les variables de base sont positives, il s'agit donc d'une solution de base réalisable, qui correspond au sommet réalisable $(0, 300)$.

D'un autre côté, en fixant les deux variables x_1 et e_2 à 0 comme variables hors base et en résolvant le système d'équations pour obtenir la valeur des variables de base x_2, e_1 et e_3 et celle de la fonction objectif z, comme ceci :

$$\begin{cases} x_1 = 0, e_2 = 0 \\ -40x_1 + \boldsymbol{e_1} - 3e_2 = -6000 \\ x_1 + \boldsymbol{x_2} + \frac{1}{20}e_2 = 400 \\ 20x_1 - e_2 + \boldsymbol{e_3} = 6000 \\ z + 10x_1 + \frac{3}{2}e_2 = 12000 \end{cases} \tag{4.13}$$

Nous obtenons :

$$\begin{cases} x_1 = 0, e_2 = 0 \\ e_1 = -6000 \\ x_2 = 400 \\ e_3 = 6000 \\ z = 12000 \end{cases} \tag{4.14}$$

Ce qui correspond à la solution de base $(x_1, x_2, e_1, e_2, e_3) = (0, 400, -6000, 0, 6000)$ et à un $z = 12000$.

Toutes les variables de base ne sont pas positives : $e_1 = -6000$. Il s'agit donc d'une solution de base non réalisable, qui correspond au sommet non réalisable $(0, 400)$.

Deux solutions de base sont dites **adjacentes** si leurs variables de base ne diffèrent que d'un seul élément.

Remarques

— Pour deux solutions de base adjacentes, les variables hors bases ne diffèrent également que d'un seul élément.

— Deux solutions de base adjacentes correspondent graphiquement à deux sommets adjacents.

— Se déplacer d'une solution de base réalisable actuelle vers autre solution de base réalisable adjacente implique faire sortir une variable de la base actuelle et la remplacer par une variable actuellement hors base (et ajuster les valeurs des variables en conséquence afin que le système d'équations reste satisfait).

Exemple Les deux solutions de bases $(0, 0, 18000, 8000, 14000)$ et $(0, 300, 0, 2000, 8000)$ sont adjacentes et elles correspondent aux deux sommets adjacents $(0, 0)$ et $(0, 300)$.

Les deux bases des solutions précédentes sont (e_1, e_2, e_3) et (x_2, e_2, e_3). Elles diffèrent uniquement en ce qui concerne e_1 pour la 1re et x_2 pour la 2e. Se déplacer de la 1re vers la 2e signifie faire sortir e_1 de la base et la remplacer par x_2.

Méthode du Simplexe

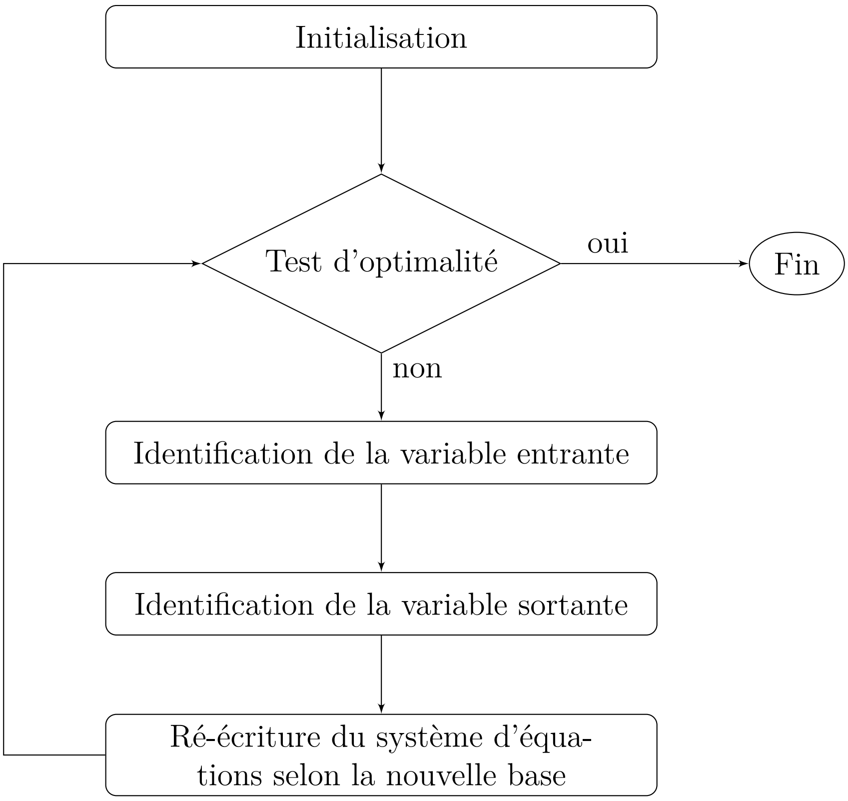


FIGURE 4.6 – Organigramme du Simplexe : résolution algébrique.

Algébriquement, le fonctionnement du Simplexe est basé sur l'organigramme donné à la figure 4.6. Ses différentes étapes sont expliquées ci-dessous :

1. **Initialisation :** considérer une base réalisable et la valeur de la fonction objectif qui lui correspond. Le système d'équations est écrit selon la base considérée ;
2. **Test d'optimalité :** les coefficients des variables hors base dans l'équation qui correspond à la fonction objectif sont tous positifs :
   — **oui :** optimalité atteinte ;
   — **non :** la valeur de la fonction objectif peut être améliorée en augmentant la valeur d'une des variables actuellement hors base (en la faisant entrer à la base tout en remplaçant une variable actuellement de base) ;
3. **Identification de la variable entrante :** celle qui est la plus susceptible de faire augmenter la fonction objectif rapidement, i.e. celle dont le coefficient négatif a la plus grande valeur absolue dans la dernière équation ;
4. **Identification de la variable sortante :** celle qui s'annule en premier à l'augmentation de la variable entrante. Elle est identifiée par le **test du ratio minimum** : pour chaque équation sauf la dernière, diviser le membre droit par le coefficient de la nouvelle variable entrante, si ce dernier est strictement positif. Le plus petit ratio indiquera la variable sortante ;
5. **Ré-écriture du système d'équations selon la nouvelle base :** ré-écrire le système d'équations en appliquant la méthode d'élimination de Gauss-Jordan.

Remarque

— Quand cela est possible, la base considérée pour l'étape d'initialisation est constituée des variables d'écart.

— Étant donné qu'à chaque itération du Simplexe, chaque équation comporte une seule variable de base avec un coefficient égal à 1 et d'autres variables qui sont hors base et donc nulles, il est facile d'identifier sa solution de base réalisable. Chaque variable de base aura comme valeur le membre droit de l'équation où elle apparait.

— Si à une certaine itération, le coefficient de la variable entrante au niveau de l'équation de chaque variable de base est négatif ou nul, alors le PL est non borné. Cela indique que la variable entrante, et par la même occasion z, peuvent être augmentés à l'infini sans risque de violation de contrainte. Le Simplexe s'arrête alors.

Exemple Reprenons notre problème précédent écrit sous forme canonique :

$$\begin{array}{rl} \text{maximiser} & z \\ \text{s.c.} & \left\{ \begin{array}{l} 20x_1 + 60x_2 + e_1 = 18000 \\ 20x_1 + 20x_2 + e_2 = 8000 \\ 40x_1 + 20x_2 + e_3 = 14000 \\ z - 20x_1 - 30x_2 = 0 \\ x_1, x_2, e_1, e_2, e_3 \geq 0 \end{array} \right. \end{array} \tag{4.15}$$

Initialisation : L'étape d'initialisation consiste à considérer la base (e_1, e_2, e_3). Ce qui donne :

$$\begin{cases} x_1 = 0, x_2 = 0 \\ \quad 20x_1 + 60x_2 + \boldsymbol{e_1} \qquad\qquad = 18000 \\ \quad 20x_1 + 20x_2 \qquad + \boldsymbol{e_2} \qquad = 8000 \\ \quad 40x_1 + 20x_2 \qquad\qquad + \boldsymbol{e_3} = 14000 \\ z - 20x_1 - 30x_2 \qquad\qquad\quad = 0 \end{cases}$$

La solution de base réalisable initiale est donc : $(0, 0, 18000, 8000, 14000)$ avec $z = 0$.

Test d'optimalité : Étant donné que les coefficients des deux variables hors base x_1 et x_2 sont strictement négatifs à la dernière équation du système, on en déduit que la solution de base réalisable actuelle avec $z = 0$ n'est pas optimale, i.e. z peut être augmenté en augmentant la valeur de x_1 ou bien celle de x_2.

ITÉRATION 1 :

Identification de la variable entrante : En se basant sur la dernière équation du système précédent, l'augmentation de la valeur de la variable hors base x_2 est plus susceptible de faire augmenter z rapidement par rapport à x_1, étant donné que $|-30| > |-20|$. x_2 est donc la nouvelle variable entrante vers la prochaine base.

Identification de la variable sortante : La variable sortante sera celle qui s'annulera en premier en augmentant la valeur de x_2 en considérant que x_1 est toujours égale à 0. En ne considérant plus pour le moment la dernière équation, le système précédent peut être ré-écrit comme suit :

$$\begin{cases} x_1 = 0 \\ 60x_2 + e_1 = 18000 \\ 20x_2 + e_2 = 8000 \\ 20x_2 + e_3 = 14000 \end{cases} \tag{4.16}$$

ou :

$$\begin{cases} x_1 = 0 \\ e_1 = 18000 - 60x_2 \\ e_2 = 8000 - 20x_2 \\ e_3 = 14000 - 20x_2 \end{cases} \tag{4.17}$$

Il s'agit à présent de vérifier à quel point on peut augmenter x_2 tout en respectant la contrainte de positivité sur toutes les variables de base :

$$\begin{cases} (e_1 = 18000 - 60x_2 \wedge e_1 \geq 0) \Rightarrow x_2 \leq \frac{18000}{60} = 300 \leftarrow \text{minimum} \\ (e_2 = 8000 - 20x_2 \wedge e_2 \geq 0) \Rightarrow x_2 \leq \frac{8000}{20} = 400 \\ (e_3 = 14000 - 20x_2 \wedge e_3 \geq 0) \Rightarrow x_2 \leq \frac{14000}{20} = 700 \end{cases} \tag{4.18}$$

Cette dernière vérification correspond au test du ratio minimum.

On constate que x_2 peut être augmenté jusqu'à 300 étant donné que cette valeur annule e_1. Au delà de cette valeur, la contrainte de positivité sur e_1 serait violée.

e_1 est identifiée comme variable sortante, i.e. elle sera hors base par rapport à la prochaine base.

Ré-écriture du système d'équations selon la nouvelle base : En utilisant l'ensemble des transformations algébriques nécessaires, le système d'équations est à présent ré-écrit en considérant la nouvelle base (x_2, e_2, e_3) :

$$\begin{cases} x_1 = 0, e_1 = 0 \\ \quad \frac{1}{3}x_1 + \boldsymbol{x_2} + \frac{1}{60}e_1 = 300 \\ \quad \frac{40}{3}x_1 - \frac{1}{3}e_1 + \boldsymbol{e_2} = 2000 \\ \quad \frac{100}{3}x_1 - \frac{1}{3}e_1 + \boldsymbol{e_3} = 8000 \\ z - 10x_1 + \frac{1}{2}e_1 = 9000 \end{cases}$$

La nouvelle solution de base réalisable est $(0, 300, 0, 2000, 8000)$ avec $z = 9000$.

Test d'optimalité : Le coefficient de la variable hors base x_1 est strictement négatif (-10) à la dernière équation du système ce qui signifie que l'optimum n'est toujours pas atteint.

ITÉRATION 2 :

Identification de la variable entrante : x_1 est la seule variable hors base dont l'augmentation ferait augmenter z. C'est la nouvelle variable entrante.

Identification de la variable sortante : On applique le test du ratio minimum pour identifier la nouvelle variable entrante :

— Pour x_2, le ratio = 300 / (1/3) = 900
— Pour e_2, le ratio = 2000/(40/3) = 150 $\leftarrow$ minimum
— Pour e_3, le ratio = 8000 / (100/3) = 240

e_2 est la nouvelle variable sortante de la base.

Ré-écriture du système d'équations selon la nouvelle base : En considérant la nouvelle base (x_2, x_1, e_3) le système d'équations est ré-écrit comme suit :

$$\begin{cases} e_1 = 0, e_2 = 0 \\ \quad \boldsymbol{x_2} + \frac{1}{40}e_1 - \frac{1}{40}e_2 = 250 \\ \quad \boldsymbol{x_1} - \frac{1}{40}e_1 + \frac{3}{40}e_2 = 150 \\ \quad \frac{1}{2}e_1 - \frac{5}{2}e_2 + \boldsymbol{e_3} = 3000 \\ z + \frac{1}{40}e_1 + \frac{3}{40}e_2 = 10500 \end{cases} \tag{4.19}$$

Ce qui donne la nouvelle solution de base réalisable $(150, 250, 0, 0, 3000)$ avec $z = 10500$.

Test d'optimalité : Tous les coefficients des variables hors base au niveau de la dernière équation du système sont positifs. La solution de base réalisable $(150, 250, 0, 0, 3000)$ avec $z^* = 10500$ est donc optimale. Elle correspond à la solution optimale $(150, 250)$ par rapport au problème originale. On remarque qu'elle est identique à la solution optimale trouvée par la méthode de résolution graphique.

4.5.3 Simplexe : résolution à l'aide du tableau

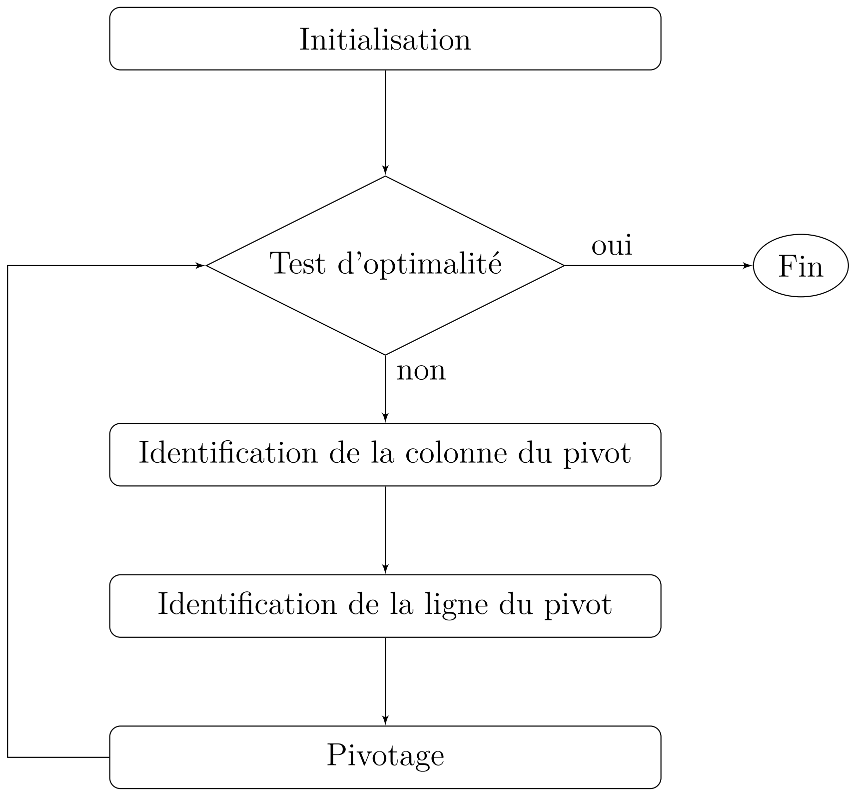


FIGURE 4.7 – Organigramme du Simplexe : résolution à l'aide du tableau.

La méthode du Simplexe par le tableau suit le principe de la résolution algébrique en utilisant un formalisme de tableaux. Elle peut être synthétisée par l'organigramme donné à la figure 4.7. Chacune de ses étapes est détaillée ci-dessous :

1. **Initialisation :** construction du 1^{er} tableau en considérant la forme canonique du PL selon une première base réalisable, comme suit :
   - La première ligne du tableau contient z et toutes les variables du problème en forme canonique, c.à.d : les variables de décision et les variables d'écart.
   - La suite du tableau est divisée comme suit :
     - La colonne de gauche contient les variables de base du premier tableau. Le z est placé au niveau de la dernière cellule.
     - Les colonnes centrales contiennent les coefficients des variables dans les équations de la forme canonique.

— La colonne de droite contient les membres droits des équations de la forme canonique.

	z	x_1	x_2	$\dots$	x_n	e_1	e_2	$\dots$	e_m	
e_1	0	a_{11}	a_{12}	$\dots$	a_{1n}	1	0	$\dots$	0	b_1
e_2	0	a_{21}	a_{22}	$\dots$	a_{2n}	0	1	$\dots$	0	b_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
e_m	0	a_{m1}	a_{m2}	$\dots$	a_{mn}	0	0	$\dots$	1	b_m
z	1	$-c_1$	$-c_2$	$\dots$	$-c_n$	0	0	$\dots$	0	0

2. **Test d'optimalité :** la solution de base réalisable du tableau actuel est optimale si et seulement si chaque élément à la dernière ligne du tableau est positif (sans prendre en compte la valeur de droite). En cas de non optimalité, la méthode continue en construisant un nouveau tableau avec sa nouvelle base.
3. **Identification de la colonne du pivot :** consiste à identifier la nouvelle variable entrante : celle qui a l'élément négatif le plus grand en valeur absolue à la dernière ligne du tableau actuel. Sa colonne est appelée **colonne du pivot**.
4. **Identification de la ligne du pivot :** consiste à identifier la nouvelle variable sortante par le test du ratio minimum : pour chaque ligne où l'élément de la colonne du pivot est strictement positif, diviser l'élément de la colonne de droite par l'élément de la colonne du pivot ; la ligne avec le plus petit ratio est la **ligne du pivot**. La variable de base de cette ligne est la variable sortante de la base.
5. **Pivotage :** le **pivot** est l'intersection entre la colonne et la ligne du pivot.

   Le nouveau tableau est obtenu en appliquant la méthode de l'élimination de Gauss :

   — diviser la ligne du pivot par le pivot ;

   — de chaque autre ligne comprenant l'élément e dans la colonne du pivot, soustraire le produit de e et de la nouvelle ligne du pivot.

Remarques

— Le 1[er] tableau du simplexe peut être synthétisé en considérant l'écriture matricielle comme suit :

	z	$\mathbf{x}^\mathsf{T}$	$\mathbf{e}^\mathsf{T}$	
$\mathbf{e}$	0	$\mathbf{A}$	$\mathbf{I}$	$\mathbf{b}$
z	1	$-\mathbf{c}$	0	0

— Pour chaque tableau, la valeur de chaque variable hors base est égale à 0 et la valeur de chaque variable de base est indiquée par la valeur de l'élément de la colonne de droite sur la même ligne du tableau.

— La valeur de la fonction objectif (z) est également indiquée par la valeur de l'élément de la colonne de droite sur la dernière ligne du tableau.

— La non existence d'un élément de la colonne du pivot strictement positif indiquerait que le PL est non borné. L'algorithme du Simplexe s'arrête alors.

Exemple Reprenons la forme canonique de notre précédent PL :

$$\text{maximiser} \quad z$$
$$\text{s.c.} \quad \begin{cases} 20x_1 + 60x_2 + e_1 = 18000 \\ 20x_1 + 20x_2 + e_2 = 8000 \\ 40x_1 + 20x_2 + e_3 = 14000 \\ z - 20x_1 - 30x_2 = 0 \\ x_1, x_2, e_1, e_2, e_3 \geq 0 \end{cases} \tag{4.20}$$

Initialisation : Le 1[er] tableau du Simplexe donne alors :

	z	x_1	x_2	e_1	e_2	e_3	
e_1	0	20	60	1	0	0	18000
e_2	0	20	20	0	1	0	8000
e_3	0	40	20	0	0	1	14000
z	1	-20	-30	0	0	0	0

Pour ce tableau :

- La base est (e_1, e_2, e_3). Les variables de base sont donc e_1, e_2 et e_3 ; et les variables hors base sont x_1 et x_2.
- La solution de base réalisable est $(x_1, x_2, e_1, e_2, e_3) = (0, 0, 18000, 8000, 14000)$.
- La valeur de la fonction objectif est égale à 0.

Test d'optimalité : Au niveau de la dernière ligne du tableau, il existe des éléments négatifs, ce qui indique que la solution de base réalisable actuelle n'est pas optimale. Il faut alors construire un nouveau tableau avec une nouvelle base.

ITÉRATION 1 :

Identification de la colonne du pivot : La variable entrante à la base est celle qui a un élément négatif et qui est le plus grand en valeur absolue au niveau de la dernière ligne du tableau. Il s'agit ici donc de la variable x_2. Sa colonne est la colonne du pivot.

	z	x_1	$\downarrow$ x_2	e_1	e_2	e_3	
e_1	0	20	60	1	0	0	18000
e_2	0	20	20	0	1	0	8000
e_3	0	40	20	0	0	1	14000
z	1	-20	-30	0	0	0	0

Identification de la ligne du pivot : Afin d'identifier la variable sortante de la base, on effectue le test du ratio minimum : (élément de la colonne de droite / élément de la colonne du pivot) pour les lignes où l'élément de la colonne du pivot est strictement positif ; la ligne avec le plus petit ratio correspondra à la ligne de la variable sortante, c'est à dire la ligne du pivot. Il s'agit ici de la ligne de la variable e_1.

	z	x_1	x_2 ↓	e_1	e_2	e_3			
e_1	0	20	$\boxed{60}$	1	0	0	18000	18000/60=300	$\rightarrow$
e_2	0	20	20	0	1	0	8000	8000/20=400	
e_3	0	40	20	0	0	1	14000	14000/20=700	
z	1	-20	-30	0	0	0	0		

Pivotage : Nous obtenons alors un nouveau tableau avec la nouvelle base (x_2, e_2, e_3).

	z	x_1	x_2	e_1	e_2	e_3	
x_2							
e_2							
e_3							
z							

Pour son remplissage, la méthode d'élimination de Gauss est appliquée.

Tout d'abord, la ligne du pivot, i.e. la ligne de la nouvelle variable de base x_2, est divisé par le pivot, i.e. par la valeur 60.

Ensuite :

— Pour la ligne de e_2, l'élément de la colonne du pivot est égal à 20. La nouvelle ligne est obtenue en soustrayant le produit de la ligne de x_2 dans le nouveau tableau et la valeur 20, de la ligne de e_2 dans le précédent tableau.

— Pour la ligne de e_3, l'élément de la colonne du pivot est égal à 20. La nouvelle ligne est obtenue en soustrayant le produit de la ligne de x_2 dans le nouveau tableau et la valeur 20, de la ligne de e_3 dans le précédent tableau.

— Pour la ligne de z, l'élément de la colonne du pivot est égal à -30. La nouvelle ligne est obtenue en soustrayant le produit de la ligne de x_2 dans le nouveau tableau et la valeur -30, de la ligne de z dans le précédent tableau.

	z	x_1	x_2	e_1	e_2	e_3	
x_2	0	1/3	1	1/60	0	0	300
e_2	0	40/3	0	-1/3	1	0	2000
e_3	0	100/3	0	-1/3	0	1	8000
z	1	-10	0	1/2	0	0	9000

Pour ce nouveau tableau :

— La base est : (x_2, e_2, e_3). Les variables de base sont donc x_2, e_2 et e_3 ; et les variables hors base sont x_1 et e_1.

— La solution de base réalisable est $(x_1, x_2, e_1, e_2, e_3) = (0, 300, 0, 2000, 8000)$.

— La valeur de la fonction objectif est égale à 9000.

Test d'optimalité : Ce tableau n'est toujours pas optimal étant donné qu'il y au moins une variable hors base avec un coefficient strictement négatif à la dernière ligne.

ITÉRATION 2 :

De la même manière que précédemment, il y a identification des nouvelles colonne et ligne du pivot, puis calcul du nouveau tableau.

	z	x_1 ↓	x_2	e_1	e_2	e_3			
x_2	0	1/3	1	1/60	0	0	300	300/(1/3)=900	
e_2	0	$\boxed{40/3}$	0	1/3	1	0	2000	2000/(40/3)=150	→
e_3	0	100/3	0	1/3	0	1	8000	8000/(100/3)=240	
z	1	-10	0	1/2	0	0	9000		

x_1 est la nouvelle variable entrante, e_2 la nouvelle variable sortante, et nous obtenons le nouveau tableau :

	z	x_1	x_2	e_1	e_2	e_3	
x_2	0	0	1	1/40	-1/40	0	250
x_1	0	1	0	-1/40	3/40	0	150
e_3	0	0	0	1/2	-5/2	1	3000
z	1	0	0	1/4	3/4	0	10500

Dans ce nouveau tableau, il n'y a plus de coefficient strictement négatif au niveau de la dernière ligne. Ce tableau est donc optimal :

— Sa solution de base réalisable $(x_1, x_2, e_1, e_2, e_3) = (150, 250, 0, 0, 3000)$ est une solution de base optimale.

— La valeur de la fonction objectif qui est égale à 10500 est optimale.

— La solution qui correspond au problème original est $(x_1, x_2) = (150, 250)$.

Nous obtenons bien évidemment la même solution optimale que précédemment avec la résolution graphique et la résolution algébrique du Simplexe.

4.6 Dualité

Chaque problème linéaire est associé à un autre problème linéaire dit **dual**. Le premier problème est alors dit problème **primal**.

4.6.1 Dual d'un PL en forme standard

Soit un problème linéaire écrit sous forme standard :

$$\text{maximiser} \quad z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$
$$\text{s.c.} \quad \begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n & \leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n & \leq b_2 \\ & \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n & \leq b_m \\ x_1, x_2, \ldots x_n \geq 0 \end{cases}$$

Son problème dual sera alors :

$$\text{minimiser} \quad w = b_1y_1 + b_2y_2 + \ldots + b_my_m$$
$$\text{s.c.} \quad \begin{cases} a_{11}y_1 + a_{21}y_2 + \ldots + a_{m1}y_m & \geq c_1 \\ a_{12}y_1 + a_{22}y_2 + \ldots + a_{m2}y_m & \geq c_2 \\ & \vdots \\ a_{1n}y_1 + a_{2n}y_2 + \ldots + a_{mn}y_m & \geq c_n \\ y_1, y_2, \ldots y_m \geq 0 \end{cases}$$

Remarque Le dual du dual est le problème primal.

Exemple Pour rappel, la formulation en programme linéaire du problème précédent de l'artisan chocolatier était la suivante :

$$\text{maximiser} \quad z = 20x_1 + 30x_2$$
$$\text{s.c.} \quad \begin{cases} 20x_1 + 60x_2 \leq 18000 \\ 20x_1 + 20x_2 \leq 8000 \\ 40x_1 + 20x_2 \leq 14000 \\ x_1, x_2 \geq 0 \end{cases}$$

Son problème dual est alors :

$$\text{minimiser} \quad w = 18000y_1 + 8000y_2 + 14000y_3$$
$$\text{s.c.} \quad \begin{cases} 20y_1 + 20y_2 + 40y_3 \geq 20 \\ 60y_1 + 20y_2 + 20y_3 \geq 30 \\ y_1, y_2, y_3 \geq 0 \end{cases} \tag{4.21}$$

Ainsi :
- Le primal possède 2 variables de décision => le dual possède 2 contraintes.
- Le primal possède 3 contraintes => le dual possède 3 variables de décision.

Interprétation économique Le problème dual ci-dessus peut être interprété comme un nouveau problème où nous avons un 2e artisan chocolatier qui cherche à racheter tous les ingrédients disponibles dans les réserves du 1er artisan à coût minimum (fonction objectif du dual) et où le 1er artisan aurait un prix de vente au moins supérieur au bénéfice qu'il ferait s'il fabriquait et vendait lui-même les tablettes de chocolat (contraintes fonctionnelles du dual).

Les variables duales auraient comme interprétation les prix d'achat par le 2e artisan comme suit :

— y_1 : prix d'achat d'une unité de cacao (unité=1g).

— y_2 : prix d'achat d'une unité de noisettes (unité=1g).

— y_3 : prix d'achat d'une unité de lait (unité=1g).

La fonction objectif w serait le coût total d'achat des ingrédients par le 2e artisan.

4.6.2 Écriture matricielle du dual d'un PL en forme standard

Pour un PL en forme standard à n variables de décision et m contraintes :

$$\begin{aligned} \text{maximiser} \quad & z = \mathbf{c}^\intercal \mathbf{x} \\ \text{s.c.} \quad & \mathbf{A}\mathbf{x} \leq \mathbf{b} \text{ et } \mathbf{x} \geq 0 \end{aligned}$$

l'écriture matricielle de son dual est la suivante :

$$\begin{aligned} \text{minimiser} \quad & w = \mathbf{b}^\intercal \mathbf{y} \\ \text{s.c.} \quad & \mathbf{A}^\intercal \mathbf{y} \geq \mathbf{c} \text{ et } \mathbf{y} \geq 0 \end{aligned}$$

avec $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$

Exemple Pour notre précédent problème sous forme standard en écriture matricielle :

$$\begin{aligned} \text{maximiser} \quad & z = \begin{bmatrix} 20 & 30 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \text{s.c.} \quad & \begin{bmatrix} 20 & 60 \\ 20 & 20 \\ 40 & 20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} 18000 \\ 8000 \\ 14000 \end{bmatrix} \text{ et } \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \end{aligned}$$

le dual en forme matricielle est

$$\begin{aligned} \text{minimiser} \quad & z = \begin{bmatrix} 18000 & 8000 & 14000 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \\ \text{s.c.} \quad & \begin{bmatrix} 20 & 20 & 40 \\ 60 & 20 & 20 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \geq \begin{bmatrix} 20 \\ 30 \end{bmatrix} \text{ et } \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \geq \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \end{aligned}$$

4.6.3 Dual d'un PL sous forme générale

Pour un PL sous forme générale, i.e. non nécessairement écrit en forme standard, considéré comme primal, sa relation avec son dual est donnée à la table 4.1.

TABLEAU 4.1 – Primal vs. Dual

primal	dual
dual	primal
maximiser	minimiser
contrainte $\leq$	variable ≥ 0
contrainte $=$	variable quelconque
contrainte $\geq$	variable ≤ 0
variable ≥ 0	contrainte $\geq$
variable quelconque	contrainte $=$
variable ≤ 0	contrainte $\leq$

Exemple Soit un PL donné comme suit :

$$\begin{array}{ll} \text{minimiser} & z = 15x_1 - 5x_3 \\ \text{s.c.} & \left\{ \begin{array}{l} 40x_1 + 40x_3 \geq 20 \\ x_1 + x_2 + x_3 = 5 \\ x_1 - x_3 \leq 0 \\ x_2 + x_3 = 12 \\ x_1 \geq 0, x_2 \leq 0, x_3 \text{ quelconque} \end{array} \right. \end{array} \tag{4.22}$$

Si on considère le PL ci-dessus comme primal, alors son dual sera :

$$\begin{array}{ll} \text{maximiser} & w = 20y_1 + 5y_2 + 12y_4 \\ \text{s.c.} & \left\{ \begin{array}{l} 40y_1 + y_2 + y_3 \leq 15 \\ y_2 + y_4 \geq 0 \\ 40y_1 + y_2 - y_3 + y_4 = -5 \\ y_1 \geq 0, y_2 \text{ quelconque}, y_3 \leq 0, y_4 \text{ quelconque} \end{array} \right. \end{array} \tag{4.23}$$

4.6.4 Théorème faible de la dualité

Pour toute solution réalisable $\mathbf{x}$ d'un problème primal en forme standard et toute solution réalisable $\mathbf{y}$ de son problème dual, nous avons :

$$z = \mathbf{c}^\intercal \mathbf{x} \leq \mathbf{b}^\intercal \mathbf{y} = w$$

Exemple $(x_1, x_2) = (100, 100)$ est une solution réalisable de notre problème primal précédent de l'artisan chocolatier, étant donné qu'elle satisfait toutes ses contraintes. Cette solution produit un $z = 20 \times 100 + 30 \times 100 = 5000$.

$(y_1, y_2, y_3) = (1, 1, 1)$ est une solution réalisable de son dual étant donné qu'elle satisfait toutes ses contraintes. Cette solution produit un $w = 18000 \times 1 + 8000 \times 1 + 14000 \times 1 = 40000$.

Le théorème est bien vérifié sur cet exemple. Nous avons $5000 \leq 40000$.

4.6.5 Théorème fort de la dualité

Pour toute solution optimale $\mathbf{x}^*$ d'un problème primal en forme standard et toute solution optimale $\mathbf{y}^*$ de son problème dual, nous avons :

$$z^* = \mathbf{c}^\intercal \mathbf{x}^* = \mathbf{b}^\intercal \mathbf{y}^* = w^*$$

Exemple $(x_1, x_2) = (150, 250)$ est la solution optimale de notre problème primal et elle produit un $z^* = 20 \times 150 + 30 \times 250 = 10500$.

La résolution du problème dual donnerait $(y_1, y_2, y_3) = (1/4, 3/4, 0)$ comme solution optimale. Celle-ci produit un $w^* = 18000 \times 1/4 + 8000 \times 3/4 + 14000 \times 0 = 10500$.

Le théorème est bien vérifié sur cet exemple.

4.6.6 Théorème des solutions complémentaires

À chaque itération du Simplexe, correspond une solution réalisable $\mathbf{x}$ du problème primal en forme standard et une solution complémentaire $\mathbf{y}$ du problème dual. Les deux solutions produisent la même valeur pour les deux fonctions objectif ($z = w$).

La valeur de chaque variable duale y_i de la solution complémentaire est donnée à la dernière ligne du tableau à l'intersection avec la colonne de la variable d'écart e_i.

Si $\mathbf{x}$ n'est pas optimale pour le primal alors $\mathbf{y}$ est non réalisable pour le dual.

Exemple Prenons un des tableaux non optimaux de la résolution de notre problème primal précédent par le Simplexe :

	z	x_1	x_2	e_1	e_2	e_3	
x_2	0	1/3	1	1/60	0	0	300
e_2	0	40/3	0	-1/3	1	0	2000
e_3	0	100/3	0	-1/3	0	1	8000
z	1	-10	0	1/2	0	0	9000

La solution non réalisable complémentaire du problème dual pouvant être déduite à partir de ce tableau est $(y_1, y_2, y_3) = (1/2, 0, 0)$ avec $w = 9000$.

4.6.7 Théorème des solutions optimales complémentaires

À la dernière itération du Simplexe, la solution optimale $\mathbf{x}^*$ du problème primal en forme standard et sa solution complémentaire optimale $\mathbf{y}^*$ du problème dual peuvent être identifiées simultanément. Les deux solutions produisent la même valeur de fonctions objectif optimale ($z^* = w^*$).

La valeur de chaque variable duale y_i est donnée à la dernière ligne du tableau à l'intersection avec la colonne de la variable d'écart e_i.

Exemple Pour rappel, le dernier tableau de la résolution de notre problème primal précédent par le Simplexe était le suivant :

	z	x_1	x_2	e_1	e_2	e_3	
x_2	0	0	1	1/40	-1/40	0	250
x_1	0	1	0	-1/40	3/40	0	150
e_3	0	0	0	1/2	-5/2	1	3000
z	1	0	0	1/4	3/4	0	10500

La solution optimale du problème dual peut être déduite à partir du tableau optimal du primal sans avoir à résoudre le problème dual en lui même. La solution duale optimale est $(y_1, y_2, y_3) = (1/4, 3/4, 0)$ avec $w^* = 10500$.

4.7 Exercices

Exercice 33

1. Pour quel type de programmes linéaires est-il possible d'appliquer une résolution graphique ?

2. Soit un programme linéaire en forme standard à n variables de décision et m contraintes. Lors de la résolution de ce programme par le Simplexe, une base contiendra combien de variables ?

3. Quel est le test d'optimalité d'un tableau du Simplexe ?

4. Qu'est-ce qu'une solution non réalisable pour un programme linéaire ?

Exercice 34

Un fabricant produit 3 types de yaourts aux fruits A, B et C, à partir de fruits, de lait et de sucre. Chaque pot de yaourt doit respecter les quantités suivantes de matières premières (en grammes).

	A	B	C
Fruit	20	30	30
Lait	80	70	70
Sucre	0	5	0

Le fabricant dispose de 60 kg de fruits, de 150 kg de lait, et de 30 kg de sucre.

La vente d'un pot de chacun des types de yaourt A, B et C, rapporte respectivement 3 DA, 5 DA et 4 DA de profit.

Le fabricant cherche à maximiser son profit total.

Modéliser ce problème par un programme linéaire.

Exercice 35

Un fabricant de raquettes de tennis fait un bénéfice de 800 DA sur chaque raquette ordinaire et de 1500 DA sur chaque grande raquette. Pour satisfaire la demande des vendeurs, la production journalière de raquettes ordinaires devrait se situer entre 30 et 80, et la production journalière de grandes raquettes entre 10 et 30. Pour maintenir une bonne qualité, le nombre de raquettes produites ne devrait pas dépasser 80 par jour. La fabricant souhaite optimiser sa production afin d'obtenir le meilleur bénéfice total possible.

Modéliser le problème sous forme de programme linéaire.

Exercice 36

Soit le PL suivant :

$$\begin{array}{rl} \text{maximiser} & z = x_1 + x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} x_1 + x_2 \geq 2 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

1. Combien y-t-il de variables de décision dans ce PL ?
2. Quelles sont ces variables de décision ?
3. Quelle est la fonction objectif de ce PL ?
4. Quelles sont les contraintes de ce PL ?
5. Résoudre ce PL graphiquement.

Exercice 37

Résoudre graphiquement les deux programmes linéaires PL_1 et PL_2 ci-dessous :

$$\text{maximiser} \quad z = 2x_1 + 10x_2$$
$$\text{s.c.} \quad \left\{ \begin{array}{l} 2x_1 + x_2 \leq 4 \\ 3x_1 + 2x_2 \geq 12 \\ x_1 \geq 6 \\ x_1, x_2 \geq 0 \end{array} \right. \quad (PL_1)$$

$$\text{maximiser} \quad z = x_1 - x_2$$
$$\text{s.c.} \quad \left\{ \begin{array}{l} -x_1 + 2x_2 \geq 4 \\ 2x_1 + x_2 \geq 6 \\ x_2 \geq 0 \end{array} \right. \quad (PL_2)$$

Exercice 38

Le tableau suivant représente un des tableaux de la résolution d'un programme linéaire en utilisant la méthode du simplexe.

	z	x_1	x_2	x_3	e_1	e_2	
e_2	0	0	1	2	2	1	4
x_1	0	1	2	1	2	0	10
z	1	0	-1	-2	-4	0	11

1. Quelles sont les variables de base et quelles sont les variables hors base correspondant à ce tableau ?

2. Quelle est la solution de base réalisable correspondant à ce tableau ?

3. Donner le tableau suivant de la résolution du programme linéaire ?

4. Est-ce que la solution de base réalisable correspondant à ce nouveau tableau est optimale ? Pourquoi ?

Exercice 39

Soit le programme linéaire suivant :

$$\text{minimiser} \quad z = -x_1 - 2x_2$$
$$\text{s.c.} \quad \left\{ \begin{array}{l} -x_1 + x_2 \leq 2 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{array} \right.$$

1. Résoudre graphiquement le PL.

2. Donner la forme standard du PL.

3. Résoudre le PL par l'algorithme du Simplexe.

Exercice 40

Soit le PL donné ci-dessous :

$$\begin{array}{rl} \text{maximiser} & z = 2x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} x_1 + x_2 \leq 10 \\ x_1 \leq 7 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

1. Résoudre le PL par la méthode graphique.

2. La résolution du PL par la méthode du Simplexe nécessite l'utilisation de combien de variables d'écart ?

3. Lister toutes les solutions de base réalisables du PL.

4. Résoudre le PL par la méthode du Simplexe.

5. Quelles sont les solutions de base réalisables non parcourues par le Simplexe.

6. Donner la contrainte dont la suppression rendrait le PL non borné.

7. Donner une contrainte dont l'ajout rendrait le PL non réalisable.

Exercice 41

Résoudre le programme linéaire suivant par la méthode du Simplexe.

$$\begin{array}{rl} \text{maximiser} & z = x_1 + x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} x_1 - x_2 \leq 3 \\ x_1 - 2x_2 \leq 3 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

4.8 Solutions des exercices

Solution de l'exercice 33

1. Les programmes linéaires pour lesquelles il est possible d'appliquer une résolution graphique sont ceux où le nombre de variables de décision est inférieur ou égal à 2 (voir 3).

2. La base d'un programme linéaire à n variables de décision et m contraintes contient m variables.

3. Un tableau du Simplexe est optimale si tous les coefficients de la dernière ligne du tableau sont positifs ou nuls.

4. Une solution non réalisable pour un programme linéaire est un ensemble de valeurs affectées aux variables de décision du programme (une valeur par variable de décision), qui ne respecte pas au moins une des contraintes du programme.

Solution de l'exercice 34

Afin de modéliser le problème par un programme linéaire, on considère les variables de décision suivantes :

— x_A : nombre de pots de yaourt de type A à produire ;
— x_B : nombre de pots de yaourt de type B à produire ;
— x_C : nombre de pots de yaourt de type C à produire.

Le programme linéaire est donné comme suit :

$$\text{maximiser} \quad z = 3x_A + 5x_B + 4x_C$$

$$\text{s.c.} \quad \begin{cases} 20x_A + 30x_B + 30x_C \leq 60000 \\ 80x_A + 70x_B + 70x_C \leq 150000 \\ 5x_B \leq 30000 \\ x_A, x_B, x_C \geq 0 \end{cases}$$

Solution de l'exercice 35

Afin de modéliser le problème par un programme linéaire, on considère les variables de décision suivantes :

— x_1 : nombre de raquettes ordinaires à produire par jour,
— x_2 : nombre de grandes raquettes à produire par jour,

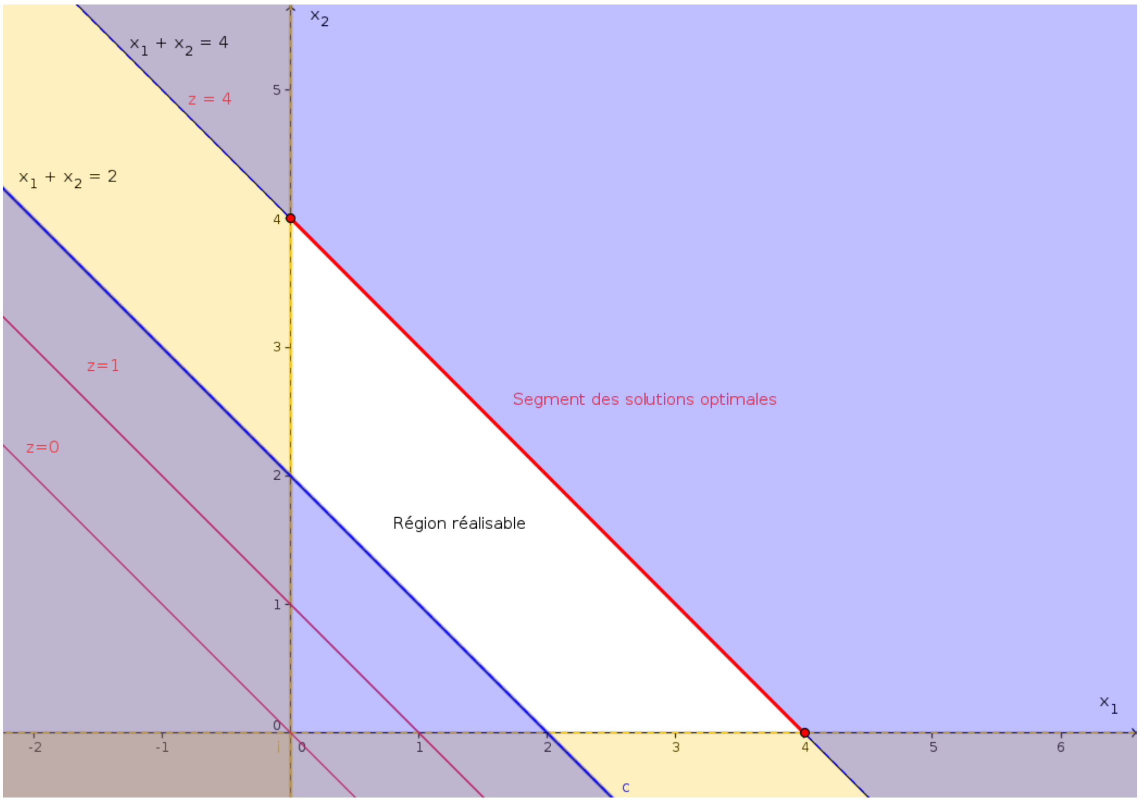
Le programme linéaire est donné comme suit :

$$\text{maximiser} \quad z = 800x_1 + 1500x_2$$

$$\text{s.c.} \quad \begin{cases} x_1 \geq 30 \\ x_1 \leq 80 \\ x_2 \geq 10 \\ x_2 \leq 30 \\ x_1 + x_2 \leq 80 \\ x_1, x_2 \geq 0 \end{cases}$$

Solution de l'exercice 36

1. Le PL contient 2 variables de décision.
2. Les variables de décision sont : x_1 et x_2.
3. La fonction objectif du PL est : $z = x_1 + x_2$.
4. Les contraintes du PL sont : $x_1 + x_2 \geq 2, x_1 + x_2 \leq 4$ et $x_1, x_2 \geq 0$.
5. Résolution graphique du PL :

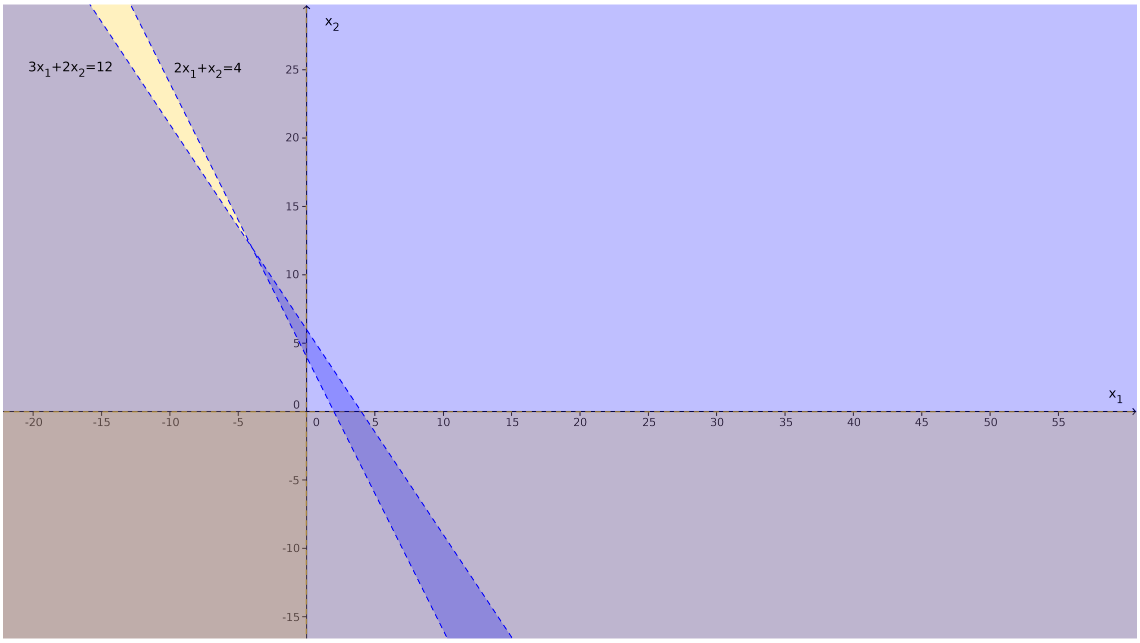


Les solutions optimales du PL sont toutes celles qui se trouvent au niveau du segment compris entre les deux points $(4, 0)$ et $(0, 4)$, ce qui génère une valeur de fonction objectif optimale égale à 4.

Solution de l'exercice 37

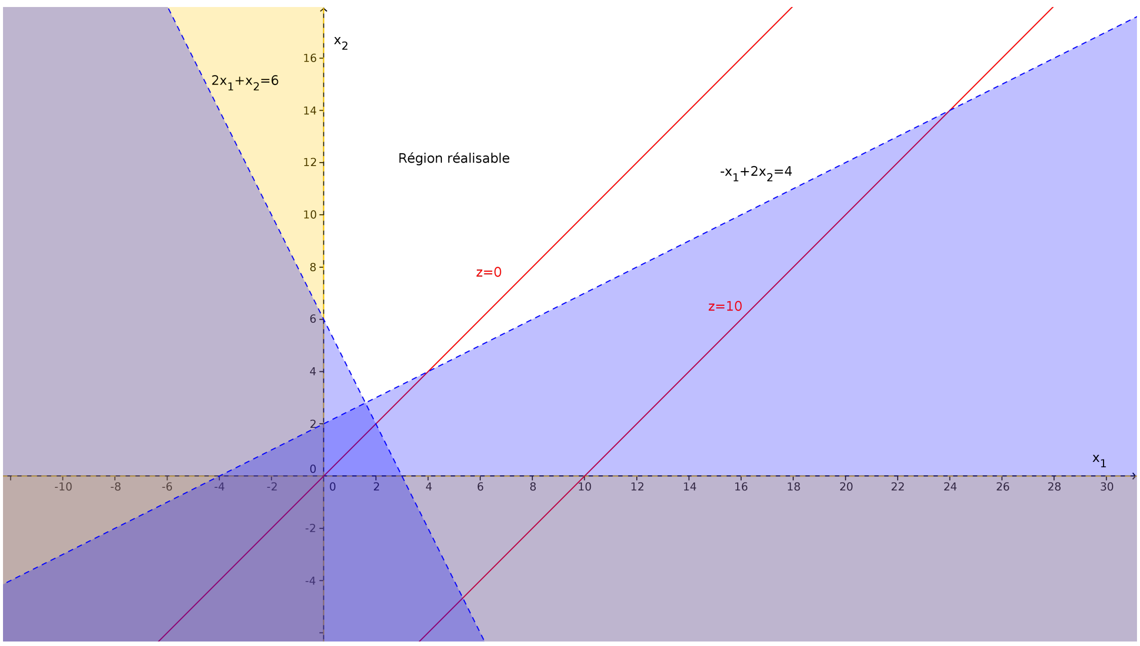
$$\text{maximiser} \quad z = 2x_1 + 10x_2$$

$$\text{s.c.} \quad \left\{ \begin{array}{l} 2x_1 + x_2 \leq 4 \\ 3x_1 + 2x_2 \geq 12 \\ x_1 \geq 6 \\ x_1, x_2 \geq 0 \end{array} \right. \quad (PL_1)$$



Aucune solution dans le plan n'est réalisable. La région réalisable est vide. Ce problème est donc non réalisable.

$$
\begin{array}{ll}
\text{maximiser} & z = x_1 - x_2 \\
\text{s.c.} & \left\{ \begin{array}{l} -x_1 + 2x_2 \geq 4 \\ 2x_1 + x_2 \geq 6 \\ x_2 \geq 0 \end{array} \right. \quad (PL_2)
\end{array}
$$



Ce problème est non borné.

Solution de l'exercice 38

1. Les variables de base pour le tableau donné sont e_2 et x_1. Les variables hors base sont x_2, x_3 et e_1.

2. La solution de base réalisable de ce tableau est $(x_1, x_2, x_3, e_1, e_2) = (10, 0, 0, 0, 4)$.

3. Le tableau suivant de la résolution du programme linéaire est le suivant :

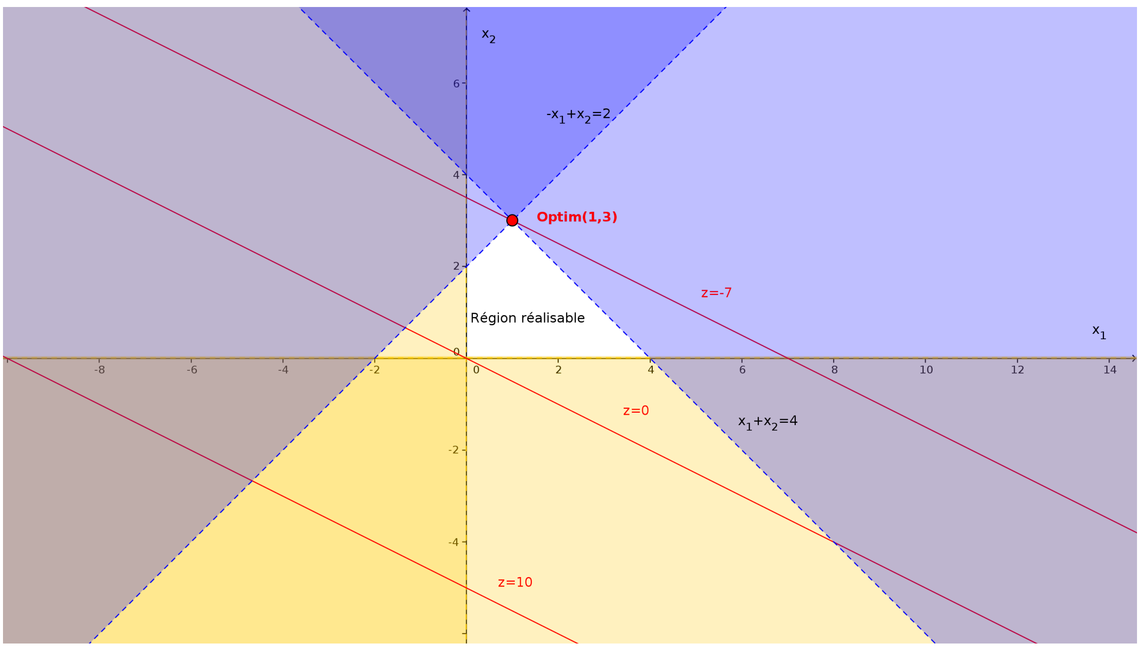
	z	x_1	x_2	x_3	e_1	e_2	
e_1	0	0	$\frac{1}{2}$	1	1	$\frac{1}{2}$	2
x_1	0	1	1	-1	0	-1	6
z	1	0	1	2	0	2	19

4. La solution de base réalisable correspondant à ce nouveau tableau est optimale car tous les éléments centraux de la dernière ligne du tableau sont positifs.

Solution de l'exercice 39

$$\begin{array}{rl} \text{minimiser} & z = -x_1 - 2x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} -x_1 + x_2 \leq 2 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array} \qquad \text{(PL)}$$

1. Résolution graphique du PL :



Solution optimale à $(x_1, x_2) = (1, 3)$ avec $z = -7$.

2. Le PL en forme standard :

$$\begin{array}{ll} \text{maximiser} & y = -z = x_1 + 2x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} -x_1 + x_2 \leq 2 \\ x_1 + x_2 \leq 4 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

3. Résolution du PL par le Simplexe.

Intégration des variables d'écart e_1 et e_2 et forme canonique :

$$\begin{array}{ll} \text{maximiser} & y \\ \text{s.c.} & \left\{ \begin{array}{l} -x_1 + x_2 + e_1 = 2 \\ x_1 + x_2 + e_2 = 4 \\ y - x_1 - 2x_2 = 0 \\ x_1, x_2, e_1, e_2 \geq 0 \end{array} \right. \end{array}$$

	y	x_1	x_2	e_1	e_2	
e_1	0	-1	$\boxed{1}$	1	0	2
e_2	0	1	1	0	1	4
y	1	-1	-2	0	0	0

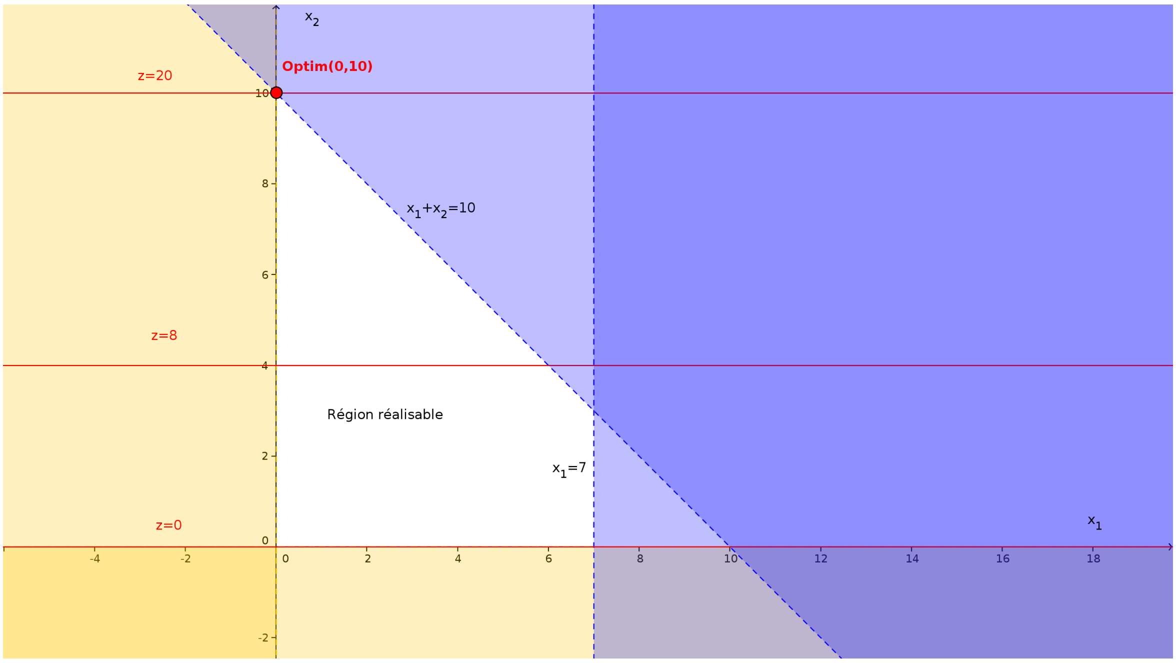
	y	x_1	x_2	e_1	e_2	
x_2	0	-1	1	1	0	2
e_2	0	$\boxed{2}$	0	-1	1	2
y	1	-3	0	2	0	4

	y	x_1	x_2	e_1	e_2	
x_2	0	0	1	$\frac{1}{2}$	$\frac{1}{2}$	3
x_1	0	1	0	$\frac{-1}{2}$	$\frac{1}{2}$	1
y	1	0	0	$\frac{1}{2}$	$\frac{3}{2}$	7

Tableau optimal avec solution de base réalisable $(x_1, x_2, e_1, e_2) = (1, 3, 0, 0)$ et $y = 7$. La solution optimal du PL est donc $(x_1, x_2) = (1, 3)$ avec $z = -y = -7$.

Solution de l'exercice 40

1. Résolution graphique du PL (voir figure ci-après).



Solution optimale $(x_1, x_2) = (0, 10)$ avec $z = 20$.

2. La résolution du PL par la méthode du Simplexe nécessite l'utilisation de 2 variables d'écart : une pour chaque contrainte.

3. En admettant que les deux variables d'écart e_1 et e_2 sont introduites au PL comme suit :

$$\text{maximiser} \quad z$$
$$\text{s.c.} \quad \begin{cases} x_1 + x_2 + e_1 = 10 \\ x_1 + e_2 = 7 \\ z - 2x_2 = 0 \\ x_1, x_2, e_1, e_2 \geq 0 \end{cases}$$

Les solutions de base réalisables peuvent être obtenues à partir du graphe : elles sont équivalentes aux 4 sommets réalisables. Les solutions de base réalisables correspondant à (x_1, x_2, e_1, e_2) sont :

— (0,0,10,7)

— (0,10,0,7)

— (7,3,0,0)

— (7,0,3,0)

4.

	z	x_1	x_2	e_1	e_2	
e_1	0	1	$\boxed{1}$	1	0	10
e_2	0	1	0	0	1	7
z	1	0	-2	0	0	0

	z	x_1	x_2	e_1	e_2	
x_2	0	1	1	1	0	10
e_2	0	1	0	0	1	7
z	1	2	0	2	0	20

Solution de base réalisable optimale $(x_1, x_2, e_1, e_2) = (0, 10, 0, 7)$ et donc solution optimale $(x_1, x_2) = (0, 10)$ avec $z = 20$.

5. Les solutions de base réalisables non parcourues par le simplexe sont :
   — (7,3,0,0)
   — (7,0,3,0)
6. La contrainte dont la suppression rendrait le PL non borné est : $x_1 + x_2 \leq 10$.
7. Une contrainte dont l'ajout rendrait le PL non réalisable : $x_2 \geq 12$.

Solution de l'exercice 41

$$\begin{array}{rl} \text{maximiser} & z = x_1 + x_2 \\ \text{s.c.} & \left\{ \begin{array}{l} x_1 - x_2 \leq 3 \\ x_1 - 2x_2 \leq 3 \\ x_1, x_2 \geq 0 \end{array} \right. \end{array}$$

Ajout des variables d'écart e_1 et e_2 et écriture en forme canonique :

$$\begin{array}{rl} \text{maximiser} & z \\ \text{s.c.} & \left\{ \begin{array}{l} x_1 - x_2 + e_1 = 3 \\ x_1 - 2x_2 + e_2 = 3 \\ z - x_1 - x_2 = 0 \\ x_1, x_2, e_1, e_2 \geq 0 \end{array} \right. \end{array}$$

	z	x_1	x_2	e_1	e_2	
e_1	0	$\boxed{1}$	-1	1	0	3
e_2	0	1	-2	0	1	3
z	-1	-1	-1	0	0	0

	z	x_1	x_2	e_1	e_2	
e_1	0	1	-1	1	0	3
e_2	0	0	-1	-1	1	0
z	1	0	-2	1	0	3

Si on prend la colonne x_2 comme colonne du pivot (seule possibilité de toute façon), on remarque que tous les éléments sur cette colonne sont négatifs. Il n'y a donc pas de pivot possible. Le problème est non bornée (z peut augmenter à l'infini).

BIBLIOGRAPHIE

[1] John M. Harris, Jeffry L. Hirst, and Michael J. Mossinghoff. *Combinatorics and Graph Theory, second edition*. Springer, 2008.

[2] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1 :269–271, 1959.

[3] Lester R. Ford Jr. and Delbert R. Fulkerson. *Flows in networks*. Princeton University Press, 1962.

[4] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network flows : Theory, Algorithms, and Applications*. Prentice-Hall, Inc., 1993.

[5] Frederick S. Hillier and Gerald J. Lieberman. *Introduction to Operations Research (10th edition)*. McGraw-Hill, 2015.