Mathématiques et algorithmes

Graduat en informatique

Table des matières

Chapitre 1

Calcul matriciel

1.1 Notions de matrice

Les opérations que l'on effectue sur les tableaux de nombres ou matrices ne di ?erent pas des opérations élémentaires arithmétiques ordinaires telles que "+", "-", " ." ou "/". La nouveauté contenue dans le calcul matriciel réside plutôt dans la saisie simultanée de toutes les composantes de la matrice, auxquelles les mêmes opérations sont systématiquement appliquées. Cette visée de manipulation globale, qui rejoint ici les préoccupations de l'analyse multivariée, se reflète dans la notation de la matrice, que l'on représente par un seul symbole. L'écriture s'en trouve alors (considérablement) simplifiée, au prix d'une plus grande abstraction symbolique.
Une matrice de type (n * m) est un tableau de nombres à n lignes et m colonnes, que l'on note :

$$\begin{pmatrix} a_{11} & a_{12} & ... & a_{1m} \\ a_{21} & a_{22} & ... & a_{2m} \\ ... & ... & ... & ... \\ a_{n1} & a_{n2} & ... & a_{nm} \end{pmatrix}$$

a_{ij} représente le nombre ou composante situé dans la i-ème ligne (i = 1, ..., n) et la j-ème colonne (j = 1, ...,m) de la matrice A. Si n = m, on parle de matrice carrée. Les composantes a_{11}, a_{22},...,a_{nn} d'une matrice carrée forment sa diagonale principale.

1.2 Cas particuliers

- Vecteur ligne
- Vecteur colonne
- Matrice carrée

1.3 Opérations classiques

1.3.1 Addition

- Principe On additionne deux matrices de même forme

- Exemple
- Algorithme

1.3.2 Multiplication

Multiplication de deux matrices

- Principe
- Exemple
- Algorithmes
  Soit deux matrice carrée d'ordre n $\mathcal{A}$ et $\mathcal{B}$

```
Pour i allant de 1 à n, faire
        Pour j allant de 1 à n faire
                C(i,j)←0 ;
                pour k allant de 1 à n faire
                        C(i,j) ← C(i,j)+ A(i,k)*B(k,j)
```

Si les matrices ne sont pas carrées, par exemple $\mathcal{A}(n, m)$ et $\mathcal{B}(m, r)$

```
Pour i allant de 1 à m, faire
        Pour j allant de 1 à n faire
                C(i,j)←0 ;
                pour k allant de 1 à m faire
                        C(i,j) ← C(i,j)+ A(i,k)*B(k,j)
```

On constate que ces algorithmes sont d'ordre 3. On note $O(n^3)$.

Multiplication d'une matrice par un scalaire

1.3.3 Matrices carrées

Matrice identité

Matrice régulière

Matrice diagonale

Si toutes les composantes hors de la diagonale principale d'une matrice carrée sont nulles, cette matrice est dite diagonale.

Matrice triangulaire

Matrice inverse

Matrice transposée

On appelle matrice transposée de la matrice $\mathcal{A}$ (n, m) et on note $\mathcal{A}^t$ la matrice (m, n) qui est obtenue à partir de A en échangeant lignes et colonnes.
Si

$$\mathcal{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

alors

$$\mathcal{A}^t = \begin{pmatrix} a_{11} & a_{21} & ... & a_{n1} \\ a_{12} & a_{22} & ... & a_{n2} \\ ... & ... & ... & ... \\ a_{1m} & a_{2m} & ... & a_{mn} \end{pmatrix}$$

1.3.4 Matrice orthogonale

Une matrice carrée $\mathcal{A}$ est orthogonale si sa transposée est égale à son inverse :

$$\mathcal{A}^{-1} = \mathcal{A}^t$$

1.4 Opérations booléennes

Nous avons :

+	0	1
0	0	1
1	0	1

et aussi

*	0	1
0	0	0
1	0	1

A ne pas confondre avec le calcul binaire

Si les matrices ne sont pas carrées, par exemple $\mathcal{A}(n, m)$ et $\mathcal{B}(m, r)$

```
Pour i allant de 1 à m, faire
    Pour j allant de 1 à n faire
        C(i,j)←0 ;
        pour k allant de 1 à m faire
            si(C(i,j)=0) alors C(i,j) ← A(i,k)*B(k,j)
```

1.5 Inverse d'une matrice carrée

1.5.1 définition

L'inverse d'une matrice carrée $\mathcal{A}$ est notée $\mathcal{A}^{-1}$ et est telle que $\mathcal{A}$.$\mathcal{A}^{-1}$=$\mathcal{E}$ où $\mathcal{E}$ est la matrice unité.

1.5.2 Calcul de l'inverse d'une matrice carrée

Il faut :

1. S'assurer que l'inverse existe. A cet effet, il faut calculer le déterminant de la matrice.
   Si ce déterminant est nul, l'inverse n'existe pas et on arrete. Sinon, on poursuit ;
2. Construire une matrice auxiliaire de la manière suivante :
   (a) Pour obtenir l'élément (i,j), supprimer la ligne i et la colonne j de la matrice de départ.
   (b) Calculer le déterminant de la matrice restante ;
   (c) Multiplier cet élément par $(-1)^{i+j}$

3. Transposer la matrice ainsi obtenue.
4. Diviser chaque éléménet par le déterminant de la matrice de départ.

1.5.3 Exemple

Soit la matrice

$$\begin{pmatrix} 1 & 2 & -1 \\ 3 & 4 & 2 \\ 1 & 0 & 5 \end{pmatrix}$$

1. Calcul du déterminant.
   Le déterminant est $det\mathcal{A} = -2$. Il est différent de zéro.
   L'inverse existe donc.
2. Remplacons chque élément comme décrit ci-dessus. On obtient
$$\begin{pmatrix} 20 & -13 & -4 \\ -10 & 6 & 2 \\ 8 & -5 & -2 \end{pmatrix}$$
3. Transposons
$$\begin{pmatrix} 20 & -10 & 8 \\ -13 & 6 & -5 \\ -4 & 2 & -2 \end{pmatrix}$$
4. Divisons par le déterminant. On a :
$$\begin{pmatrix} -10 & 5 & -4 \\ \frac{13}{2} & -3 & \frac{5}{2} \\ 2 & -1 & 1 \end{pmatrix}$$
5. Vérifions :
$$\begin{pmatrix} -10 & 5 & -4 \\ \frac{13}{2} & -3 & \frac{5}{2} \\ 2 & -1 & 1 \end{pmatrix} . \begin{pmatrix} 1 & 2 & -1 \\ 3 & 4 & 2 \\ 1 & 0 & 5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$
   Et aussi :
$$\begin{pmatrix} 1 & 2 & -1 \\ 3 & 4 & 2 \\ 1 & 0 & 5 \end{pmatrix} . \begin{pmatrix} -10 & 5 & -4 \\ \frac{13}{2} & -3 & \frac{5}{2} \\ 2 & -1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

1.6 Applications

1.6.1 Matrice de transition

Exemple 1

La main-d'œuvre d'une petite localité est de 1 800 personnes. Sur ces personnes, 120 sont au chômage et 1 680 ont un emploi. Au cours d'une année, 10 % des travailleurs qui ont un emploi perdront leur emploi et 60 % des personnes qui sont au chômage trouveront un emploi.

a) Créez une matrice de 1 * 2 illustrant le nombre de personnes ayant un emploi et le nombre de personnes au chômage.
b) Complétez la matrice de transition et nommez cette matrice T.
c) Déterminez le nombre de personnes ayant un emploi et au chômage pendant la deuxième année.
d) Déterminez le nombre de personnes ayant un emploi et au chômage pendant la troisième année.
e) Répétez le même processus qu'à la question d) pour déterminer le nombre de personnes ayant un emploi et au chômage pendant les quatrième et cinquième années. Donnez des commentaires sur les résultats.

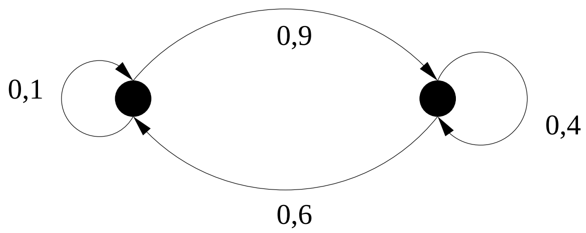


FIG. 1.1 –

On a :

$$\begin{pmatrix} 1680 & 120 \end{pmatrix} \begin{pmatrix} 0,9 & 0,1 \\ 0,6 & 0,4 \end{pmatrix} = \begin{pmatrix} 1584 & 216 \end{pmatrix}$$

$$\begin{pmatrix} 1584 & 216 \end{pmatrix} \begin{pmatrix} 0,9 & 0,1 \\ 0,6 & 0,4 \end{pmatrix} = \begin{pmatrix} 1555 & 245 \end{pmatrix}$$

$$\begin{pmatrix} 1555 & 245 \end{pmatrix} \begin{pmatrix} 0,9 & 0,1 \\ 0,6 & 0,4 \end{pmatrix} = \begin{pmatrix} 1546 & 253 \end{pmatrix}$$

$$\begin{pmatrix} 1546 & 253 \end{pmatrix} \begin{pmatrix} 0,9 & 0,1 \\ 0,6 & 0,4 \end{pmatrix} = \begin{pmatrix} 1544 & 256 \end{pmatrix}$$

Exemple 2

Soit la situation suivante :

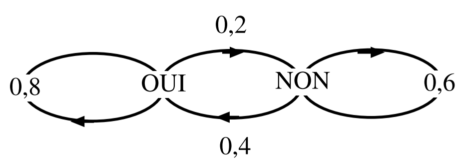


FIG. 1.2 –

On peut mettre sous forme matricielle :

$$\begin{array}{cc} & \begin{array}{cc} oui & non \end{array} \\ \begin{array}{c} oui \\ non \end{array} & \begin{pmatrix} 0,8 & 0,2 \\ 0,4 & 0,6 \end{pmatrix} \end{array}$$

Cette matrice est une matrice stochastique car, pour chaque ligne, la somme des éléments vaut 1(somme de probabilités)
On obtient alors :

$$\mathcal{M}^2 = \begin{pmatrix} 0,72 & 0,28 \\ 0,56 & 0,44 \end{pmatrix}$$

$$\mathcal{M}^3 = \begin{pmatrix} 0,69 & 0,31 \\ 0,62 & 0,38 \end{pmatrix}$$

$$\mathcal{M}^4 = \begin{pmatrix} 0,68 & 0,32 \\ 0,65 & 0,35 \end{pmatrix}$$

$$\mathcal{M}^5 = \begin{pmatrix} 0,67 & 0,33 \\ 0,66 & 0,34 \end{pmatrix}$$

En poursuivant, on constate que ces matrices tendent vers une limite précise.

1.6.2 transitions de durée minimale

Chapitre 2

Eléments de la théorie des graphes

2.1 Domaine d'application

Il s'agit ici de traiter essentiellement des problèmes d'existence et des problèmes d'optimisation.

1. Le voyageur de commerce
2. Le postier
3. Les 7 ponts de Koenigsberg
4. ...

2.2 Notion de graphe

2.2.1 Les types

– Orienté
– Non orienté

2.2.2 Exemples

Orienté	déplacement dans une ville avec sens interdit
	graphe des diviseurs (pour n = 12).
Non orienté	les 7 ponts de koenigsberg
	Session d'examens
Pondéré (orienté ou non)	planification des travaux
	choix d'un itineraire,

2.2.3 Représentation

1. Matrice d'adjacence : (sommets x sommets)
   (a) Concept orienté
   (b) Extension au concept non orienté
   La matrice est ici symétrique

2. Matrice d'incidence : (sommets x arcs)
   (a) Concept non orinté
   (b) Extension au concept orienté.
   Ici on utilise 1 si le sommet est le départ de l'arc et -1 si le sommet est l'extrémité de l'arc.
   (c)
3. Liste des successeurs : (liste chainée)

2.3 Quelques concepts

Les divers concepts s'articulent autour de la notion de sommet et d'arc ou arête

Concept orienté	concept non orienté
Arc	arete
chemin	chaine
circuit	cycle
p-graphe	multi-graphe
graphe fortement connexe	graphe connexe
Composante fortement connexe	composante connexe
arborescence	arbre

Remarque : Il y a toujours un graphe non orienté sous-jacent a un graphe orienté.

Notions communes
Parcours
simple
Elementaire
Cloture transitive d'un graphe simple

Le terme de parcours regroupe les chemins, les chaines, les circuits et les cycles.
Un parcours est :

Elémentaire si tous les sommets qui le composent sont tous différents

Simple si tous les arcs qui le composent sont tous différents

Hamiltonien passe une et une seule fois par chaque sommet du graphe.
Le voyageur de commerce

Eulerien passe une et une seule fois par chaque arc du graphe.
Les sept ponts de Koenigsberg

PréHamiltonien passe au moins une fois par chaque sommet du graphe

PréEulerien passe au moins une fois par chaque arc du graphe.
Le postier chinois

2.4 Sous graphe et graphe partiel d'un graphe donné

2.4.1 Sous graphe

On prend un sous ensemble des sommets du graphe
On prend TOUS les arcs concernés par les sommets retenus.

2.4.2 Graphe partiel

On prend TOUT les sommets du graphe
On ne considère qu'un sous ensemble des arcs.

2.5 Matrice d'adjacence

Interprétations

1. la ligne i donne les successeurs du sommet i
2. la colonne j donne les prédécesseurs du sommet j

Multiplication ordinaire

1. Soit un p-graphe :
   M^k donne le nombre de chemins de longeur k joignant i à j.

Multiplication booléenne

2. Soit un 1-graphe :
   M^k donne l'existence de chemins de longeur k joignant i à j.

3. La transformation $(I + M)$ rend le graphe reflexif (ajouter une boucle à chaque sommet)
   $(I + M)^k$ donne l'existence de chemins de longeur $\leq$ k joignant i à j avec i $\neq$ j.

4. La **fermeture transitive** est donnée par :
   $$M + M^2 + M^3 + M^4 + \ldots + M^{n-1}$$

5. $(I + M)^{n-1}$ donne l'existence de chemin joignant i à j avec i$\neq$j.
   Il s'agit de la fermeture reflexo-transitive.

6. Matrice d'accessibilité :
   Fermeture transitive
   On a
   $$\widehat{M} = M + M^2 + M^3 + M^4 + \ldots + M^{n-1}$$
   Fermeture reflexo-transitive
   On a
   $$\widehat{M} = (I + M)^{n-1} = I + M + M^2 + M^3 + M4 + \ldots + M^{n-1}$$
   Est-ce que j est accessible depuis i ?

7. $(I \oplus M)^{n-1+p}$ est nécessairement identique à $(I + M)^{n-1}$ car tout chemin élémentaire (qui ne passe pas plus d'une fois par un sommet) est nécessairement de longeur $\leq$ n-1

8. (I+M) converge bien plus vite que $M + M^2 + M3 \ldots$
   On cherche l'égalité $M^p = M^{2p}$.
   Ainsi si un graphe possède des chemins de longeur 8, on arrete la procedure apres avoir calculé M, M^2, M^4, M^8 au lieu de calculer M^2, M^3, M^4, M^5, M^6, M^7, M^8.

9. Calcul de la **fermeture transitive** par l'algorithme de **Warshall**.

   Parcourir toutes les colonnes. Dès que on rencontre un 1, parcourir la ligne et remplacer eventuellement un 0 par un 1 si, en parcourant la ligne dont le numero est égal au numéro de la colonne , on trouve un 1.

   ou encore,

   Pour chaque sommet S, examiner ses prédécesseurs P. Pour chacun d'eux, examiner les sommets qui ne sont pas ses successeurs.
   Pour chacun de ces sommets là, le designer quand meme comme successeur de P si on peut l'atteindre via le sommet S.

   ```
   pour k variant de 1 à n répéter
           pour i variant de 1 à n répéter
                   si A(i,k) alors
                           pour j variant de 1 à n répéter
                                   A(i,j) <- A(i,j) ou A(k,j)
   ```

   Ici A(k,j) designe un successeurs du sommet k.

10. Complexité
    La complexité d'un alogorithme est une mesure théorique du nombre d'opérations élémentaires **dans le pire des cas**.
    – Le produit de deux matrices est en $O(n^3)$
    – La matrice d'accessibilité est en $O(n^4)$ car on peut etre amené à faire n produits de deux matrices
    – Warshal est en $O(n^3)$ et est donc meilleur

11. Composante fortement connexe

    (a) Définition
    Une composante fortement connexe est un sous graphe (on peut ne pas prendre tous les sommets) tel que pour tout couple de sommets, il y a un chemin qui les relie (dans les deux sens)

    (b) Exemple graphique

    (c) Recherche à partir de la matrice d'accessibilité
    Ce sont les sommets qui correspondent à des lignes identiques de la matrice d'accessibilité. rappelons que l'on peut calculer la matrice d'accessibilité ($\widehat{M} = (I + M)^{n-1}$) à partir de l'algorithme de Warshall.

Soit le graphe 2.18 à la page 41.
La matrice d'adjacence est :

$$\mathcal{M} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

La matrice d'incidence est :

$$\mathcal{M} = \left(\begin{array}{c|ccccc} & a & b & c & d & e \\ \hline 1 & -1 & 0 & 1 & 0 & 0 \\ 2 & 1 & 1 & 1 & 0 & 0 \\ 3 & 0 & -1 & -1 & 0 & -1 \\ 4 & 0 & 0 & -1 & 0 & 0 \end{array}\right)$$

Cette matrice n'est pas nécessairement carrée contrairement à la matrice d'adjacence.
<u>Pour les opérations ordinaires :</u>

$$\mathcal{M}^2 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathcal{M}^3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathcal{M}^4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

<u>Pour les opérations booléennes</u>, on a :
La matrice d'adjacence est :

$$\mathcal{M} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathcal{M}^2 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathcal{M}^3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \mathcal{M}^4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

La somme de ces matrices nous donne :

$$\widehat{\mathcal{M}} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Il s'agit de la fermeture transitive.
Cette fermeture transitive peut être obtenue aussi par Warshall.

2.6 Arbres et arborescence

2.6.1 Arbre

Un arbre est un graphe
– non orienté
– connexe
– simple
– sans cycle
Il n'y a pas de racine. Un arbre ayant n sommets possède n-1 aretes.
Application : optimisation les réseaux.

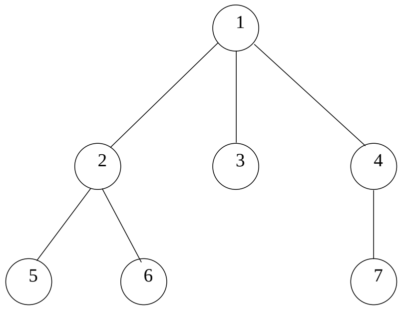


Fig. 2.1 –

2.6.2 Arbrorescence

Une arbrorescence est un graphe
– orienté
– il possède une racine
– il existe un et un seul chemin de la racine à tout autre sommet
– sans cycle

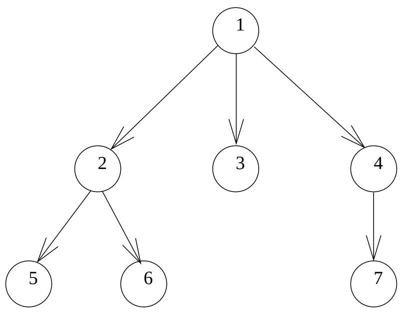


Fig. 2.2 –

La figure suivante n'est pas une arborescence.

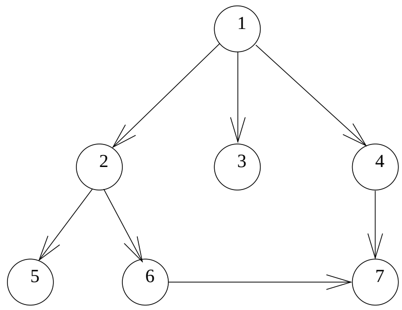


FIG. 2.3 –

2.7 Parcours de graphes

2.7.1 Pile et file

Les piles et files sont deux organisations différentes de la notion de *liste*
Les primitives sont : ajouter et supprimer.

- PILE

  - Organisation
    méthode LIFO ou *last in first out*
  - Primitives
    L'ajout se réalise au sommet
    La suppression se fait au sommet
  - Implémentation
    l'aide d'un tableau ou d'une liste

  pour un tableau, le sommet du tableau monte et descend alors que la base est fixe.
- FILE

  - Organisation
    méthode FIFO ou *first in first out*
  - Primitives
    L'ajout se réalise en queue
    La suppression se fait en tete.
  - Implémentation
    à l'aide d'un tableau ou d'une liste

  Si on utilise un tableau, l'ensemble "monte". Il faut un mécanisme pour compenser cette "montée" permanente.

2.7.2 Arborescence

En largeur

- Usage d'une file
- Exemple
- L'algorithme

```
Enfiler la racine
TANT QUE le file n'est pas vide
   soit n le premier noeud de la file
   {traitement de n}
   défiler
   enfiler tous les fils de n
FIN TANT QUE
```

– Exercices

Etape	file	liste des sommets visites
1	1	1
2	2, 3, 4	1, 2
3	3,4,5,6	1, 2, 3
4	4, 5, 6	1, 2, 3, 4
5	5, 6, 7	1,2,3,4,5
6	6, 7	1,2,3,4,5, 6
6	7	1,2,3,4,5, 6, 7
7	vide	

En profondeur

Le principe du parcours en profondeur d'un graphe (orientée ou non) est celui du parcours d'un labyrinthe : on va de sommet en sommet en marquant au fur et a mesure les sommets visitées. La visite se poursuit le plus loin possible tant qu'il reste des sommets accessibles non encore marquées. Quand on atteint un sommet z dont tous les voisins ont été déjà marquées alors on revient au sommet précédent z dans la visite.

– <u>Algorithme non récursif</u>
  On utilise une pile.

```
empiler la racine;
tant que la pile n'est pas vide
        soit s le sommet de la pile
        si s n'a pas été marqué alors
                {pré-traitement de s}
                marqué s
                si s a des fils alors
                        empiler le fils ainé de s
                fin de si
        sinon
                {Post-traitement de s}
                dépiler
                si s a des frères plus jeunes alors
                        empiler le frère puiné de s
                fin de si
        fin de si
fin de tant que
```

– <u>Procédure récursive</u>

```
visiter un noeud :
        {pré-taritemet de n}
```

```
        visiter tous les fils de n
        {post-traitement}
```

Exemple :
Voici l'évolution de la pile pour la figure 2.2. La pile évolue exactement de la même manière que l'on fasse le prétaritement ou le posttraitement.
Si le traitement consiste à écrire le numéro du noeud, on a :

1. Prétraitement : 1, 2, 3, 4, 5, 6, 7.
2. Posttraitement : 5, 6, 2, 3, 7, 4, 1.
3.

		5	5 $\surd$	6	6 $\surd$	
	2	2 $\surd$	2 $\surd$	2 $\surd$	2 $\surd$	2 $\surd$
1	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$

			7	7 $\surd$			
3	3 $\surd$	4	4 $\surd$	4 $\surd$	4 $\surd$		
1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	1 $\surd$	

Remarque

On peut ainsi déterminer les composantes connexes d'un graphe non orientée mais attention on ne détermine pas ainsi les composantes fortement connexes d'un graphe orienté

2.7.3 Arbres binaires

Principes

Un arbre binaire est un arbre tel que chaque noeud n'a pas plus de deux fils.
Il y a trois manières de parcourir un arbre binaire :

1. parcours infixe :
   La visite commence par le sous-arbre gauche, en contin uant par la racine et en finisant par le sous-arbre droit.
2. parcours préfixé :
   La racine est visitée d'abord
3. parcours postfixé :
   La racine est visitée en dernier lieu.

Exemples

Applications

Soit les arbres suivants 2.4 :
On a :

1. (a) parcours Préfixé : *23
   (b) parcours infixé : 2*3
   (c) parcours postfixé : 23*
2. (a) parcours infixé : 2*(3+5)
   (b) parcours postfixé : 235+*

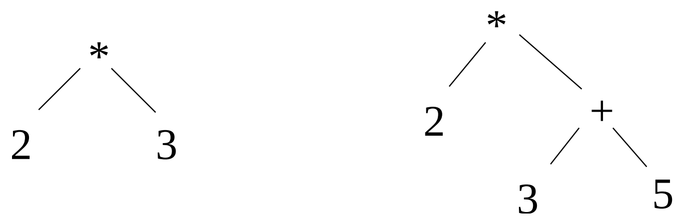


FIG. 2.4 –

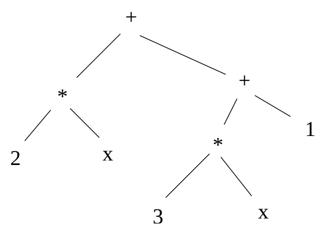


FIG. 2.5 –

La 2.5 représente l'expression $2x + 3x + 1$ en parcours infixé
La 2.6 représente l'expression $(a + \frac{b}{c}) * (d - e * f)$ en aprcours infixé
Représenter l'expression $x^2 + 2.x + 1$ sous la forme d'un arbre binaire.

2.7.4 Graphe orienté

Principe

On considére que l'on parcourt des arborescences associées aus sommets du graphe.
On part d'un sommet et puis on parcourt son arborescence.
on recommence à partir d'un sommet non atteint.

Parours en lageur

- C'est l'un des algorithmes de parcours les plus simple sur un graphe, et la base de nombreux algorithmes importants sur les graphes.
  Etant donne un graphe G = (S, A) et un sommet origine s, le parcours en largeur emprunte systematiquement les arc G pour "decouvrir" tous les sommet accessibles depuis s. Il calcule la distance (le plus petit nombre d'arcs) entre s et tous les sommets accessibles.
  Il construit egalement une "arborescence en largeur" de racine s, qui contient tous les sommets accessibles depuis s. Pour tout sommet v accessible depuis s, le chemin reliant s a v dans l'arborescence en largeur correspond a un "plus court chemin" de s vers v dans G, autrement dit un chemin contenant le plus petit nombre d'arc.
  L'algorithme de parcours en largeur tient son nom au fait qu'il decouvre d'abord tous les sommets situes a une distances k de s avant de decouvrir

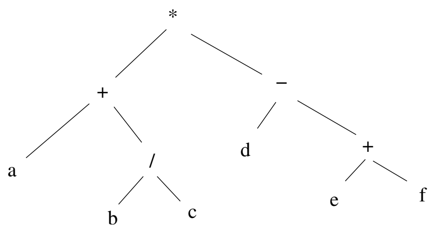


FIG. 2.6 –

tous les sommets situes a une distance k+1 de s.
– Exemple

Parcours en profondeur

– La strategie suivie par un parcours en profondeur est, comme son nom l'indique, de descendre plus "profondement" dans le graphe chaque fois que c'est possible.
  Lors d'un parcours en profondeur, les arcs sont explores a partir du sommet s decouvert le plus recemment et dont on n'a pas encore explore tous les arcs incidents. Lorsque tous les arcs de s ont ete explores, l'algorithme "revient en arriere" pour explorer les arc qui partent du sommet a partir duquel s a ete decouvert.
  Ce processus se repete jusqu'a ce que tous les sommets accessibles a partir du sommet origine initial aient ete decouverts.
  S'il reste des sommets non decouverts, on en choisit un qui servira de nouvelle origine, et le parcours reprend a partir de cette origine. Le processus complet est repete jusqu'a ce que tous les sommets aient ete decouverts.
  Contrairement au parcours en largeur, pour lequel le sous-graphe de liaison forme une arborescence, le sous-graphe de liaison obtenu par un parcours en profondeur peut etre compose de plusieurs arborescences, car le parcours peut etre repete a partir de plusieurs origines.
– Exemple

Remarques

– Selon le point de départ, on obtient un parcours différent.
– Pour un même point de départ, selon le type de parcours, on obtient une succession différente de sommets.

Ordre d'exploration en profondeur (prétraitement) : 1, 2,4, 3, 6, 5.
On peut alors y associer le graphe suivant :
Il s'agit d'un arbre recouvrant (spanning tree).

2.7.5 Graphe non orienté

On considère que tout sommet relié à un autre est accessible depuis cet autre.

On appelle **squelette** un **arbre** contenant tous les sommets du graphe. Il s'agit donc d'un graphe partiel. Pn parle aussi d'arbre couvrant.

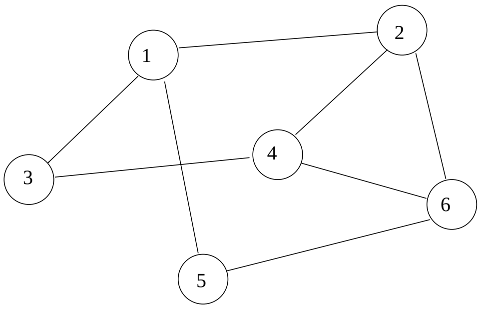


Fig. 2.7 –

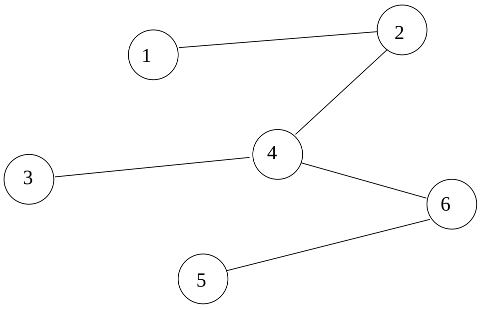


Fig. 2.8 –

Profondeur

– Cet algorithme s'appelle aussi algorithme du Petit Poucet (Tarjan).
– Le graphe partiel obtenu s'appelle un squelette si le graphe est simplement connexe.

Largeur

On obtient un arbre. Si tous les sommets du graphe on été atteint, on obtient un squelette. Sinon, le graphe n'est pas simplement connexe et on obtient un arbre de la foret

Remarques

Un graphe peut donc posséder plusieurs squelettes liés au parcours en largeur, en profondeur et selon aussi le point de départ ...

Applications des parcours

1. Parcours en profondeur Un parcours en profondeur permet de déterminer les composantes connexes d'un graphe non orienté mais attention on ne détermine pas ainsi les composantes fortement connexes d'un graphe orienté. En effectuant le parcours en profondeur d'un graphe à partir d'un sommet,

on note que l'on engendre un arbre couvrant de la composante connexe de x.

2. Parcours en largeur Un parcours en largeur permet de déterminer les composantes connexes d'un graphe non orienté mais attention, comme pour le parcours en profondeur, on ne détermine pas ainsi les composantes fortement connexes d'un graphe orienté.

3. Accessibilité.
   Un parcours nous permet d'obtenir les sommets accessibles depuis le point de départ choisi.
4. Composantes simplement connexes d'un graphe non orienté.
   Adopter une des méthodes de parcours.
   La composante connexe d'un sommet s est l'ensemble des sommets accessibles depuis s
5. Composantes fortements connexes d'un graphe orienté
   Pour connaitre les composantes fortement connexes, il suffit de rechercher des lignes identiques dans la matrice de la fermeture reflexo transitive.
   On peut le faire aussi par l'exploration du graphe, mais la méthode est plus complexe (voir plus loin) .
6. Présence d'un circuit.
   On part d'un sommet. Si lors du parcours en **profondeur** on retombe sur un sommet, il y a circuit. Si le parcours était en largeur, la difficulté provient du fait qu'un meme sommet peut etre à des "distances" différentes du point de départ sans qu'il n'y ait de circuit.

Exemples

Les figures 2.10 et 2.11 donne deux parcours obtenu en partant de E et aussi en partant de C à l'aide d'un parcours en profondeur pretraitement.

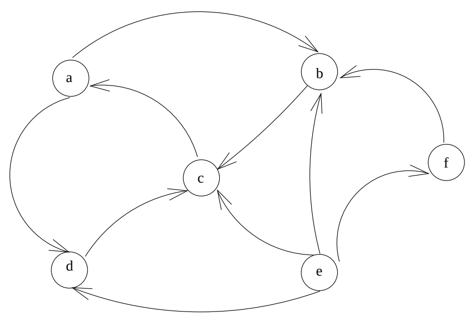


FIG. 2.9 –

Composantes fortement connexes

Contrairement au cas d'un graphe non orienté, un parcours de graphe (en largeur ou en profondeur) ne permet pas de déterminer les composantes fortement connexes d'un graphe G orienté.

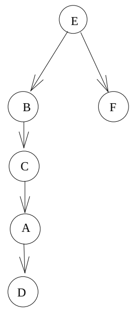


Fig. 2.10 –

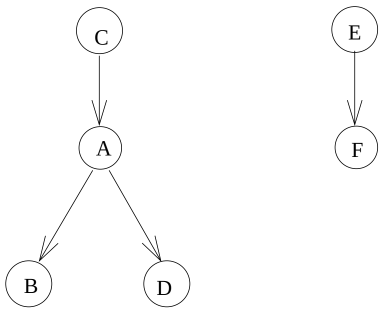


Fig. 2.11 –

Cependant, une procédure existe :

1. On réalise un parcours en profondeur à partir d'un sommet et on numérote les sommets en post-visite. (relancer le parcours jusqu'a ce que tous les sommets soient marqués).

2. Construire le graphe qui a les mêmes sommets que G mais dont les arcs ont l'orientation inverse.
3. Effectuer un parcours en profondeur en pre-traitement de ce nouveau graphe à partir du sommet qui a l'indice de post-visite le plus élevé. L'arborescence obtenue est la composante fortement connexe de ce sommet.
4. S'il reste des sommets non marqués par cette deuxième visite, il faut choisir celui qui a l'indice de post-visite le plus elevé et relancer un parcours.

2.8 Parcours particuliers

2.8.1 Parcours Hamiltonien

– Définition
  C'est un parcours qui passe une et une seule fois par chaque **sommet** du graphe.

- Exemple
  Problème d'ordonnancement

- Théorème de König : Tout graphe complet adment au moins un chemin hamiltonien

2.8.2 Parcours Eulérien

- Définition :
  C'est un parcours qui passe une et une seule fois par chaque **arc** du graphe.

- Exemples :

  - Pour le graphe 2.27
    Le chemin 3, 1, 2, 3, 4, 5, 6, 4 est eulerien
    Il n'y a pas de circuit eulerien.
    Pour $G^{'}$, le chemin 3, 1, 2, 3, 4, 5, 6, 3 est eulerien

  - Pour le graphe 2.24, G est un graphe qui adment un chemin hamiltonien, mais pas de circuit hamiltonien.

  - Pour le graphe 2.25, le chemin passant par 3, 1, 2, 4, 3, 2 est eulerien mais non hamiltonien. Par contre, 3, 1, 2, 4 l'est.

- Quelques théorèmes :

  1. Graphe orienté : **Présence d'un circuit** : Théorème de Camion
     La CNS pour admetre un circuit Eulerien est d'etre fortement connexe

  2. Graphe non orienté :

     Présence d'une chaine eulérienne Pour qu'un graphe simplement connexe admette une chaine eulérienne, il faut et il suffit que le nombre de sommets de degré impair soit 0 ou 2.

     Présence d'un circuit eulérien Pour qu'ungraphe simplement connexe admette un circuit eulerien, il faut et il suffit que chaque sommet soit de degré pair.

- Applications :
  Les 7 ponts de Konigsberg.
  La modelisation de ce problème nous montre qu'il faut trouver un circuit eulerien. Or on constate que tous les sommets ne sont pas de degré pair. Donc il est impossible de trouver un circuit eulerien.
  Lettre fermée et ouverte

2.9 Le plus court chemin

2.9.1 Les données

Graphe simple valué
Tous les poids sont positifs prcequ'il s'agit du plus **court** chemin.

2.9.2 Principe d'optimalité

Le principe d'optimalité dit que les sous-chemins des plus courts chemins sont des plus courts chemins. Cette propriété de sous-structure optimale permet d'envisager deux techniques pour rechercher un plus court chemin :

1. un algorithme glouton : Dijkstra : un algorithme glouton est un algorithme qui confronté à un choix, choisit ce qui lui semble le meilleur pour avancer. C'est un choix local, et on espère que la succession de choix locaux va amener à une " bonne solution ".(Dijkstra, Kruskall, Prim, Huffmann, ... sont des algorithmes gloutons)
2. un algorithme de programmation dynamique : Floyd-Warshall : un algorithme de programmation dynamique, résout les sous-problèmes d'un problème une fois et une seule, et stocke les résultats dans un tableau. Le triangle de Pascal est un algorithme de programmation dynamique pour le calcul des coefficients du binôme.

2.9.3 Circuit absorbant

Circuit qui provoque une diminution perpétuel de la longueur du chemin lorsque l'on cherche un chemin minimal
Circuit qui provoque une augmentation perpétuel de la longueur du chemin lorsque l'on cherche un chemin maximum

2.9.4 D'un sommet à tous les autres ou Dijkstra

D'un sommet à tous les autres
Algorithme non matriciel

Contexte

- un graphe pondéré
- le graphe est complet, sinon le compléter
- Tous les poids sont positifs
- ici, puisque tous les poids sont positifs, il n'y a pas de circuit absorbant

Données

- la matrice d'adjacence du graphe
- les ensembles D et X.
- : x_r : sommet de départ
- $pi(x_i)$: poids de x_r à x_i
- etiq

Représentation des données

D, $\overline{D}$ *et* X sont représentés à l'aide d'un seul tableau.
Ce tableau X est scindé en deux parties séparées par un indice l qui varie au cours de l'algorithme.

Algorithme de Dijkstra

initialisation de :
D
$pi(x_r)$
les pi : $pi(x) \leftarrow p(x_r, x_i)$

$\underline{\forall x \in D \neq X}$
choisir $y \in \overline{D} : pi(y) = min\{pi(x) \mid x \in \overline{D}\}$
$D \leftarrow D \cup \{y\}$(mise à jour de D)
$\underline{\forall x \in \overline{D}}$
$pi(x) \leftarrow min\{pi(x)\ ,\ pi(y) + p(y,x)\}$

Le tableau pi nous donne la longueur minimale en utilisant les sommets de D sans se soucier du chemin emprunté par ces sommets de D

Pour la **mémorisation du chemin**, on utilise un tableau etiq qui mémorise le sommet qui a provoqué une mise à jour ;

Exemple

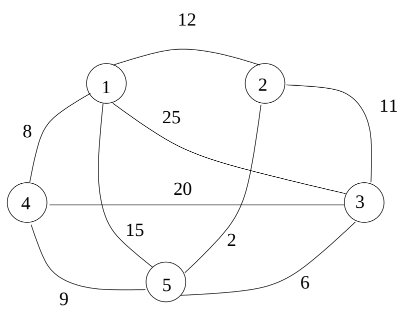


FIG. 2.12 –

La matrice d'adjacence de la figure 2.12 est :

$$\begin{pmatrix} 0 & 12 & 25 & 8 & 15 \\ 12 & 0 & 11 & \infty & 2 \\ 25 & 11 & 0 & 20 & 6 \\ 8 & \infty & 20 & 0 & 9 \\ 15 & 2 & 6 & 9 & 0 \end{pmatrix}$$

On recherche les chemins les plus courts en partant du sommet x_1.
On classe les sommets en deus sous-ensembles : Les sommets pour lesquels on

a dèjà une solution (D) et ceux qui sont en attente d'une solution ($\bar{D}$).
Au départ on a : $D = \{x_1\}$ et $\bar{D} = \{x_2, x_3, x_4, x_5\}$

Initialisation

On a le tableau de départ suivant :

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	25	8	15

Itération 1

On choisit le sommet le plus proche de x_1. C'est ici x_4. Le chemin qui mène de x_1 vers un autre sommet est-il plus court en passant par x_4 ?
Pi(x_2)=min(12,8+∞)=(12,∞)=12 pas de changement.
Pi(x_3)=min(25, 8+20)=(25,28)=25 pas de changement
Pi(x_5)=min(15, 17)=15 pas de mise à jour
La réponse est non pour tous les sommets. Le tableau est donc inchangé.
On obtient : $D = \{x_1, x_4\}$ et $\bar{D} = \{x_2, x_3, , x_5\}$

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	25	8	15

Itération 2

On choisit le sommet dont le poids est le plus faible. C'est x_2. Le chemin qui mène de x_1 vers un autre sommet est-il plus court en passant par x_2 ?
Pi(x_3)=min(25, 12+11)=(25,23)=25 mise à jour
Pi(x_4)=min(15, 12+2)=14 mise à jour

Ici, on a des mise à jour. On obtient : $D = \{x_1, x_4, x_2\}$ et $\bar{D} = \{x_3, x_5\}$

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	23	8	14

Itération 3

On considère le sommet x_5. Le chemin qui mène de x_1 vers chacun des sommets restant est-il plus court en passant par x_5 ?
$D = \{x_1, x_4, x_2, x_5\}$ et $\bar{D} = \{x_3\}$
Pi(x_3)=min(23, 14+6)=(23,20)=25 mise à jour

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	20	8	14

Ce tableau donne les longueurs les plus courtes possibles pour atteindre tous les sommets en partant de x_1. Mais ce tableau donne le poids des chemins mais ne permet pas de reconstituer le chemin. pour pouvoir le faire, il faut ajouter de l'information à chacun des tableaux.

2.9.5 Nouvel algorithme

Initialisation

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	25	8	15
$Etiq$	x_1	x_1	x_1	x_1	x_1

Itération 1

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	25	8	15
$Etiq$	x_1	x_1	x_1	x_1	x_1

Itération 2

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	25	8	15
$Etiq$	x_1	x_1	x_2	x_1	x_2

Itération 3

	x_1	x_2	x_3	x_4	x_5
Pi	0	12	23	8	14
$Etiq$	x_1	x_1	x_5	x_1	x_2

On reconstitue le chemin en aprtant de la solution. Par exemple quel est le plus court chemin pour aboutir à x_3 ?
Le dernier tableau nous indique c'est x_5 qui précede x_3
Puis l'avant dernier tableau nous dit que c'est x_2 qui précede x_5.
Puis le tableau précedent nous indique que c'est x_1 qui precede x_2.
On a donc

$$x_1 \to x_2 \to x_5 \to x_3$$

.

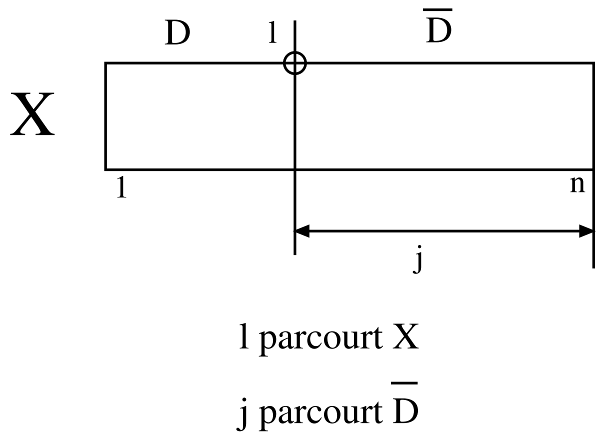


FIG. 2.13 –

2.9.6 Algorithme de FLOYD

principe

On part de la matrice d'adjacence.
On la transforme comme suit :
Pour chaque colonne, la parcourir.
Pour tout élément A de cette colonne qui n'est pas l'infini,
parcourir la ligne qui le contient.
Comparer chacun des éléments B de cette ligne avec A plus l'élément de la meme colonne que B situé sur la ligne symétrique à la colonne de A. Si cette somme est plus petite remplacer B par cette somme.
ou encore :
Pour chaque sommet :
Examiner les prédecesseurs. Pour chaque prédecesseurs, examiner les successeurs . On se demande si en allant directement d'un sommet au successeur d'un de ses prédecesseurs le chemin ne serait pas plus court que d'aller directement de ce sommet à ce successeur.

Exemple

Pour la la matrice de la figure 2.12 nous n'allons signaler que les mise à jour. Notons aussi qu'il faut considéré comme successeurs des points dont la distance est marquée comme ∞ car elle peut etre modifiée par le détour que l'on calcule. On a :

Sommet 1 Les prédécesseurs sont : 2, 3, 4, 5.

les successeurs de 2 sont : 3, 4, 5.
On a : $2 \rightarrow 1 \rightarrow 4? <?2 \rightarrow 4 : 12 + 8 < \infty$ mise à jour

les successeurs de 4 sont : 2, 3, 5.
On a : $4 \rightarrow 1 \rightarrow 2? <?4 \rightarrow 2 : 8 + 12 < \infty$ mise à jour
on a

$$\begin{pmatrix} 0 & 12 & 25 & 8 & 15 \\ 12 & 0 & 11 & 20 & 2 \\ 25 & 11 & 0 & 20 & 6 \\ 8 & 20 & 20 & 0 & 9 \\ 15 & 2 & 6 & 9 & 0 \end{pmatrix}$$

Sommet 2 Les prédécesseurs sont : 1, 3, 5.

les successeurs de 1 sont : 3, 4, 5.
$1 \rightarrow 2 \rightarrow 3? <?1 \rightarrow 3 : 12 + 11 < 25$ mise à jour
$1 \rightarrow 2 \rightarrow 5? <?1 \rightarrow 5 : 12 + 2 < 15$ mise à jour

les successeurs de 3 sont : 1, 4, 5.
$3 \rightarrow 2 \rightarrow 1? <?3 \rightarrow 1 : 11 + 12 < 25$ mise à jour

les successeurs de 5 sont : 1, 3, 4.
$5 \rightarrow 2 \rightarrow 1? <?5 \rightarrow 1 : 12 + 2 < 15$ mise à jour

on a

$$\begin{pmatrix} 0 & 12 & 23 & 8 & 14 \\ 12 & 0 & 11 & 20 & 2 \\ 23 & 11 & 0 & 20 & 6 \\ 8 & 20 & 20 & 0 & 9 \\ 14 & 2 & 6 & 9 & 0 \end{pmatrix}$$

Sommet 3 Les prédécesseurs sont : 1, 2, 4, 5.

les successeurs de 2 sont : 1, 4, 5.
$2 \rightarrow 3 \rightarrow 4? <?2 \rightarrow 4 : 11 + 20 < \infty$ mise à jour

les successeurs de 4 sont : 1, 2, 5.
$4 \rightarrow 3 \rightarrow 2? <?4 \rightarrow 2 : 11 + 20 < \infty$ mise à jour

on a

$$\begin{pmatrix} 0 & 12 & 23 & 8 & 14 \\ 12 & 0 & 11 & 20 & 2 \\ 23 & 11 & 0 & 20 & 6 \\ 8 & 20 & 20 & 0 & 9 \\ 14 & 2 & 6 & 9 & 0 \end{pmatrix}$$

Sommet 4

Sommet 5 Les prédécesseurs sont : 1, 2, 3, 4.

les successeurs de 1 sont : 2, 3, 4.
$1 \rightarrow 5 \rightarrow 3? <?1 \rightarrow 3 : 14 + 6 < 23$ mise à jour

les successeurs de 2 sont : 1, 3, 4. $2 \rightarrow 5 \rightarrow 3? <?2 \rightarrow 3 : 2 + 6 < 11$ mise à jour
$2 \rightarrow 5 \rightarrow 4? <?2 \rightarrow 4 : 2 + 9 < 20$ mise à jour

les successeurs de 3 sont : 1, 2, 4, 5. $3 \rightarrow 5 \rightarrow 1? <?3 \rightarrow 1 : 6 + 14 < 23$ mise à jour
$3 \rightarrow 5 \rightarrow 2? <?3 \rightarrow 2 : 6 + 2 < 11$ mise à jour
$3 \rightarrow 5 \rightarrow 4? <?3 \rightarrow 4 : 6 + 9 < 20$ mise à jour

les successeurs de 4 sont : 1, 2, 3. $4 \rightarrow 5 \rightarrow 3? <?4 \rightarrow 3 : 9 + 6 < 20$ mise à jour
$4 \rightarrow 5 \rightarrow 2? <?4 \rightarrow 2 : 9 + 2 < 20$ mise à jour

on a

$$\begin{pmatrix} 0 & 12 & 20 & 8 & 14 \\ 12 & 0 & 8 & 11 & 2 \\ 20 & 8 & 0 & 15 & 6 \\ 8 & 11 & 15 & 0 & 9 \\ 14 & 2 & 6 & 9 & 0 \end{pmatrix}$$

Le plus court chemin

↓

Principe d'optimalité :

Les sous chemins des plus courts chemins sont des plus courts chemins

↓

Remarque :

Pas de circuit absorbant.

Entre tout couple du graphe	**D'un sommet vers tous les autres**
FLOYD-WARSHALL Algorithme matriciel	DIJKSTRA : il faut que - tous les arcs soient positifs - rendre le graphe complet. Il s'agit d'un algorithme glouton non matriciel BELLMAN Algorithme - rapide - pas matriciel - peut détecter la présence de circuit absorbant ; Si il n'y a pas de circuit, on peut avoir des arcs positifs et négatifs. C'est donc de ce point de vue une amelioration de Dijkstra.

2.9.7 Algorithme de FLOYD

Entre tous les couples de sommets
Algorithme matriciel.

2.10 Graphes sans circuit

Les graphes orientés sans circuit peuvent représenter des relations.Voir figure 2.15
L'expression $a(a/(a+b+c)) - (a : (a+b+c))((a+b+c)+d)(c/d)$ Voir figure 2.16.

La relation "plus petit" définie sur des entiers.

2.10.1 Tri topologique

- Définition
  Le tri topologique d'un graphe orienté sans circuit est une numérotation des sommets dans laquelle les descendants d'un sommet de numéro k sont nécessairement de numéro supérieur à k. Voir figure 2.17

- Construction
  L'ordre inverse de l'ordre de post-visite, dans le parcours en profondeur d'un graphe, permet le tri topologique.
  L'ordre de post visite est : G, D, C, E, B, F, A.
  Inversons : A, F, B, E, C, D, G. On constate qu'un noeud quelconque a bien ses descendants situés à sa droite !

2.10.2 Décomposition en niveaux

1. Notion Décomposer un graphe en niveaux consiste à partitionner l'ensemble des sommets du graphe en sous ensembles appelès niveaux de telle manière que :
   - Tous les sommets d'un meme niveau ne sont pas reliés entre eux.
   - aucun sommet d'un niveau i ne possède d'ascendant dans les niveaux i+1, i+2 ...
   - Les sommets du niveau 1 n'ont pas de précédent et ce sont les seuls
   - les sommets du dernier niveau n'ont pas de suivants.
2. Exemples
3. CNS
   Pour qu'un graphe soit décomposable en niveaux, il faut et il suffit qu'il soit sans circuit.
4. Méthode
   - **A partir de la matrice d'adjacence**.
     On supprime, à chaque itération, tous les sommets qui correspondent à une colonne nulle, donc qui n'ont pas de précédents. Soit la matrice

d'adjacence suivante :

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
x_1	0	1	0	1	1	0	0	0
x_2	0	0	0	1	0	1	0	0
x_3	0	0	0	1	0	0	0	0
x_4	0	0	0	0	0	0	0	0
x_5	0	1	2	0	0	1	0	0
x_6	0	0	0	0	0	0	0	0
x_7	0	0	1	0	0	3	0	1
x_8	0	0	1	0	0	1	0	0

On sélectionne les sommets n'ayant aucun précédent. Ce sont donc ceux dont la colonne est constituée de zéro uniquement. Ici x_1 et x_7. On supprime les lignes et colonnes realtives aux sommets 1 et 7. on recommance.

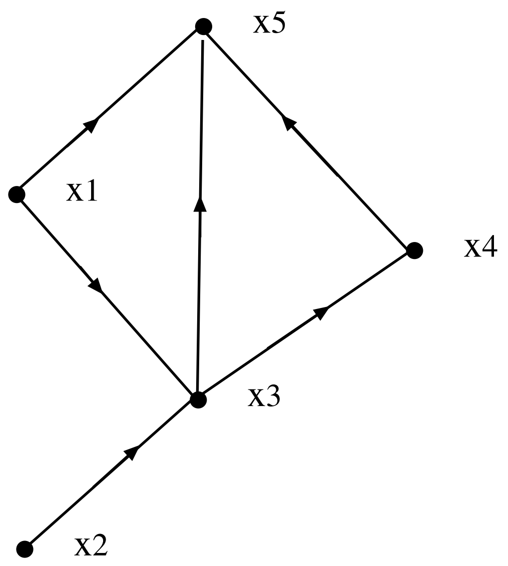
– **A partir de la fermeture reflexo-transitive**
  Sélectionner, les sommets pour lesquels les colonnes correspondantes contiennet un et un seul 1 : ce sont les sommets de niveau 1.
  Supprimer les lignes et colonnes correspondant aux sommets de niveau 1.
  Recommencer.

5. <u>Remarque</u>

– Le fait que ce soit une CNS implique que la décomposition en niveaux peut servir de test pour l'existence d'un circuit.
– La décomposition en niveaux donne un tri topologique

2.10.3 Exercice

Soit la figure suivante :



Faites la décomposition

1. à partie du graphe
2. à partir de la amtrice d'adjacence
3. à partir de la amtrice d'accessibilité

2.11 Arbre recouvrant de poids minimal

On donne un graphe connexe non orienté valué.
Un arbre de recouvrement est un graphe partiel qui est un arbre contenant chaque sommet de G.
Un arbre recouvrant de poids minimal est un arbre de poids total minimal.
Il s'agit d'enlever des aretes de façon qu'il reste connexe et que la somme des valuations soit minimale.
On recherche donc un graphe partiel pour obtenir un arbre.

1. Exemple
2. Algorithme de Kruskal
   On enleve toutes les aretes
   On considere les aretes par ordre de poids en commencant par la plus petite.
   On ajoute les aretes tant qu'elles ne forment pas un cycle ; une arete qui provoque un circuit est éliminée.

2.12 Quelques paradigmes

- Diviser pour regner
  Séparation et évaluation
- Programmation dynamique
- Algorithme glouton
  Dijkstra

2.13 Optimisation

- **Exemples :**
  Le voyageur de commerce
  Le sac à dos
- **Méthodes de résolution**
  <u>Heuristiques</u> :
  algorithme ne fournisant pas nécessairement un esolution optimale.
  <u>Métaheuristiques</u> :
  Méthode tabou
  le recuit simulé
  Algorithmes génétiques
  Algorithmes de fourmis
  Approximations

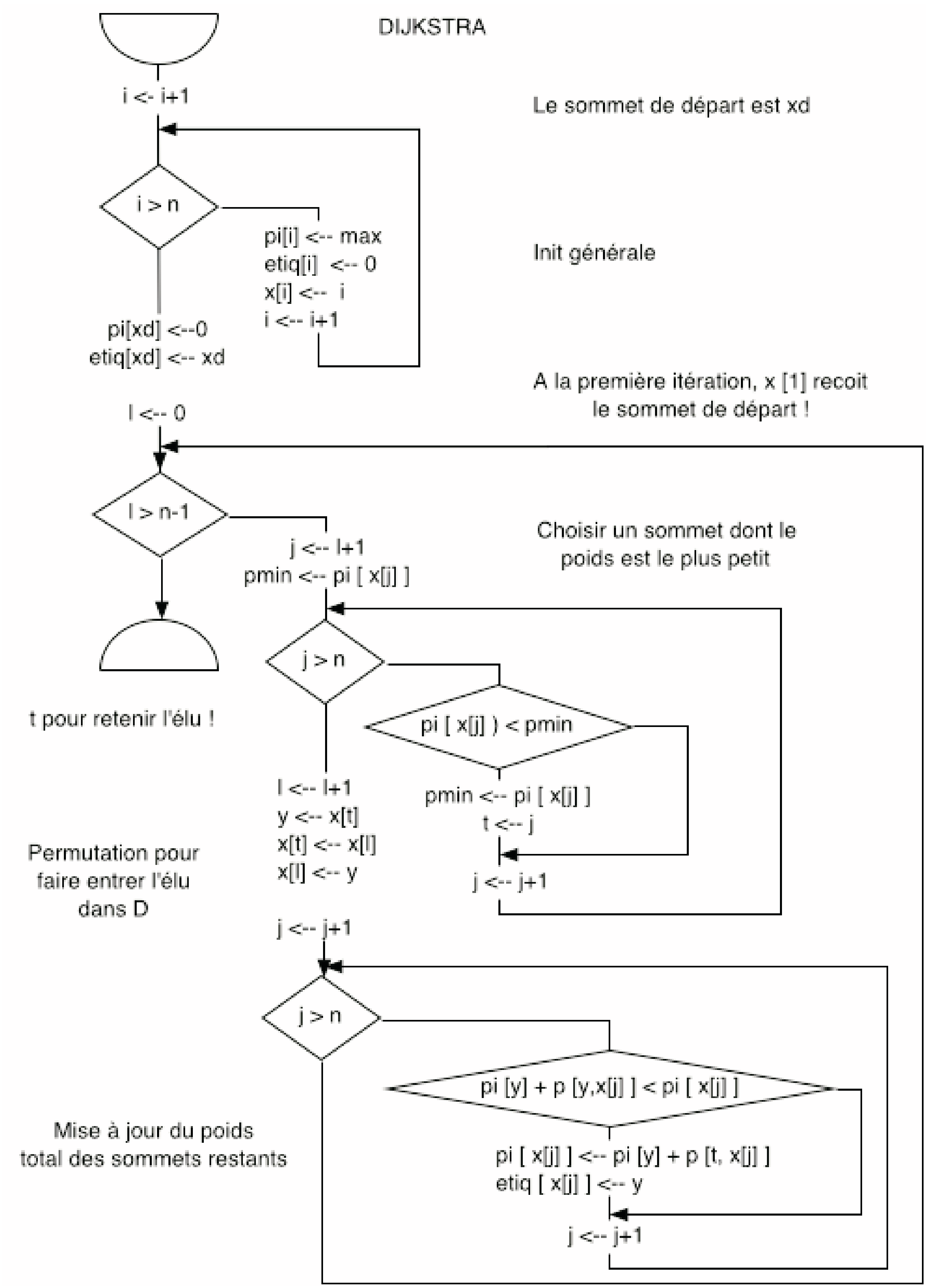

DIJKSTRA
i <- i+1
Le sommet de départ est xd
i > n
pi[i] <-- max
etiq[i] <-- 0
x[i] <-- i
i <-- i+1
Init générale
pi[xd] <--0
etiq[xd] <-- xd
A la première itération, x [1] recoit
le sommet de départ !
l <-- 0
l > n-1
j <-- l+1
pmin <-- pi [x[j]]
Choisir un sommet dont le
poids est le plus petit
j > n
pi [x[j]) < pmin
t pour retenir l'élu !
l <-- l+1
y <-- x[t]
x[t] <-- x[l]
x[l] <-- y
pmin <-- pi [x[j]]
t <-- j
j <-- j+1
Permutation pour
faire entrer l'élu
dans D
j <-- j+1
j > n
pi [y] + p [y,x[j]] < pi [x[j]]
Mise à jour du poids
total des sommets restants
pi [x[j]] <-- pi [y] + p [t, x[j]]
etiq [x[j]] <-- y
j <-- j+1

Fig. 2.14 –

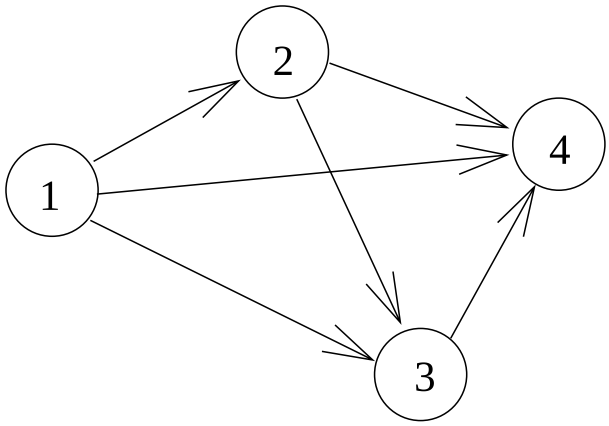


FIG. 2.15 –

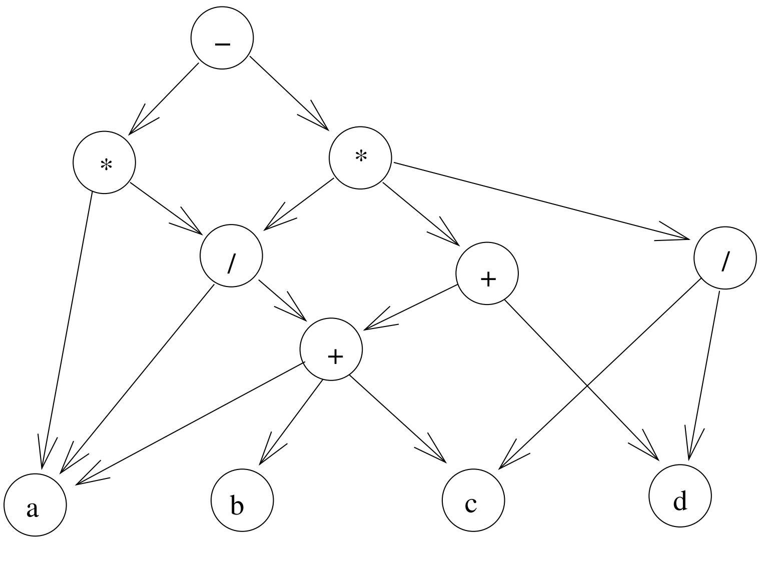


FIG. 2.16 –

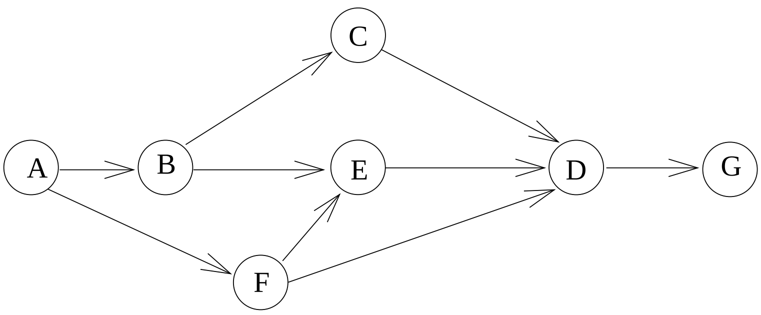


FIG. 2.17 –

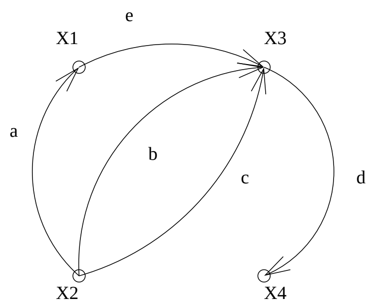


FIG. 2.18 –
[p]

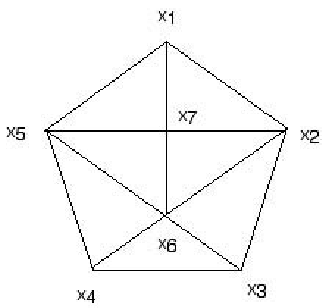


FIG. 2.19 –

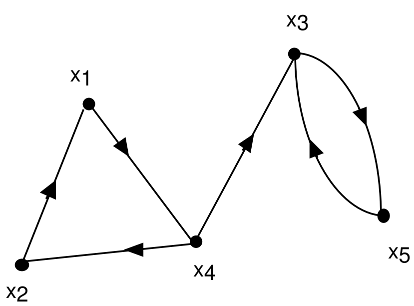


FIG. 2.20 –

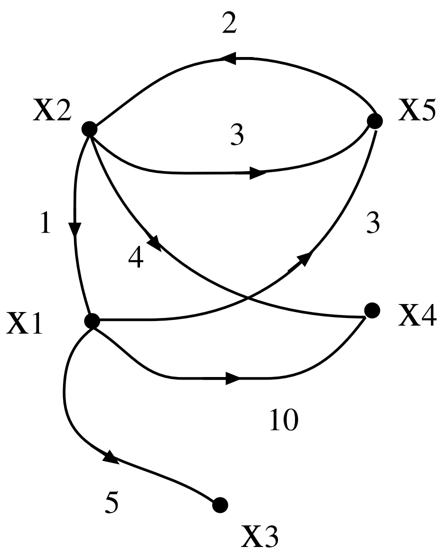


Fig. 2.21 –

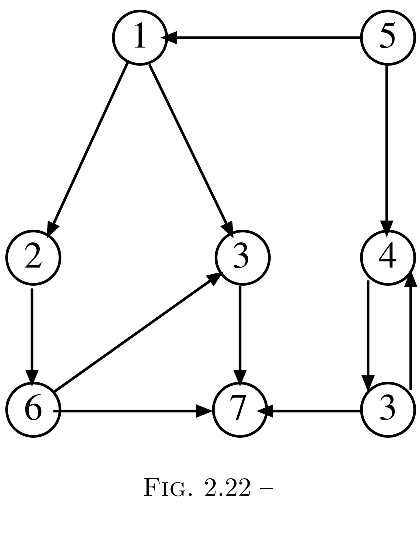


Fig. 2.22 –

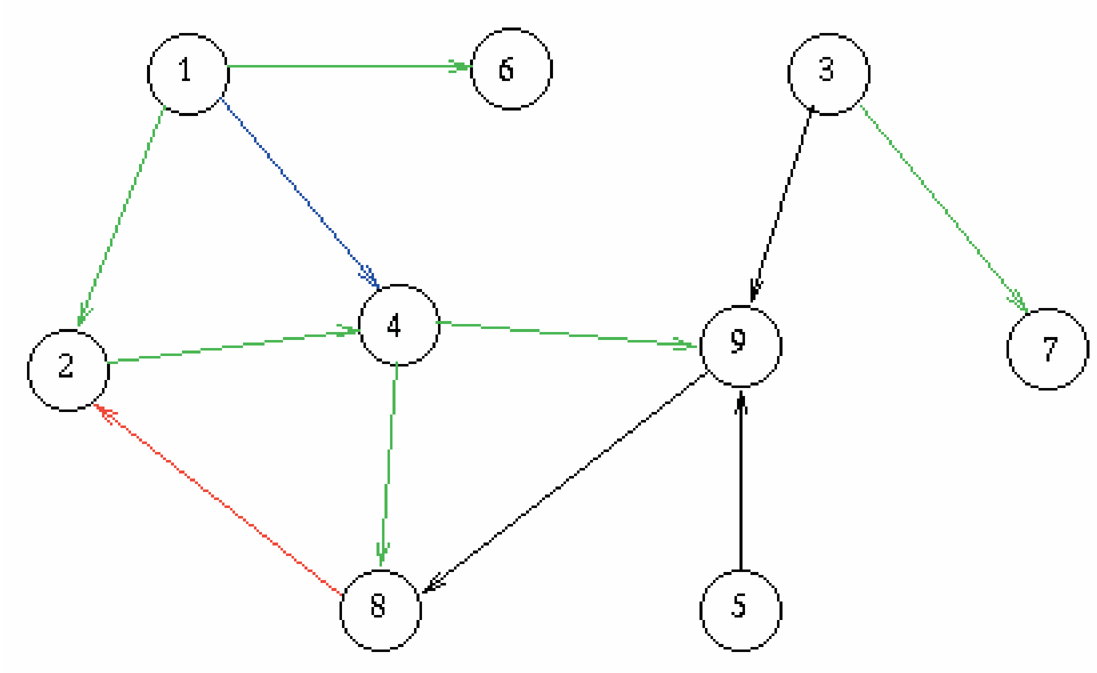


FIG. 2.23 –

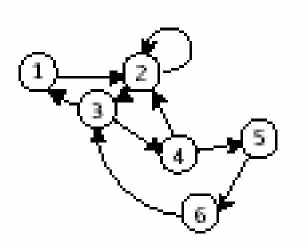


FIG. 2.24 –

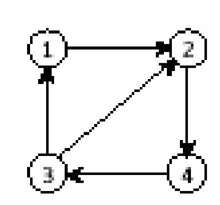


FIG. 2.25 –

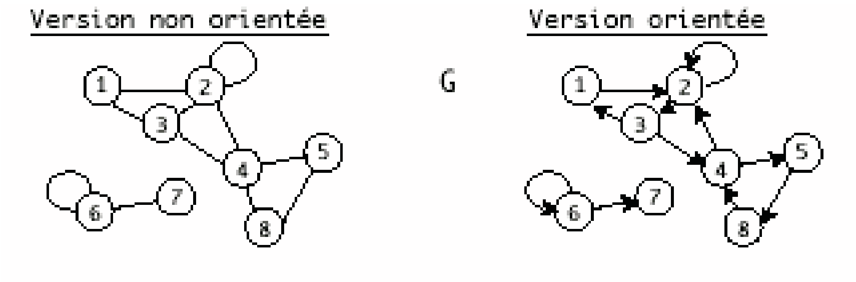


Fig. 2.26 –

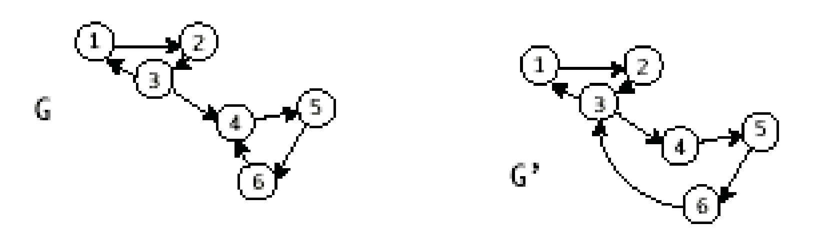


Fig. 2.27 –

Chapitre 3

Arbres binaires

3.1 Arbres de recherche

3.1.1 Définition

On a deux possibilités :

1. à gauche si $<$
2. à gauche si $\leq$

3.1.2 Exemples et contre-exemples

3.1.3 Intérêt

3.1.4 Représentations

Par pointeurs

Par tableau

On peut utiliser trois tableaux en parallèles : un pour les noeuds, un pour le SAG et un pour le SAD.
Par exemple :

i	SAG	Noeud	SAD
5	0	10	0
4	1	9	5
3	4	5	2
2	0	8	0
1	0	6	0

3.1.5 Arbre binaire "plein" ou "complet"

Tous les noeuds, sauf peut-être au dernier niveau, ont deux fils.

3.1.6 Insertion

L'ordre d'insertion influence la forme de l'arbre.
Exemple : 2, 1, 4 puis 4, 2, 1.

3.1.7 Suppression

1. Une feuille
2. un noeud avec un seul fils
   (a) si le noeud n'a pas de fils gauche, le remplacer par le sous arbre droit
   (b) si le noeud n'a pas de fils droit, le remplacer par le sous arbre gauche
3. un noeud avec deux fils. Dans ce cas nous avons deux possibilités :
   (a) On remplace le noeud à supprimer par le noeud le plus à droite du fils gauche : clé immédiatement inférieure.
   (b) On remplace le noeud à supprimer par le noeud le plus à gauche du fils froit : clé immédiatement supérieure.

La suppression se réalise donc en trois étapes :

1. Localiser l'élément à supprimer
2. trouver l'éléments qui va prendre sa place
3. TRéaliser le remplacement

3.1.8 Exercices

3.2 Arbres équilibrés

3.2.1 Définition

Un arbre binaire équilibré, ou arbre AVL du nom des auteurs de la méthode : Adelson, Velskij, et Landis) est un arbre binaire tel que les hauteurs des deux sous-arbres de tout noeud de l'arbre diffèrent de 1 au plus .

3.2.2 Exemples et contre-exemples

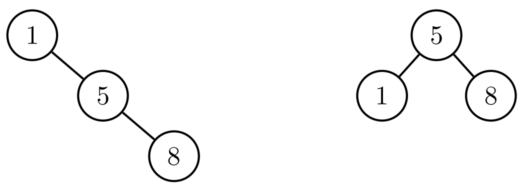
3.2.3 Intérêt

Une lecture infixée (SAG, Racine, SAD) donne de suite la liste triée.

3.2.4 Insertion

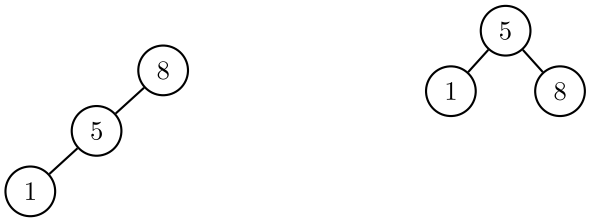
Inséré un élément risque de déséquilibré l'arbre. On se pose la question : "Au niveau de quel noeud est le déséquilibre ?"
On distingue 4 cas :

1. RR



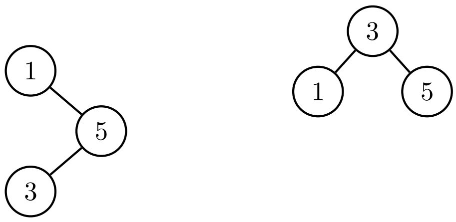
Equilibrage d'un arbre RR

2. LL



Equilibrage d'un arbre LL

3. RL



Equilibrage d'un arbre RL

4. LR

Equilibrage d'un arbre LR

3.2.5 Suppression

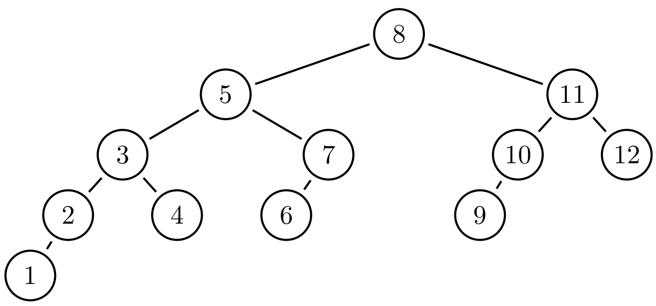


FIG. 3.1 –

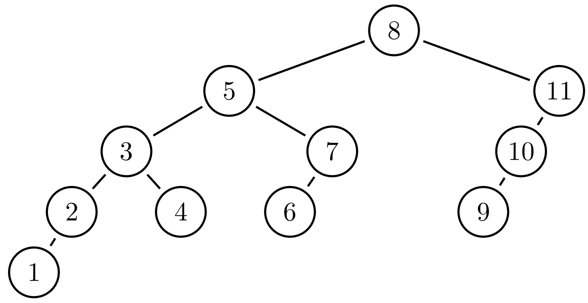


Fig. 3.2 –

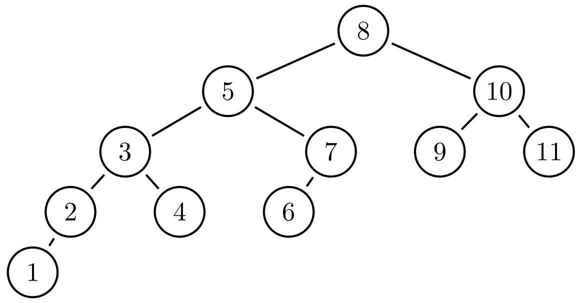


Fig. 3.3 –

3.2.6 Exercices

3.3 Tas

3.3.1 Définition

On appelle TAS un arbre ayant les caractéristiques suivantes :

1. Arbre binaire
2. Un noeud est supérieur à ses fils
3. Pour un noeud N, tous les noeuds d'étiquette inférieur existe.
   Ceci équivaut à dire que la dernière ligne est serrée à gauche. De plus il est équilibré.

Cet arbre est un maximier c'est à dire qu'il a la propriété d'être ordonné verticalement : chaque nœud contient une valeur supérieure à la valeur contenue dans ses descendants.

3.3.2 Exemples et contre-exemples

exemple : voir figure 3.5
Contre-exemple : voir figure 3.6

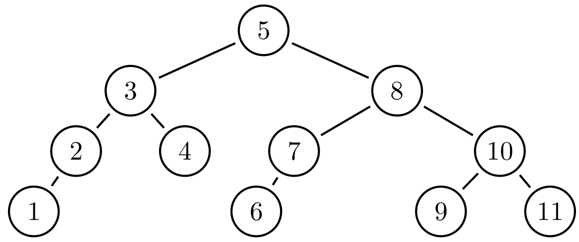


FIG. 3.4 –

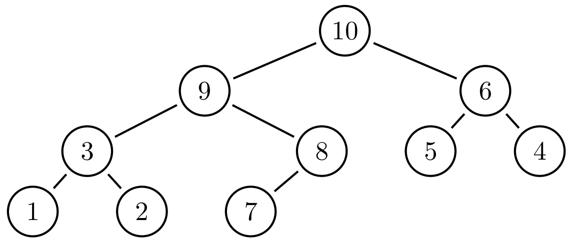


FIG. 3.5 – Un TAS

3.3.3 Intérêt

Un TAS peut etre codé dans un tableau. Les fils de i sont aux places 2*i et 2*i+1 si la numérotation des cellules débute à 1.
La hauteur est ainsi la plus petite possible. On a :

1. niveau 0 (racine) : 1 élément (2^0)
2. niveau 1 : 2 éléments : (2^1)
3. niveau 2 : 4 éléments (2^2)
4. niveau 3 : 8 éléments (2^3)

Ceci se traduit par le fait que la hauteur de l'arbre est en $log_2 n$. C'est la longueur la plus grande possible depuis la racine jusqu'a une feuille. C'est donc la borne supérieur du nombre d'étapes nécessaires pour effectuer une recherche.

3.3.4 Construction à partir d'un AB

Un tas à partir de deux tas

Cela se fait par échange et comparaison de clés. On échange la racine avec le fils ayant la plus grande clé. Ensuite on recommence avec les fils si nécessaire. cette opération s'appelle "Entasser"

3.3.5 En insérant

Le tri tas est un tri "en place", c'est-à-dire que les éléments sont insérés et triés dans un unique tableau. Après leur insertion dans le tableau, les éléments sont tout d'abord rangés "en tas", le tableau simulant ce tas. Un tas est un arbre binaire "parfait" : toutes les rangées de l'arbre sont pleines sauf éventuellement la dernière ; la dernière rangée est remplie de gauche à droite. Si on numérote

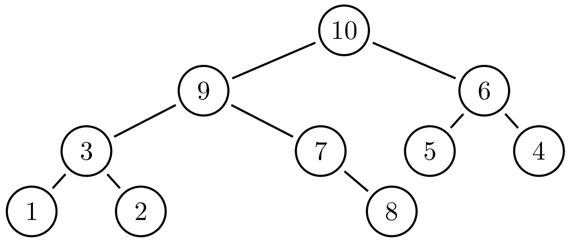


FIG. 3.6 – Ceci n'est pas un TAS

les sommets de cet arbre de gauche à droite, dans chaque rangée, et de haut en bas, la racine ayant le numéro 1, on voit que les fils du sommet numéroté i sont numérotés 2i et 2i+1. Il y a donc une bijection naturelle entre les noeuds d'un tel arbre et les positions d'indice i, pour i variant entre 1 et n dans le tableau. A chaque sommet de l'arbre on associe un des éléments à trier. Pour que l'arbre binaire parfait soit un tas, il faut qu'en chaque sommet l'élément qui s'y trouve soit plus grand que ceux situés en ses deux fils. Un algorithme possible de construction d'un tas est le suivant : on considère que l'on introduit les uns après les autres les éléments dans le tas, dont la structure est reconstituée après chaque insertion. Pour mettre à sa place l'élément en cours d'insertion, on l'introduit à la première place disponible, puis on le compare à son père : s'il est plus grand que son père, on l'échange avec son celui-ci ; on recommence ces échanges jusqu'à ce que l'élément à insérer soit plus petit que son père ou qu'il ait atteint la racine. Une fois le tas construit avec tous les éléments à trier, le maximum de ceux-ci se trouve à la racine.

A partir d'un AB déjà construit

On parcourt les noeuds ayant des fils en commençant par le plus bas à droite en appliquant chaque fois l'opération "Entasser".

3.3.6 Insertion

3.3.7 Suppression

3.3.8 Exercices

3.4 Tri arbre

3.4.1 Introduction

Imaginé par Williams en 1964, ce tri est aussi appelé tri en tas ou tri maximier. Ce tri est une accélération du tri par sélection et échange : une organisation préalable de l'ensemble de départ permet d'obtenir une sélection du maximum en un temps qui est dans l'ordre de log2n, ce qui permet à l'algorithme de redevenir optimum.

On peut utiliser le tas pour trier les éléments : on retire la racine, on restaure la structure de tas sur les éléments restants et on recommence. Pour restaurer

la structure de tas, on place le dernier élément du tas à la racine (à la place de l'élément qu'on vient de retirer pour construire la liste triée ; en fait, pour ne pas consommer de place supplémentaire, on échange la racine du tas courant et le dernier élément du tas courant, et on considère que le tas courant est maintenant l'ancien tas courant privé de son dernier élément), puis on le fait "descendre" dans le tas en l'échangeant avec le plus grand de ses fils, si le plus grand de ses fils est plus grand que lui. Quand on a fini, le tableau contient les éléments dans l'ordre croissant.
On donne une suite.

1. Construire un AB équilibré
2. Transformer cet arbre en maximier ou TAS
3. Travailler ce maximier
4. Réaliser une lecture infixée de ce nouvel arbre.

3.4.2 Exemple

- Soit la suite à trier : 7, 3, 9, 5, 6, 8.
- Construction d'un AB : Voir figure 3.7
- Transformer cet AB en Maximier :
- Réaliser le tri à partir du Maximier
    1. Etape 1 : Le maximier de départ. Voir figure 3.8
    2. Etape 2 : Nouvel arbre partiel. 9 est à sa place. Voir figure 3.9
    3. Etape 3 : Construction du nouveau maximier. Voir figure 3.10
    4. Etape 4 : Nouvel arbre partiel. 8 et 9 sont à la bonne place. Voir figure 3.11
    5. Etape 5 : Construction du nouveau maximier. Voir figure 3.13
    6. Etape 6 : Construction du nouveau maximier. Voir figure 3.14
    7. Etape 7 : Construction du nouveau maximier. Voir figure 3.15

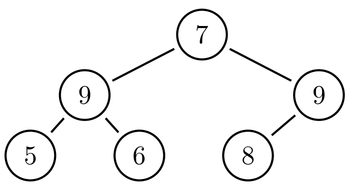


Fig. 3.7 – AB à partir de la suite

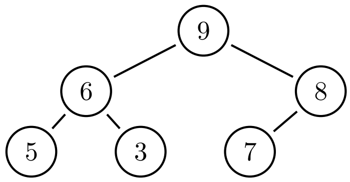


FIG. 3.8 – Maximier de départ

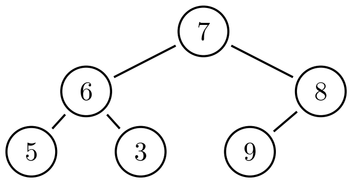


FIG. 3.9 – Etape 1

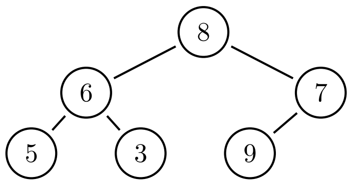


FIG. 3.10 – Etape 2 construire nouveau maximier

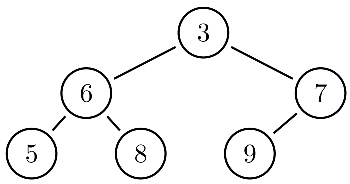


FIG. 3.11 – Etape 3

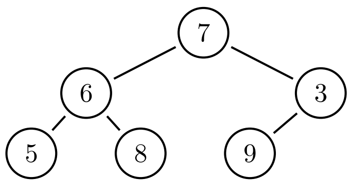


FIG. 3.12 – Etape 4

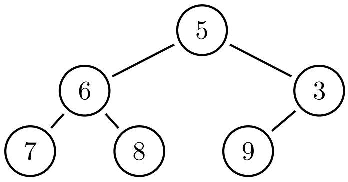


FIG. 3.13 – Etape 5

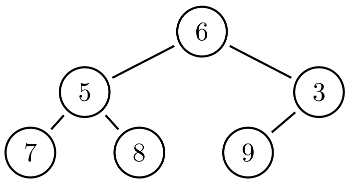


FIG. 3.14 – Etape 6

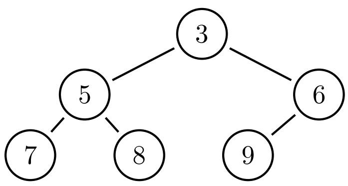


FIG. 3.15 – Etape 7

Chapitre 4

Programmation linéaire

4.1 Introduction

4.1.1 Enoncé

Un agriculteur peut utiliser 2 type d'engrais X et Y pour épandre sur ses cultures. Les besoins par an et par hectare de 60 kg de potasse, 120 kg de calcium et 90 kg de nittrates.
Pour une meme quantité, les 2 types d'engrais coutent la meme chose. Leur composition pour 10 kg est de :
produit X : 1 kg de potasse, 3 kg de calcium, 3 kg de nitrates et 3 kg de produit neutre
produit Y : 2 kg de potasse, 2 kg de calcium, 1 kg de nitrates et 5 kg de produit neutre
Question : Comment fertiliser les cultures à moindre cout ?

4.1.2 Mise en forme

Ce problème comporte 2 inconnues, et , qui constituent respectivement les quantités d'engrais X et Y à utiliser par an et par hectare.
Les contraintes peuvent etre écrites sous la forme suivante :

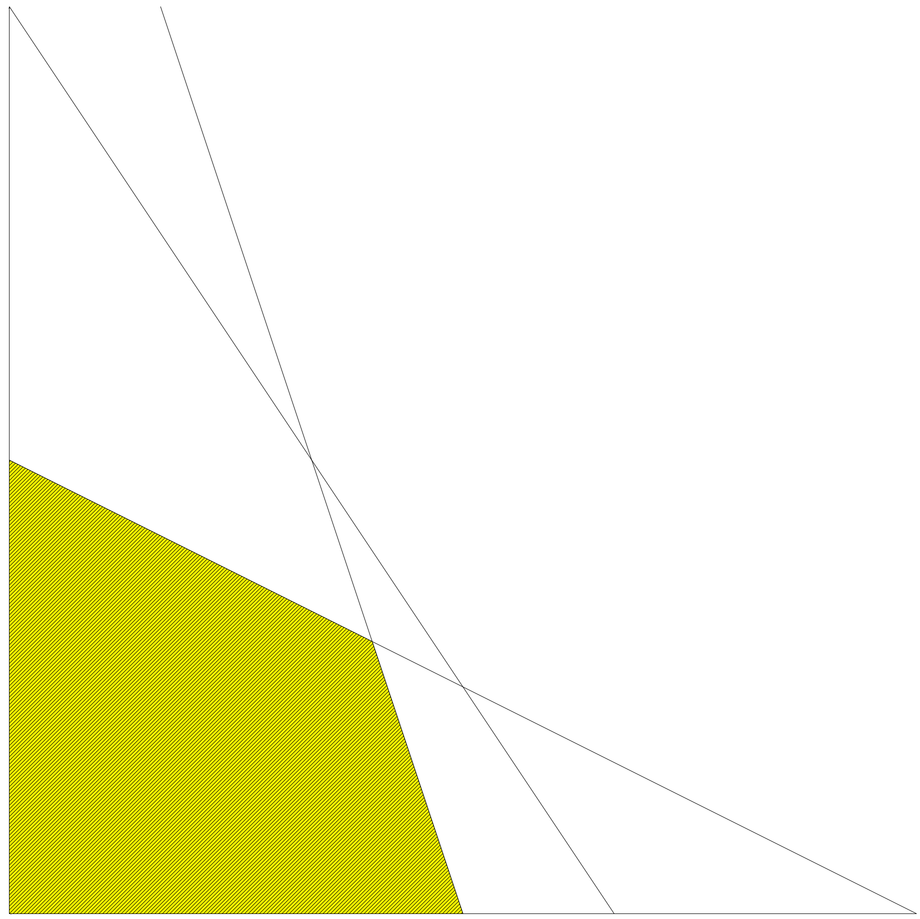
$$\begin{cases} x + 2y \geq 60 \\ 3x + 2y \geq 120 \\ 3x + y \geq 90 \end{cases}$$

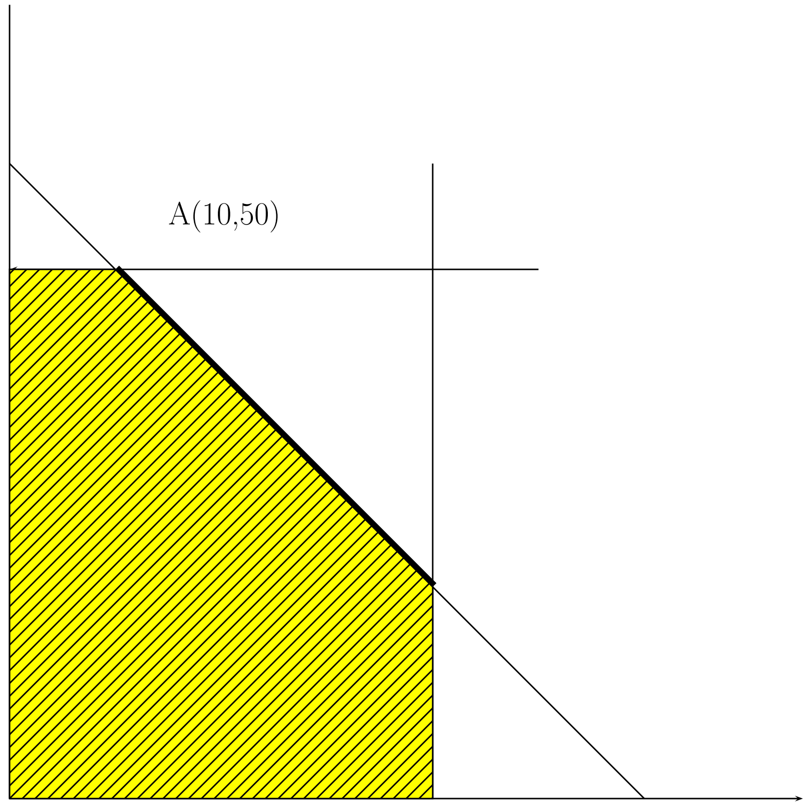
4.1.3 Solution graphique

4.2 Un problème

4.2.1 Enoncé

Un entrepreneur utilise 60 tonnes de béton par jour. Il passe commande à deux centrales de béton : l'une peut lui fournir au maximum 40 tonnes par jour, l'autre 50 tonnes. Les prix de transport sont de 50 BEF par tonne pour la première centrale et de 40 BEF par tonne pour la seconde. Le béton coute 1210 bef la tonne





à la la première centrale et 1160 BEF la tonne à la seconde.
Combien de béton doit-il commander à chaque centrale pour que le prix de revinet soit minimum ?

4.2.2 Solution graphique

La ligne grasse indique l'ensemble des solutions pour le système des contraintes. Maintenant, il faut minimiser la fonction qui nous donne le prix de revient.
On a

$$F = (1210 + 50)x_1 + (1160 + 40)x_2$$

ou encore

$$F = 21x_1 + 20x_2$$

4.2.3 Remarque

Une solution graphique est possible lorsque nous avons affaire à un problème linéaire à deux variables. Au-delà, il faut trouver une solution algébrique.

4.3 Un autre problème

Soit a résoudre le système suivant :

$$\begin{cases} x_1 + x_2 \leq 10 \\ 3x_1 + x_2 \leq 24 \\ 2x_1 + 5x_2 \leq 38 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

On demande :

1. de maximiser la fonction $F = 2x_1 + x_2$
2. Refaites le même exercice avec la fonction $F = x_1 + 9x_2$

Les réponses sont :

1. Pour $F = 2x_1 + x_2$: R=17 (le point B)
2. Pour $F = 6x_1 + 9x_2$: R=58 (le point C)

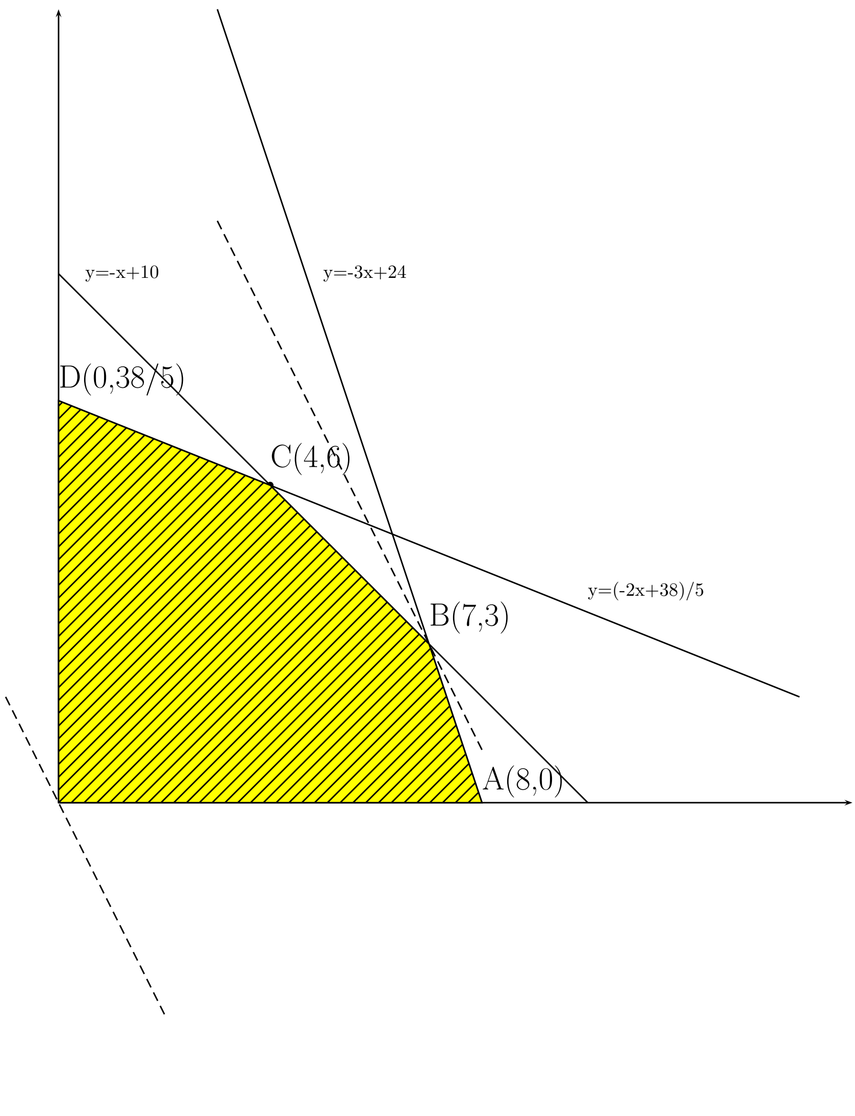
4.4 Résolution

4.4.1 Rappel

Droite et demi-plan

4.4.2 Démarche

1. Tracer les cinq droites
2. Trouver les demi-plans solution du système
3. Trouver le polygone des solutions
4. Tracer la droite $2x_1 + x_2$
5. Trouver le point ou le coté solution


y=-x+10
y=-3x+24
D(0,38/5)
C(4,6)
y=(-2x+38)/5
B(7,3)
A(8,0)

Chapitre 5

Solutions

5.1 Exercice 1

On a :

$$\mathcal{M} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} ; \mathcal{M}^2 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} ; \mathcal{M}^3 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

on a $\widehat{\mathcal{M}} = \mathcal{M} + \mathcal{M}^2 + \mathcal{M}^3$ avec :

$$\widehat{\mathcal{M}} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

On a aussi :

$$\mathcal{I}+\mathcal{M} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} ; (\mathcal{I}+\mathcal{M})^2 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} ; (\mathcal{I}+\mathcal{M})^4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} ;$$

La matrice est stable !
On constate que les sommets {1,2,3} forment une composante fortement connexe

5.2 Exercice 2

On a :

$$\mathcal{M} = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

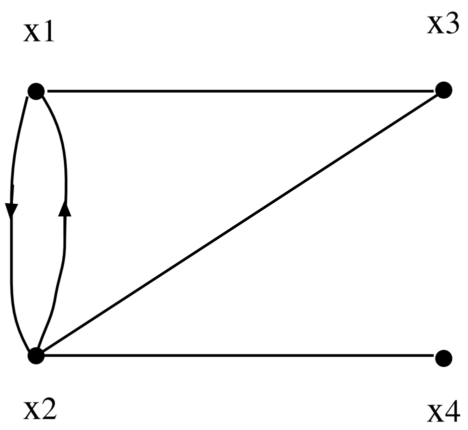


FIG. 5.1 –

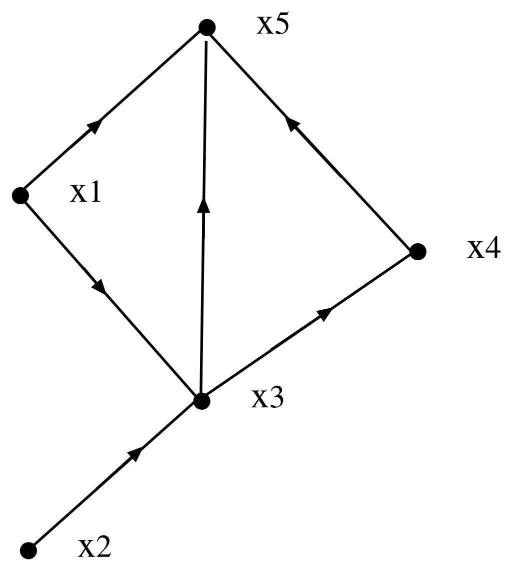


FIG. 5.2 –

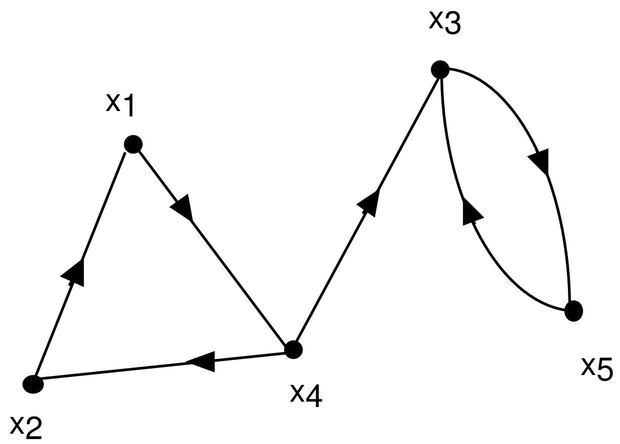


FIG. 5.3 –

et aussi

$$\mathcal{I}et\mathcal{M} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} ; (\mathcal{I}et\mathcal{M})^2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} ; (\mathcal{I}et\mathcal{M})^4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

On voit que :

$$\widehat{\mathcal{M}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

5.3 Exercice 3

On a :

$$\mathcal{M} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

On voit que :

$$\widehat{\mathcal{M}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Il y a ici deux composantes fortement connexes

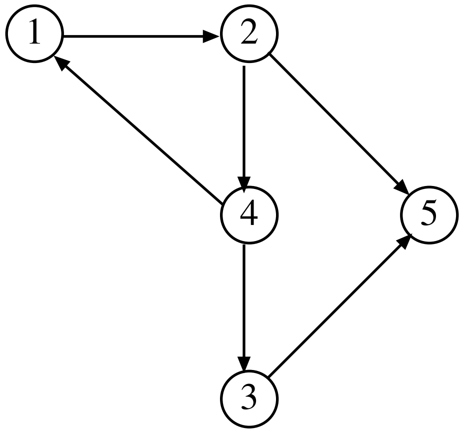


FIG. 5.4 –

5.4 Exercice 4

On demande :

1. Ce graphe est-il complet ?
2. La matrice d'adjacence
3. La matrice d'accessibilité
4. Les composantes fortement connexes
5. Une exploration
   – en largeur en partant du sommet 1, puis en partant du sommet 2
   – en longueur
6. Décomposer en niveaux

1. Ce graphe n'est pas complet
2. La matrice d'adjacence : On a :

$$\mathcal{M} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

3. La matrice d'accessibilité : On a :

$$\widehat{\mathcal{M}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

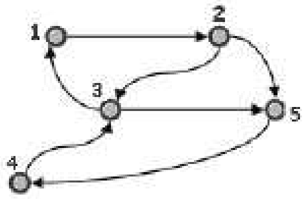
4. Les composantes fortement connexes : On a : {1,2,4}, {3} et {5}
5. Une exploration
   – en largeur en partant du sommet 1 : 1,2,4,5,3

– en largeur en partant du sommet 2 : 2,4,5,1,3
– en profondeur préfixé : 1,2,4,3,5

6. Décomposer en niveaux :
   Impossible car il y a un circuit.
   Si on supprime l'arc (1,2), on n'a plus de circuite et la décomposition donne :
   niveau 1 : 2
   niveau 2 : 4
   niveau 3 : 1 et 3
   niveau 4 : 5.

Exercice : Graphe et calcul matriciel

Soit le graphe suivant :



1° donnez la matrice d'adjacence.
2° calculez la fermeture transitive de 2 façons différentes.
3° quels sont les champs fortement connexes.
4° est ce que 5 est accessible depuis 2 par 1 chemin de longueur 4 ?

1° Matrice d'adjacence :

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

On peut remarquer dans ce cas précis que la matrice d'adjacence sera identique à la matrice booléenne, car il n'y a jamais plus d'un arc allant d'un sommet à un autre.

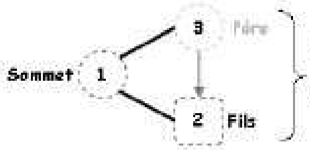
2° Fermeture transitive :

a) par calcul des puissances :

$$\mathfrak{M}^2=\begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \mathfrak{M}^3=\begin{pmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \mathfrak{M}^4=\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \hat{\mathfrak{M}}=\mathfrak{M}+\mathfrak{M}^2+\mathfrak{M}^3+\mathfrak{M}^4=\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

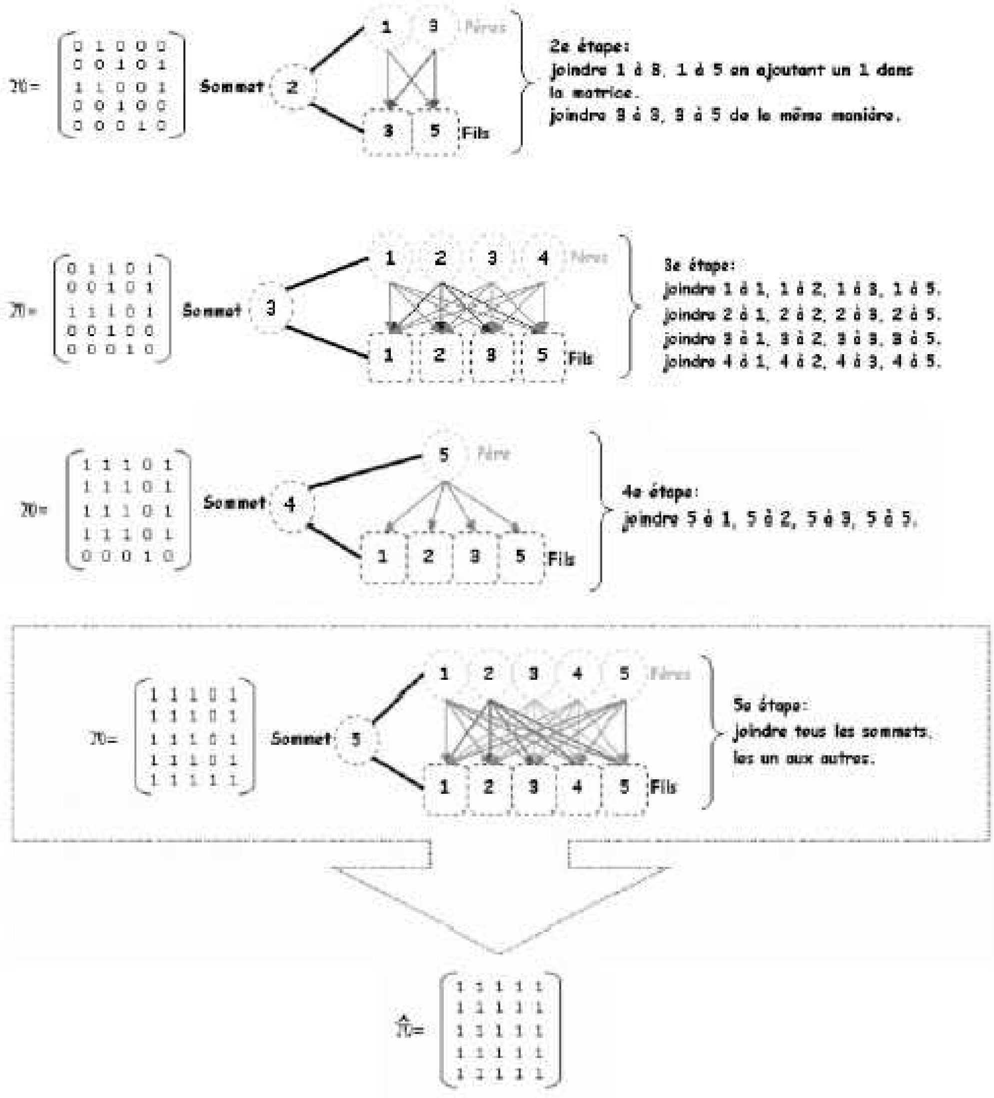
b) par l'algorithme de Warshall :

$$\mathfrak{M}=\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$



1e étape de transformation de la matrice: joindre le père (3) au fils (2), ajouter un 1 3e ligne 2e colonne.

Fig. 5.5 –



3° champs fortement connexes :

En observant la matrice de fermeture transitive, on constate que les lignes de 1 à 5 sont identiques donc les champs fortement connexes sont **{1, 2, 3, 4, 5}**.

4° Existe-t-il un chemin de longueur 4 partant du sommet (2) vers le sommet (5) ?

$$M^4 = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & \boxed{1} \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix} \rightarrow \text{OUI}$$

FIG. 5.6 –

Chapitre 6

Le simplexe

6.1 Principe

Le principe de l'algorithme consiste à améliorer petit à petit une solution de base située à l'origine du repère, en se déplacant le long des aretes.

6.2 Exemple 1

6.2.1 Forme mathématique

Partons d'une situation simple à deux variables afin de pouvoir en faire la vérificaton graphique. Soit le systeme d'inéquations

$$\begin{cases} x_1 + x_2 \leq 4 \\ -x_1 + 2x_2 \leq 6 \\ x_1 - x_2 \leq 5 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

on cherche à maximiser le fonction $F = 3x_1 + x_2$

6.2.2 Mise sous forme d'équations

On obtient

$$\begin{cases} x_1 + x_2 + e_1 = 4 \\ -x_1 + 2x_2 + e_2 = 6 \\ x_1 - x_2 + e_3 = 5 \\ x_1 \geq 0 \\ x_2 \geq 0 \\ e_1 \geq 0 \\ e_2 \geq 0 \\ e_3 \geq 0 \end{cases} \qquad (*)$$

Les nouvelles variables doivent etre positives et avoir comme coefficient 1.

6.2.3 Une première solution

On part du sommet (0,0).
On voit que $x_1 = x_2 = 0$ est une solution. qui est telle que F=0. En fait la solution complète est $x_1 = x_2 = 0$, $e_1 = 4, e_2 = 6$ et $e_3 = 5$. Elle correspond à un "coin" du polygone des solutions

6.2.4 Améliorer F

Se déplacer le long d'une arête à partir de (0,0) signifie q'une des deux variables nulles va quitter cette valeur. On fait délibérément le choix de prendre la valeur qui va faire croître F le plus vite possible.
C'est x_1 qui fait croitre F le plus rapidement possible. On va donc faire varier x_1. Soit $x_1 = \theta$ et $x_2 = 0$. On a alors :

$$\left\{\begin{array}{l} e_1 = 4 - \theta \geq 0 \\ e_2 = 6 + \theta \geq 0 \\ e_3 = 5 - \theta \geq 0 \end{array}\right. \Rightarrow \left\{\begin{array}{l} \theta \leq 4 \\ \theta \geq -6 \\ \theta \leq 5 \end{array}\right. \Rightarrow 0 \leq \theta \leq 4$$

La plus grande valeur possible pour θ est donc 4. Pour cette valeur on est alors sur un autre sommet. On a alors comme seconde solution

$$e_1 = 0,\ e_2 = 10,\ e_3 = 1,\ x_1 = 4,\ x_2 = 0$$

et

$$F = 12$$

6.2.5 Encore améliorer F

On va exprimer F en fonction des valeurs nulles, puis voir celles que l'on peut augmenter. On a en partant de (*)

$$F = 3x_1 + x_2 = 3(4 - x_2 - e_1) + x_2 = 12 - 2x_2 - 3e_1$$

On constate que les coefficients sont négatifs et dès lors F ne peut plus augmenter. Le maximum de F est donc 12.

6.3 Exemple 2

6.3.1 Le problème sous forme mathématique

Soit le systeme d'inéquations

$$\begin{cases} -2x_1 + x_2 \leq 1 \\ x_1 + 3x_2 \leq 10 \\ x_1 \leq 4 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

on cherche à maximiser le fonction $F = -x_1 + x_2$

6.3.2 Mise sous forme d'équations

On obtient

$$\begin{cases} -2x_1 + x_2 + e_1 = 1 \\ x_1 + 3x_2 + e_2 = 10 \\ x_1 + e_3 = 4 \\ x_1 \geq 0 \\ x_2 \geq 0 \\ e_1 \geq 0 \\ e_2 \geq 0 \\ e_3 \geq 0 \end{cases}$$

Les nouvelles variables doivent etre positives et avoir comme coefficient 1.

6.3.3 Une première solution

On voit que $x_1 = x_2 = 0$ est une solution. qui est telle que F=0. En fait la solution complète est $x_1 = x_2 = 0$, $e_1 = 1, e_2 = 10$ et $e_3 = 4$

6.3.4 Améliorer F

C'est x_2 qui fait croître F le plus rapidement possible. On va donc faire varier x_2. Soit $x_2 = \theta$ et $x_1 = 0$. On a alors :

$$\begin{cases} e_1 = 1 - \theta \geq 0 \\ e_2 = 10 - 3\theta \geq 0 \end{cases} \Rightarrow \begin{cases} \theta \leq 1 \\ \theta \leq \frac{10}{3} \end{cases} \Rightarrow 0 \leq \theta \leq 1$$

La plus grande valeur possible pour θ est donc 1. On a alors comme solution

$$e_1 = 0,\ e_2 = 7,\ e_3 = 4,\ x_1 = 0,\ x_2 = 1$$

La valeur de F est $F = 1$

6.3.5 Encore améliorer F

On va exprimer F en fonction des valeurs nulles, puis voir celle que l'on peut augmenter. On a

$$F = x_1 - e_1 + 1$$

. cette expression est obtenue en partant de $F = -x_1 + x_2 = -x_1 + (1 - e_1 + 2x_1)$ (Voir système de départ) ; On constate que le seul coefficient positif est celui de x_1. Il faut donc faire varier x_1.
Pour reprendre le raisonnement, il faut exprimer les variables non nulles, dans la dernière solution (x_2, e_2 et e_3) en fonction des variables nulles ($e_1 = x_1 = 0$).
On obtient :

$$\begin{cases} -2x_1 + x_2 + e_1 = 1 \\ 7x_1 - 3e_1 + e_2 = 7 \\ x_1 + e_3 = 4 \end{cases} \Rightarrow \begin{cases} x_2 = 1 - \theta \geq 0 \\ e_2 = 7 - 7\theta \geq 0 \\ e_3 = 4 - \theta \geq 0 \end{cases} \Rightarrow 0 \leq \theta \leq 1$$

La plus grande valeur possible pour θ est donc 1. Une nouvelle solution est donc :

$$x_1 = 1, x_2 = 3, e_1 = 0, e_2 = 0, e_3 = 3$$

La nouvelle valeur de F est donc : $F = 2$.

6.3.6 Nouvelle amélioration de F

Pour savoir si l'on peut amélioere encore F, il faut exprimer F en fonction des variables nulles. On obtient :

$$F = 2 - \frac{4e_1}{7} - \frac{e_2}{7}$$

on constate que ce n'est plus possible puisque tous les coeffiicents sont négatifs.

6.3.7 En résumé...

Les solutions successives sont donc :

$x_1 =$	0	0	1
$x_2 =$	0	4	3
$e_1 =$	1	0	0
$e_2 =$	10	7	0
$e_3 =$	4	4	3
$F =$	0	1	2

On voit qu'une des deux variables nulles le reste et que l'autre devient positive.

6.4 Mise en forme pour l'algorithme : exemple 1

Partons du système (*).
On a

	e_1	e_2	e_3	x_1	x_2	d
e_1	1	0	0	$\boxed{1}$	1	4
e_2	0	1	0	-1	2	6
e_3	0	0	1	1	-1	5
	0	0	0	3	1	F
				P		

Constatons que, dans la solution de départ, e_1, e_2 *et* e_3 sont les variables non nulles. Elles sont regroupées. De meme, x_1 *et* x_2 sont les variables nulles et sont aussi regfroupées.
Détermination de la colonne pivot :
C'est la colonne contenant le plus grand coefficient positif pour F
Détermination de la ligne pivot :
Divisons les valeurs de la colonne d par ceux de la colonne pivot. On obtient : 4, -6 et 5. Parmi ces valeurs, on cherche la plus petite des valeurs positives. C'est x_1 qui va devenir positive. Quelle variable va devenir nulle ? On constate que c'est e_1. On permutte alors les colonnes e_1 et x_1. On obtient alors :

	x_1	e_2	e_3	e_1	x_2	d
x_1	1	0	0	1	1	4
e_2	-1	1	0	0	2	6
e_3	1	0	1	0	-1	5
	3	0	0	0	1	F

on calcul l'inverse de la matrice des variables non nulles :

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

On obtient

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

On multiplie le tableau par cette matrice. Alors le tableau devient :

x_1	e_2	e_3	e_1	x_2	d
1	0	0	1	1	4
0	1	0	1	3	10
0	0	1	-1	-2	1
3	0	0	0	1	F
0	0	0	-3	-2	F-12

La ligne relative à F se transforme après avoir fait la multiplication matricielle et cette transformation se réalise comme suit :
F ne peut etre exprimée qu'en fonction des variables nulles. On va donc sosutraire de la ligne F trois fois la ligne relative à x_1.
Les deux coefficients de F étant négatifs, on ne peut plus améliorer la valeur de F. Celle-ci est donc 12.

6.5 Mise en forme : exemple 2

On a

	e_1	e_2	e_3	x_1	x_2	d
e_1	1	0	0	-2	$\boxed{1}$	1
e_2	0	1	0	1	3	10
e_3	0	0	1	1	0	4
	0	0	0	-1	1	F
					P	

La colonne pivot : C'est x_2 qui va devenir positive.
La ligne pivot : Quelle variable va devenir nulle ? On constate que c'est e_1.
On permutte donc les colonnes e_1 et x_2. On obtient

	x_2	e_2	e_3	x_1	e_1	d
x_2	1	0	0	-2	1	1
e_2	3	1	0	1	0	10
e_3	0	0	1	1	0	4
	1	0	0	-1	0	F

On calcul l'inverse de la matrice

$$\begin{pmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

On obtient

$$\begin{pmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Le tableau devient :

	x_2	e_2	e_3	x_1	e_1	d
x_2	1	0	0	-2	1	1
e_2	0	1	0	$\boxed{7}$	-3	7
e_3	0	0	1	1	0	4
	1	0	0	-1	0	F
	0	0	0	1	-1	F-1
				P		

Ici, la colonne pivot est celle de x_1 et la ligne pivot est celle de e_2. permuttons les colonnes x_1 et e_2. On peut tenter d'améliorer F en augmentant x_1. On obtient

	x_2	x_1	e_3	e_2	e_1	d
x_2	1	-2	0	0	1	1
x_1	0	7	0	1	-3	7
e_3	0	1	1	0	0	4
	0	1	0	0	-1	F-1

L'inverse de $\begin{pmatrix} 1 & -2 & 0 \\ 0 & 7 & 0 \\ 0 & 1 & 1 \end{pmatrix}$ est $\begin{pmatrix} 1 & \frac{2}{7} & 0 \\ 0 & \frac{1}{7} & 0 \\ 0 & \frac{-1}{7} & 1 \end{pmatrix}$

On effectue la multiplication suivante :

$$\begin{pmatrix} 1 & \frac{2}{7} & 0 \\ 0 & \frac{1}{7} & 0 \\ 0 & \frac{-1}{7} & 1 \end{pmatrix} . \begin{pmatrix} 1 & -2 & 0 & 0 & 1 & 1 \\ 0 & 7 & 0 & 1 & -3 & 7 \\ 0 & 1 & 1 & 0 & 0 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & -\frac{2}{7} & \frac{1}{7} & 3 \\ 0 & 1 & 0 & \frac{1}{7} & -\frac{3}{7} & 1 \\ 0 & 0 & 1 & -\frac{1}{7} & \frac{3}{7} & 3 \end{pmatrix}$$

Le tableau devient :

	x_2	x_1	e_3	e_2	e_1	d
x_2	1	0	0	$-\frac{2}{7}$	$\frac{1}{7}$	3
x_1	0	1	0	$\frac{1}{7}$	$-\frac{3}{7}$	1
e_3	0	0	1	$-\frac{1}{7}$	$\frac{3}{7}$	3
	0	1	0	0	-1	F-1
	0	0	0	$-\frac{1}{7}$	$-\frac{4}{7}$	F-2

Les deux coefficients sont négatifs. On constate que nous obtenons bien les mêmes coefficients que ceux obtenus précedemment. On ne peut plus améliorer F. Son maximum est donc 2.

6.6 Les étapes

1. Ecrire les inéquations de contraintes et la fonction économique
2. Transformer les inéquations de contraintes en équations. Il faut introduire des variables d'écart. Mettre sous forme matricielle.
3. Choisir la variable de la colonne pivot. C'est la variable dont le coefficient de la fonction économique est le plus grand positif.
4. Déterminer la variable de la ligne pivot. Faire la permutation des deux colonnes.
5. Confectionner le nouveau tableau.
   Calculer l'inverse de la matrice des variables non nulles.
   Multiplier la matrice du tableau par l'inverse de la matrice.
6. transformer la dernière ligne (celle contenant F) pour avoir des coefficients nuls dans la partie gauche.

Chapitre 7

Le voyageur de commerce

7.1 Introduction

7.1.1 Parcours particuliers

Parcours Hamiltonien

Un parcours est hamiltonien s'il passe une et une seule fois par chaque sommet du graphe. Le problème du voyageur de commerce.

Parcours Eulérien

Un parcours est Eulérien s'il passse exactement une fois par chaque arete ;
Exemple 1 : les sept ponts de Konigsberg.
Exemple 2 : le problème du postier chinois.

7.1.2 Le voyageur de commerce

Il s'agit de

1. passer une seule fois par tous les sommets d'un graphe
2. de minimiser la longueur du parcours.

Si on souhaite examiner toutes les solutions et ensuite sélectionner la meilleure, il faudra évaluer n ! solutions si le graphe possède n sommets. Ceci est impossible.

7.1.3 Séparation et évaluation

Axiomes

La suite des sous problèmes engendrés par séparation est finie
Tout problème terminal (on ne sait plus séparé) est vide ou est une solution minimale.
Il faut donc :
Un principe de séparation
Une fonction d'évaluation.

7.1.4 Algorithme de Little

On a un graphe complet.
Le graphe est représenté par une matrice.
On effectue un parcours en profondeur en attribuant à chaque noeud

1. une évaluation par défaut
2. une valeur de regret à chaque choix auquel on renonce

La réduction

Cette réduction ne change pas le problème.
On réduit les lignes **puis** les colonnes
Il faut donc un zéro par ligne et par colonne
La somme des réduction nous donne une borne inférieure et constitue donc l'évaluation par défaut de la racine..

La séparation

Choisir un arc qui a le maximum de chance d'être dans le circuit recherché.

7.1.5 Exemple

Les données

Soit un graphe dont la matrice du poids des arcs est le tableau

	1	2	3	4	5	6
1	∞	27	43	16	30	26
2	7	∞	16	1	30	25
3	20	13	∞	35	5	0
4	21	16	25	∞	18	18
5	12	46	27	48	∞	5
6	23	5	5	9	5	∞

On constate que ce graphe est *orienté , complet et non symétrique !*

Initialisation

Réduction des lignes :

	1	2	3	4	5	6	
1		11	27	0	14	10	16
2	6		15	0	29	24	1
3	20	13		35	5	0	0
4	5	0	9		2	2	16
5	7	41	22	43		0	5
6	18	0	0	4	0		5

Réduction des colonnes

	1	2	3	4	5	6
1		11	27	0	14	10
2	1		15	0	29	24
3	15	13		35	5	0
4	0	0	9		2	2
5	2	41	22	43		0
6	13	0	0	4	0	
	5	0	0	0	0	0

Les deux réductions nous donne la valeur 43+5=48

Premier choix

Pour chaque arc nul, j'évalue ce qui se passe en passant et puis en ne passant pas par cet arc. **La séparation va consister à choisir un arc tel que son évitement coute le plus cher**.
Si on ne passe pas par l'arc (a,b), il faut quitter le sommet "a" sans aller vers "b". Donc le cout sera au moins égal au minimum des poids des arcs quittant "a" en écartant l'arc (a,b). Il faut donc considérer les **successeurs** de "a" à l'exception de l'arc(a,b) c'est-à-dire justement l'arc de poids 0. De même, il faut aboutir en "b" sans venir de "a". Donc je dois considérer les **prédecesseurs** de b à l'exception de "a". Pour obtenir ce cout, on additionne la valeur minimale de la ligne et la valeur minimale de la colonne se croisant à l'arc en question. Le trajet coutera de toute facon au moins 48. C'est ce qu'il faut au moins parcourir comme longueur dans le graphe.
On a

	1	2	3	4	5	6
1		11	27	0 (10)	14	10
2	1		15	0 (1)	29	24
3	15	13		35	5	0 (5)
4	0 (1)	0 (0)	9		2	2
5	2	41	22	43		0 (2)
6	13	0 (0)	0 (9)	4	0 (2)	

On choisi d'examiner l'arc a(1,4).
Ne pas passer par cet arc nous coute donc 48+10=58. **Pour connaitre le cout en prenant cet arc, il faut de nouveau réduite la matrice ;** mais avant, on supprime la ligne 1 et la colonne 4. En effet, supprimer la ligne 1, c'est ne plus considérer les successeurs de 1.
Supprimer la colonne 4, c'est ne plus considérer les prédecesseurs de 4. On obtient :

	1	2	3	5	6
2	1		15	29	24
3	15	13		5	0
4	∞	0	9	2	2
5	2	41	22		0
6	13	0	0	0	

On a placé ∞ pour l'arc a(4,1) afin de ne pas créer un circuit.
la nouvelle réduction nous donne :

	1	2	3	5	6
2	0 (16)		14	28	23
3	15	13		5	0 (0)
4	∞	0 (2)	9	2	2
5	2	41	22		0 (2)
6	13	0 (0)	0 (9)	0 (2)	

La somme des réductions est 1.
En passant par l'arc (1,4) le cout est donc 49. En ne passant pas il est de 58.
Donc on passe par l'arc (1,4). !

Second choix

On constate ici que c'est l'arc (2,1) qui va faire l'objet d'une analyse.
Ne pas passer par cet arc nous coute 16. En tout, le cout sera de 49+16=65. Pour connaitre le cout du passsage par cet arc, il faut de nouveau réduire la matrice après avoir supprimé la ligne 2 et la colonne 1. Ici, nous ne pouvons avoir de circuit (1,2,1), car l'arc (1,2) ne fait pas partie de la matrice. Par contre, l'arc(1,4) ayant été retenu, si je retiens aussi l'arc(2,1), je dois absolument excure l'arc (4,2) pour éviter le circuit (1,4,2,1). **On place donc ∞ à l'élément (4,2) du tableau**.

	2	3	5	6
3	13		5	0
4	∞	9	2	2
5	41	22		0
6	0	0	0	

Après réduction on obtient :

	2	3	5	6
3	13		5	0 (5)
4	∞	7	0 (0)	0 (0)
5	41	22		0 (22)
6	0 (13)	0 (7)	0 (0)	

Cette réduction nous fournit la somme 2.
Donc le passage par l'arc (2,1) donne 49+2=51. son évitement nous donne 49+16=65.
On constate ici que le cout minimal est obtenu en passant par (1,4) puis par (2,1). On continue donc dans cette branche.

Choix suivant (3)

On examine l'arc (5,6). Son evitement coute 22.
Le cout du passage par (5,6) est obtenu apres réduction de la matrice obtenue en supprimant la ligne 5 et la colonne 6. On obtient apres suppression et réduction :

	2	3	5
3	8		0 (8)
4	∞	7	0 (7)
6	0 (8)	0 (7)	0 (0)

Cette réduction a couté 5.
le passage par (5,6) coute donc 5 et au total : 5+51=56 et son evitement 22 et au total 22+51=73.
La valeur la plus basse est donc 56. On passe donc par l'arc(5,6).

Choix suivant (4)

On examine l'arc (3,5). Son evitement coute 8. Pour le cout par son passage, on réduit la matrice apres avoir supprimé la ligne et la colonne adéquates.
ici, ayant retrenu l'arc (5,6) et examinant l'arc(3,5), **il faut exclure l'arc(6,3) pour ne pas avoir le circuit (3,5,6,3)**.

	2	3
4	∞	0 (∞)
6	0 (∞)	∞

La réduction coute 7.
Le passage par (3,5) coutera au total : 7+56=63.
L'evitement de (3,5) coute 56+8=64.
Dans tous les cas le cout minimal est maintenant 58 (voir au début lorsque l'on ne passe pas par l'arc(1,4)). On reprend avec cette branche. On retourne donc au tableau adéquat qui est ici celui de départ après réduction initiale. A cette fin on place ∞ dans l'élément (1,4) de la matrice. On réduit la ligne et la colonne passant par cet élément. On obtient après réduction :

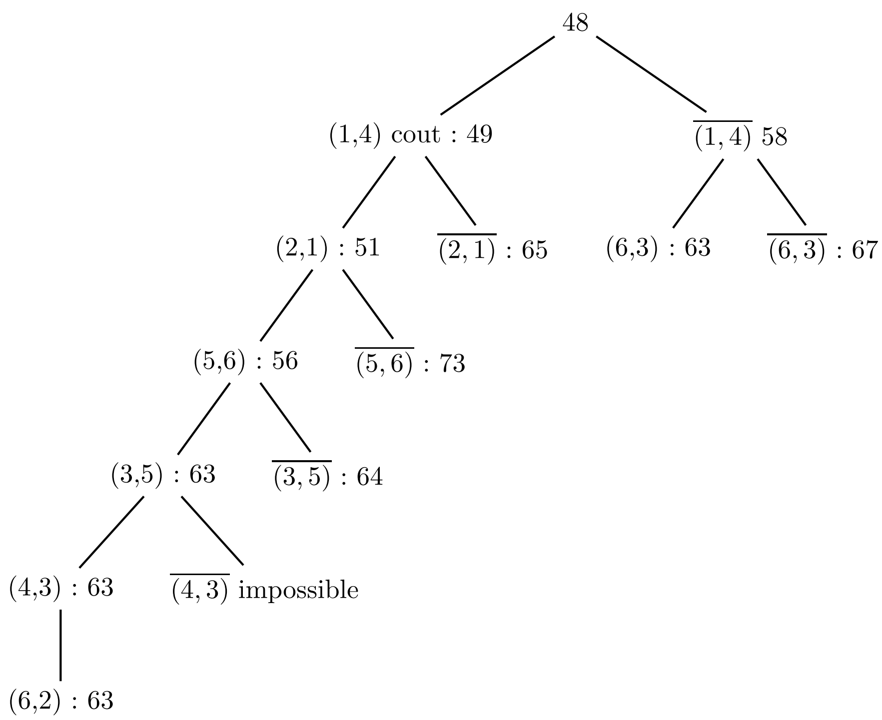
	1	2	3	4	5	6
1		1	17	∞	4	0
2	1		15	0 (1)	29	24
3	15	13		35	5	0 (5)
4	0 (1)	0 (0)	9		2	2
5	2	41	22	43		0 (2)
6	13	0 (1)	0 (9)	4	0 (2)	

On examine l'arc (6,3). Son évitement coute 9. Donc au total : 58+9=67.
Le passage par l'arc (6,3) coute 5. En effet, il faut interdire le passage par (3,6) en mettant ∞ dans la matrice obtenue en supprimant la ligne 6 et la colonne 3. La réduction donne alors 5. Donc au total : 5+58=63

Choix suivant (5)

On constate que le passage par l'arc (6,3) dans la suite [non(1,4), (6,3)] coute 63 et que la passge par (3,5) dans la suite [(1,4),(2,1), (5,6), (3,5)] coute aussi 63. On selectionne arbitrairement de poursuivre dans la phase précédente, c'est-à-dire celle où l'on passe par (3,5) car elle est la plus proche de la fin du parcours. On reconsidère donc le tableau dans "choix suivant (4)" On a le choix entre l'arc(4,3) et l'arc(6,2).
On examine l'arc (4,3) arbitrairement. On doit passer par cet arc. En effet, son évitement est impossible car il couterait ∞. Le passage par cet arc coute 0 car la matrice est réduite de fait.
Il reste l'arc(6,2) dont le cout est 0 puisque la réduction de cette matrice 1*1

est inutile car l'unique élément est 0. .
On obtient donc la parcours : (1,4) (4,3) (3,5) (5,6) (6,2) (2,1) qui représente un parcours hamiltonien de poids 63.
L'arbre des décisions ets donc :



7.2 Exemple 2

7.2.1 Données

Soit le graphe suivant :
La matrice des poids est :

	1	2	3	4
1		780	320	580
2	780		700	460
3	320	700		380
4	580	460	380	

On constate qu'ici, le graphe est *symétrique et complet*

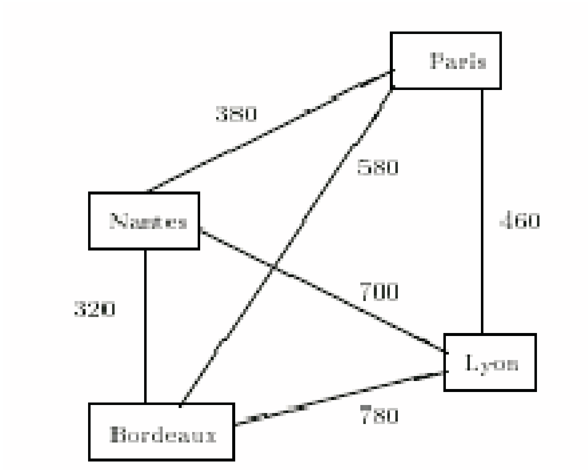


FIG. 7.1 –

7.2.2 Réduction initiale

La réduction des lignes puis des colonnes nous donne le tableau :

	1	2	3	4
1		380	0	260
2	320		240	0
3	0	300		60
4	200	0	0	

Cette première réduction nous donne le poids 1560. (Réduction des lignes : 1480. Réduction des colonnes : 80). On a :

	1	2	3	4
1		380	0 (260)	260
2	320		240	0 (300)
3	0 (260)	300		60
4	200	0 (300)	0 (0)	

7.2.3 Itération 1

On examine arbitrairement l'arc(2,4).
Si on ne passe pas par cet arc, cela coute 1560+300=1860.
Si on passe, il faut reduire la matrice apres avoir empeché de passer par l'arc(4,2) afin de ne pas avoir un circuit. Le cout est alors de 300.
le passage par l'arc(2,4) coute alors 1560+300=1860.

	1	2	3
1		80	0 (80)
3	0 (200)	0 (80)	
4	200		0 (200)

La passage ou non passage nous donne le meme cout. On choisit arbitrairement de passer par l'arc (2,4)

7.2.4 Itération 2

On examine arbitrairement l'arc(4,3).
Si on ne passe pas par cet arc, cela coute 1860+200=2060.
Si on passe, il faut réduire la matrice. on a :

	1	2
1	∞	0 (∞)
3	0 (∞)	∞

Cette réduction a coute 80.
Donc le passage par (4,3) coute 1860+80=1940.
L'évitement de (4,3) coute 2060.
On décide donc de passer par (4,3).
Actuellment les arcs retenus sont : (2,4), (4,3). Il faut interdire l'arc (3,2).

7.2.5 Itération 3

On examine arbitrairement l'arc(3,1).
On ne peut pas ne pas passer par cet arc. Le passage par cet arc coute 0 car apres suppression des lignes et colonnes adéquates, on obtient une matrice 1*1 déjà réduite.

7.2.6 Itération 4

Il reste le passage par l'arc (1,2). Le circuit est donc (1,2,4,3,1) et son cout est 1940.
Si on calcul le poids d'un autre circuit, par exemple 1,2,3,4,1, on obtient le poids 2440.
Si on calcul le poids d'un autre circuit, par exemple 1,4,2,3,1, on obtient le poids 2060.

7.3 Exemple 2 bis

Reprenont l'exemple 2. Nous avions choisi l'arc (2,4) arbitrairement. Supposons maintenant que ce soit l'arc (4,2) que l'on choisisse.
Son evitemment coute : 1560 + 300 = <u>1860</u>.
Calculons combien coute la passage par (4,2) :
Réduisons. On a

	1	3	4
1		0(200)	200
2	80	0 (80)	∞
3	0(80)	∞	0

Le cout de cette réduction est 300. Le cout total est donc 1560+300=<u>1860.</u>
Passer ou ne pas passer par (4,2) a le même cout. On choisit arbitrairement de passer apr (4,2).
<u>Examinons l'arc (1,3).</u>

Son evitemment coute donc 1860+200=2060
Combien coute le passage par (1,3) ,. Réduisons apres avoir empecher de passer par (3,1). on a

	1	4
2	80	∞
3	∞	0

La réduction nous donne

	1	4
2	0	∞
3	∞	0

Le cout est de 80. Le passage par (1,3) nous donne un cout de 1860+80=1940.
Le plus bas des couts (1860, 2060 et 1940) est 1860.
Il faut donc reprendre au debut et choisir d'éviter l'arc (4,2).
On evite l'arc (4,2) : cout 1860. Placons ∞ en (4,2) pour eviter un circuit direct.
On obtient :

	1	2	3	4
1		380	0	260
2	320		240	0
3	0	300		60
4	200	∞	0	∞

Apres réduction on a :

	1	2	3	4
1		80	0 (80)	260
2	320		240	0 (300)
3	0 (200)	0 (80)		60
4	200	∞	0 (200)	∞

On examine l'arc (2,4).
Son évitemment coute 300. Donc le cout total est alors 1860+300=2160.
Combien coute le passage par (2,4) ?. Réduisons :

	1	2	3
1		80	0 (80)
3	0 (200)	0 (80)	∞
4	200	∞	0

La réduction coute 0. Le passage par (2,4) nous dnne donc un cout total de 1860+0=1860.
On poursuit donc dans cette voix.
Examinons l'arc (3,1).
Son évitemment coute 1860+200=2060;
Combien coute le passage apr cet arc ? Réduisons :

	2	3
1	80	∞
4	∞	0

Apres réducion on a :

	2	3
1	0	∞
4	∞	0

Le cout est 80. Le cout total est donc 1860+80=1940.
Nous avons donc le choix entre poursuivre le chemin actuel (2,4),(3,1) ou bien le chemin précedent (4,2),(1,3). Tous les deux ont un cout de 1940.
Choisissons arbitrairement de continuer là où nous en sommes. On a plus précisement :

	2	3
1	0 (∞)	∞
4	∞	0 (∞)

On ne peut donc eviter les arcs (1,2) et (4,3). On choisit arbitrairement de passer par (1,2) ; Son cout est de 0 car apres réduction on a :

	3
4	0

Il reste le passage par (4,3) ;
Le trajet est donc : (2,4),(3,1),(1,2),(4,3) ou encore 1,2,4,3,1. Nous avons le même résultat.

7.4 Exemple 3

Soit un graphe dont le tableau des poids est :

	1	2	3	4	5
1		10		15	
2	10		12	8	19
3		12		10	11
4	15	8	10		20
5		19	11	20	

Il s'agit d'un graphe symetrique non orienté.
la réduction des lignes nous donne :

	1	2	3	4	5	
1		0		5		10
2	2		4	0	11	8
3		2		0	1	10
4	7	0	2		12	8
5		8	0	9		11

Le cout de cette réduction est : 10+8+10+8+11=47.
La réduction des colonnes nous donne :

	1	2	3	4	5
1		0		5	
2	0		4	0	10
3		2		0	0
4	5	0	2		11
5		8	0	9	
	2	0	0	0	1

Ce cout est de 2+1=3.
Au total : 50. On a ensuite :

	1	2	3	4	5
1		0 (5)		5	
2	0 (5)		4	0 (0)	10
3		2		0(0)	0 (10)
4	5	0 (2)	2		11
5		8	0 (10)	9	
	2	0	0	0	1

Choisissons arbitrairement l'arc (5,3).
On place ∞ en (3,5) pour éviter le circuit (5,3),(3,5).
Son evitement coute : 10. Son passage ? réduisons.

	1	2	4	5
1		0	5	
2	0		0	10
3		2	0	∞
4	5	0		11

	1	2	4	5
1		0	5	
2	0		0	0
3		2	0	∞
4	5	0		1

Cout : 10. passons par l'arc(5,3) ; On a

	1	2	4	5
1		0 (5)	5	
2	0 (5)		0 (0)	0 (1)
3		2	0 (2)	∞
4	5	0 (1)		1

Considérons l'arc(1,2). Son evitemment coute 5. Son choix : Réduisons et placons ∞ en (2,1) pour les raisons habituelles..

	1	4	5
2	∞	0 (0)	0 (0)
3		0 (∞)	∞
4	0 (∞)		0 (0)

Son cout est de 5. (1 pour la réduction des lignes et 4 pour la réduction des colonnes.
On choisit l'arc (2,5). Son evitemment coute 0. Son choix ? Réduisons :

	1	4
3		0 (∞)
4	0 (∞)	

On considère l'arc (4,1). Il reste l'arc (3,4). Le chemein est donc : 1,2,5,3,4,1 et so cout est de 65.
prenons un autre chemin : 1,2,3,4,5. On constate que son cout est de 68 et est donc supérieur à 65.

7.5 Exemple 3 bis

Dans l'exemple precedent, au lieu de choisir l'arc (5,3) prenons l'arc (3,5).
On place l'∞ en (5,3) pour éviter un circuit.
Son evitement coute 10.
Son choix coute combien ?
Réduisons la matrice :

	1	2	3	4	5
1		0 (5)		5	
2	0 (5)		2	0 (0)	10
3		2		0(0)	0 (10)
4	5	0 (2)	0		11
5		0	∞	1	
	2	0	0	0	1

Cette réduction a couté : 8+2 = 10.
Le cout du passage est donc de 50+10=60.
Choisissons de passer par l'arc (3,5).
La nouvelle matrice st donc :

	1	2	3	4
1		0 (5)	5	
2	0 (5)		2	0 (1)
4	5	0	0	11
5		0		1

Examinons arbitrairement l'arc (2,1).
Son evitement coute : 5.
Son choix coute combien ?
On obtient comme matrice, apres avoir placé ∞ en (1,2) :

	2	3	4
1	∞	5	
4		0 (5)	0(1)
5	0 (∞)		∞

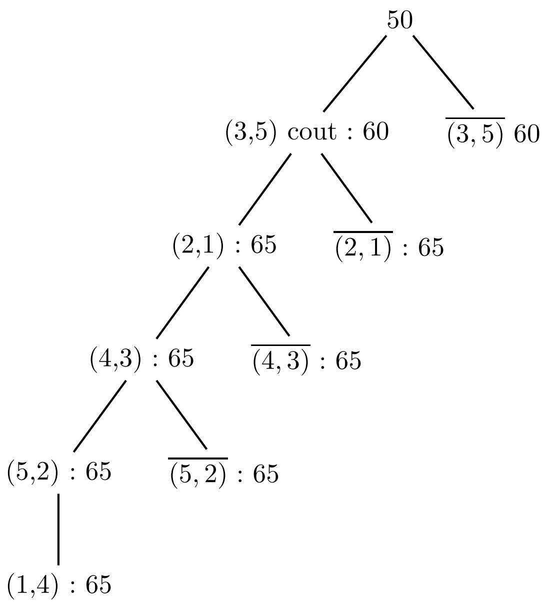
Cette matrice peut se réduire. On obtient :

	2	3	4
1	∞	0	
4		0 (5)	0(1)
5	0 (∞)		∞

Cette réduction coute 5.
Le choix de (2,1) coute donc : 65.
Examinons l'arc (4,3).
Son evitement coute : 0.
Son choix coute combien ?

	2	4
1	∞	∞
5	0 (∞)	∞

Cette réduction a un cout nul.
Il faut passer par (5,2). Il reste l'arc (1,4).
Le circuit est donc : 4,3,5,2,1,4 dont le cout est 65.



7.6 Exemple 4

Soit un graphe dont le tableau des poids est :

	1	2	3	4	5
1		15		8	
2	15		7		12
3		7		10	5
4	8		10		9
5		12	5	9	

Il s'agit d'un graphe symetrique non orienté.
la réduction des lignes nous donne :

	1	2	3	4	5	
1		7		0		8
2	8		0		5	7
3		2		5	0	5
4	0		2		1	8
5		7	0	4		5

Le cout de cette réduction est : 33.
La réduction des colonnes nous donne :

	1	2	3	4	5
1		5		0	
2	8		0		5
3		0		5	0
4	0		2		1
5		5	0	4	
	0	2	0	0	0

Ce cout est de 2.
Au total : 35. On a ensuite :
Les couts d'évitement sont :

	1	2	3	4	5
1		5		0 (9)	
2	8		0 (5)		5
3		0 (5)		5	0 (1)
4	0 (9)		2		1
5		5	0 (4)	4	
	0	2	0	0	0

Examinons l'arc (1,4)
Son évitement coute : 9
Le passage par (1,4) est de combien ? :
La réduction nous donne apres avoir placer ∞ en (4,1)

	1	2	3	5
2	0 (∞)		0 (0)	5
3		0 (5)		0 (0)
4			1	0 (1)
5		5	0 (5)	

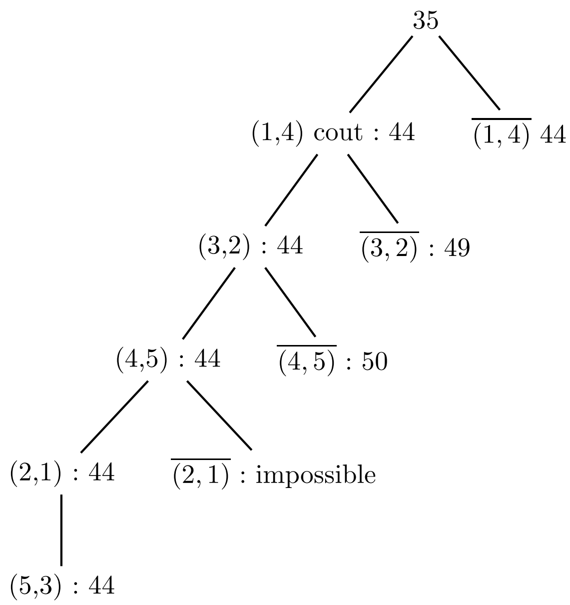
et elle coute 9. on passe apr (1,4)
Examinons l'arc (3,2)
Son évitement coute : 5
Combien coute le choix de passer par (3,2) ? Réduisons la matrice.

	1	3	5
2	0 (∞)		5
4		1	0 (6)
5		0 (∞)	

Cette réduction a un cout nul. on passe apr (3,2)
Examinons l'arc (4,5)
Son evitement coute : 6.
Le passage par cet arc coute combien ? Réduisons :

	1	3
2	0 (∞)	∞
5		0 (∞)

Cette réduction coute 0.
On passe donc par (4,5).
Il reste manifestement les arcs (2,1) et (5,3) dont le cout est nuL.
Le circuit est donc : 1, 4, 5, 3, 2, 1 dont le cout est 44.

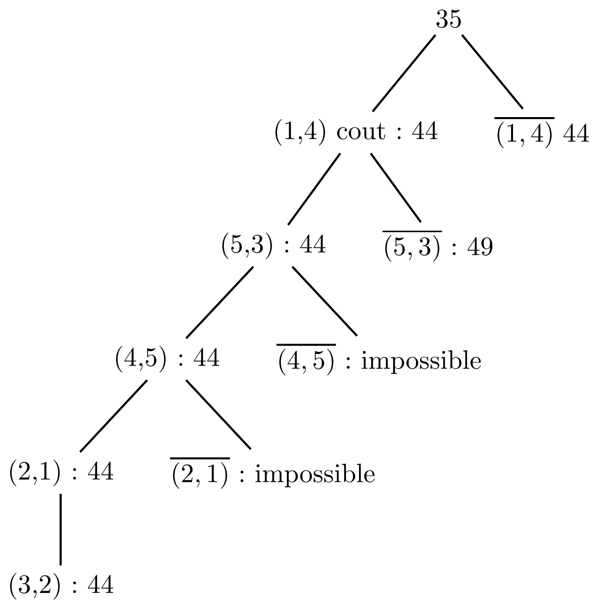


7.7 Exemple 4 bis

reprenons l'exemple precedent, mais faisons un autre choix apres avoir decidé de passer par (1,4).
Examinons l'arc (5,3) au lieu de l'arc (3,2).
Son évitement coute : 5
Combien coute le choix de passer par (5,3) ? Réduisons la matrice apres avoir placé ∞ en (3,5).

	1	2	5
2	0 (∞)		5
3		0 (∞)	∞
4			0 (∞)

Cette réduction a un cout nul. On passe apr (5,3)
Examinons l'arc (4,5)
Son evitement est impossible.
Le passage par cet arc coute 0.
Il est clair qu'il faut passer par les arcs ((2,1), (3,2) et (4,5) puisqu'ils sont inévitables ! On passe donc par (4,5).
Le circuit est donc : 1, 4, 5, 3, 2, 1 dont le cout est 44.


35
(1,4) cout : 44
$\overline{(1,4)}$ 44
(5,3) : 44
$\overline{(5,3)}$: 49
(4,5) : 44
$\overline{(4,5)}$: impossible
(2,1) : 44
$\overline{(2,1)}$: impossible
(3,2) : 44

Chapitre 8

Ordonnancement

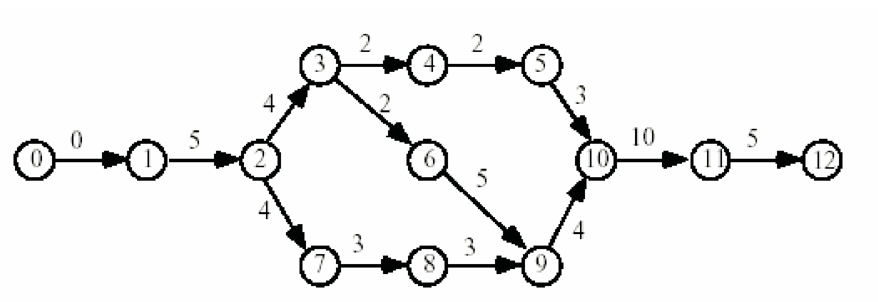
8.1 Exemple

Soit un chantier dont les tâches sont les suivantes :

No	tâches	durée	préalables
1	terassement	5	-
2	fondations	4	1
3	colonnes porteuses	2	2
4	charpente toiture	2	3
5	couverture	3	4
6	maconnerie	5	3
7	plomberie, élécticité	3	2
8	coulage dalle béton	3	7
9	chauffage	4	8 et 6
10	plâtre	10	9 et 5
11	finitions	5	10

8.2 Représentation graphique

soit :



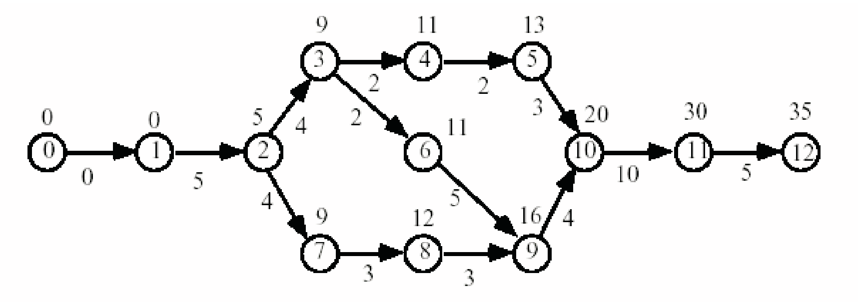
8.2.1 Généralités

Pas de circuit
Date de début au plus tôt
Date de début au plus tard
Marge totale d'une tâche.

8.2.2 Des contraintes

Contraintes potentielles
Contraintes de type disjonctif.
Contraintes de type cumulatif.

8.2.3 Méthode MPM (Potentiels/tâches)



Les principes

Tâche	=sommet
Contrainte	=arc valué.

Ajout de deux taches fictives : début et fin. On peut modifier des relations entre tâches par simple adjonction ou suppression d'arc.
Une solution est donc la recherche d'un chemin le plus long dans le graphe. Ce chemin est appelé le *chemin critique*.

Les contraintes

Localisation.
Succession.

Type d'ordonnancement

Ordonnancement au plus tôt.
Ordonnancement au plus tard.
Type de tâches : critiques ou non.
Chemin critique.
Marge d'une tâche.

8.2.4 Méthode PERT (Potentiels/étapes)

Tâche	=arc valué par la durée de la tâche
Etape	=sommet.

Les relations de succession temporelle sont représentées par les sommets. Condition de démarrage sont représentées par des sommets fictifs et/ou des tâches fictives valuées par une durée nulle..

Chapitre 9

Eléments de logique

9.1 Introduction

9.1.1 Enoncé

9.1.2 Propositions

9.1.3 Connecteurs logiques

Non

Et

Ou

Deux types de "ou"

Implique

9.1.4 Lois de De Morgan

Chapitre 10

Test

Soit un graphe dont la matrice des poids est :

$$\begin{pmatrix} 0 & \infty & 5 & 10 & 3 \\ 1 & 0 & \infty & 4 & 3 \\ \infty & \infty & 0 & \infty & \infty \\ \infty & \infty & \infty & 0 & \infty \\ \infty & 2 & \infty & \infty & 0 \end{pmatrix}$$

10.1 Dijkstra

Nous remarquons la présence de circuits. Cependant aucun n'est absorbant.
Tous les poids sont positifs.

10.1.1 Initialisation

Le tableau de départ est :

	1	2	3	4	5
pi	0	∞	5	10	3
Etiq		x_1	x_1	x_1	x_1

Le sommet de départ est x_1. On a $D = \{x_1\}$.

10.1.2 Itération 1

On choisit le sommet 5. Donc $D = \{x_1, x_5\}$. La mise à jour du poids total des autres sommets nous donne :
$pi(x_2) \leftarrow min\{\infty, 2+3\} = 5$
$pi(x_3) \leftarrow min\{5, 3+\infty\} = 5$
$pi(x_4) \leftarrow min\{10, 3+\infty\} = 10$
Le tableau devient :

	1	2	3	4	5
pi	0	5	5	10	3
Etiq		x_5	x_1	x_1	x_1

10.1.3 Itération 2

On choisit le sommet 2. Donc $D = \{x_1, x_5, x_2\}$. La mise à jour du poids total des autres sommets nous donne :
$pi(x_3) \leftarrow min\{5, 5 + \infty\} = 5$
$pi(x_4) \leftarrow min\{10, 4 + 5\} = 9$
Le tableau devient :

	1	2	3	4	5
pi	0	5	5	9	3
Etiq		x_5	x_1	x_2	x_1

10.1.4 Itération 3

On choisit le sommet 3. Donc $D = \{x_1, x_5, x_2, x_3\}$. La mise à jour du poids total des autres sommets nous donne :
$pi(x_4) \leftarrow min\{9, 5 + \infty\} = 5$
Le tableau devient :

	1	2	3	4	5
pi	0	5	5	9	3
Etiq		x_5	x_1	x_2	x_1

Les plus courts chemins sont donc :
$x_1 \rightarrow x_5 \rightarrow x_2 \rightarrow x_4$
$x_1 \rightarrow x_3$

10.1.5 Remarque

On ne peut adapter l(algorithme de Dijkstra pour la recherche d'un plus long chemin dans le cas de l'exercice. En effet, il y a un circuit qui permet d'augmenter la longueur d'un chemin de manière illimitée.

10.2 Dijkstra 2

10.2.1 Le graphe

10.2.2 Initialisation

On a

	1	2	3	4	5
pi	0	12	25	8	15
etq	x_1	x_1	x_1	x_1	x_1

10.2.3 itération 1

On sélectionne le sommet 4.
On calcul la mise à jour des autres sommets

1. sommet 2 : 8+10 < 12 ? non pas de mise à jour
2. sommet 3 : 8+20< 25 ? non pas de mise à jour
3. sommet 5 : 8+9 < 15 ? non pas de mise à jour

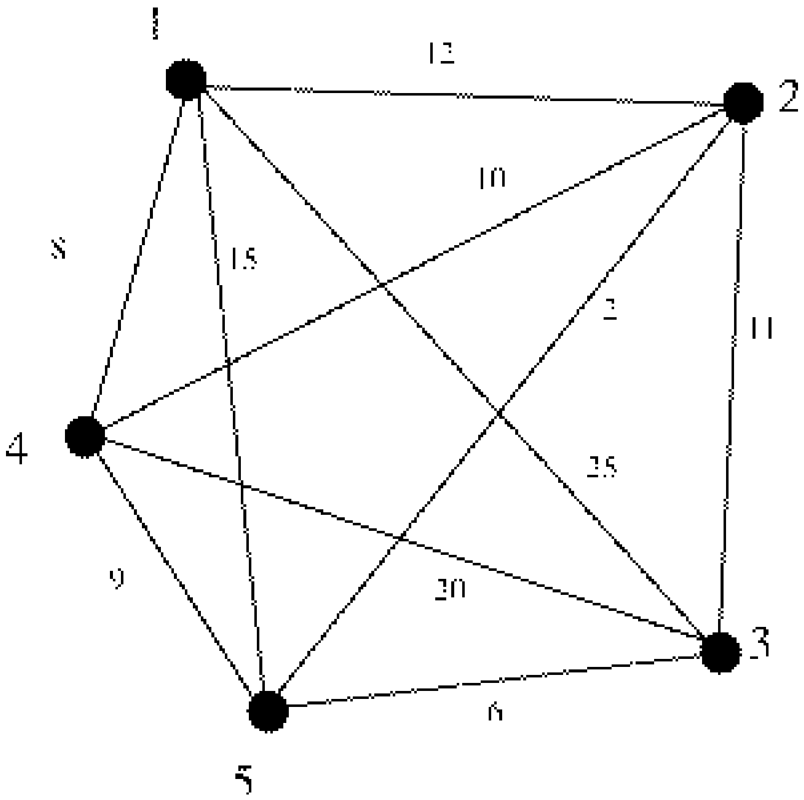


FIG. 10.1 –

10.2.4 itération 2

On sélectionne le sommet 2.
On calcul la mise à jour des autres sommets

1. sommet 3 : 12+11< 25 ? Oui mise à jour
2. sommet 5 : 12+2 < 15 ? Oui mise à jour

On obtient :

	1	2	3	4	5
pi	0	12	23	8	14
etq	x_1	x_1	x_2	x_1	x_2

10.2.5 itération 3

On sélectionne le sommet 5.
On calcul la mise à jour des autres sommets

1. sommet 3 : 14+6 < 23 ? Oui mise à jour

On obtient :

	1	2	3	4	5
pi	0	12	20	8	14
etq	x_1	x_1	x_5	x_1	x_2

10.2.6 Solution

On a donc comme chemin :
$x_1 \rightarrow x_2 \rightarrow x_5 \rightarrow x_3$
$x_1 \rightarrow x_4$

10.3 Fermeture transitive

Prenons la matrice d'adjacence et rendons la reflexive afin d'utiliser l'algorithme le plus rapide.
on a :

$$(I \oplus M) = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

En élevant au carré de manière booléenne, on a :

$$(I \oplus M)^2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Elevons de nouveau au carré.

$$(I \oplus M)^4 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Elevons de nouveau au carré.

$$(I \oplus M)^8 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

La matrice est stable. C'est la fermeture transitive.

Remarque : Pour l'examen, il faut etre capable de :

1. Calculer la fermeture transitive d'un graphe.
   Il faut connaître la méthode rapide (rendre reflexif, puis élévations au carré successives) et Warschall.
2. Restituer les algorithmes vus au cours
3. Calculer le plus court chemin à l'aide de Dijkstra.
4. Appliquer Little sur un graphe à 5 sommets au maximum.
5. Réaliser un parcours de graphe selon toutes le méthodes vues au cours.
6. Trouver le chemin critique dans un graphe d'ordonnancement.

10.4 FLOYD

Soit un graphe dont la matrice d'adjacence st :

$$M_1 = \begin{pmatrix} 0 & 3 & \infty & 3 \\ 2 & 0 & 2 & 2 \\ -2 & \infty & 0 & 1 \\ \infty & 4 & 4 & 0 \end{pmatrix}$$

L'examen de la **première** colonne nous donne :
Les predecesseurs sont : 2 et 3.

Les successeurs de 2 sont : 3 et 4
Les successeurs de 3 sont : 2 et 4.
On examine :
Les successeurs de 2 :
$2 \to 1 \to 3 < 2 \to 3$? ou encore $2 + \infty < 2$? non
$2 \to 1 \to 4 < 2 \to 4$? ou encore $2 + 3 < 2$? non
Les successeurs de 3 :
$3 \to 1 \to 2 < 3 \to 2$? ou encore $-2 + 3 < \infty$? OUI donc MAJ
$3 \to 1 \to 4 < 3 \to 4$? ou encore $-2 + 3 < 1$? non

$$M_2 = \begin{pmatrix} 0 & 3 & \infty & 3 \\ 2 & 0 & 2 & 2 \\ -2 & 1 & 0 & 1 \\ \infty & 4 & 4 & 0 \end{pmatrix}$$

L'examen de la **deuxième** colonne de M_2 nous donne :

Les predecesseurs sont : 1, 3 et 4.
Les successeurs de 1 sont : 3 et 4
Les successeurs de 3 sont : 1 et 4
Les successeurs de 4 sont : 1 et 3.
On examine :
Les successeurs de 1 :
$1 \to 2 \to 3 < 1 \to 3$? ou encore $3 + 2 < \infty$? OUI donc MAJ
$1 \to 2 \to 4 < 1 \to 4$? ou encore $3 + 2 < 3$? non
Les successeurs de 3 :

$3 \to 2 \to 1 < 3 \to 1$? ou encore $1 + 2 < -2$? non
$3 \to 2 \to 4 < 3 \to 4$? ou encore $1 + 2 < 1$? non
Les successeurs de 4 :
$4 \to 2 \to 1 < 4 \to 1$? ou encore $4 + 2 < \infty$? OUI donc MAJ
$4 \to 2 \to 3 < 4 \to 3$? ou encore $4 + 2 < 4$? non

$$M_3 = \begin{pmatrix} 0 & 3 & 5 & 3 \\ 2 & 0 & 2 & 2 \\ -2 & 1 & 0 & 1 \\ 6 & 4 & 4 & 0 \end{pmatrix}$$

L'examen de la **troisième** colonne de M_3 nous donne :
Les predecesseurs sont : 1, 2 et 4.
Les successeurs de 1 sont : 2 et 4
Les successeurs de 2 sont : 1 et 4.
Les successeurs de 4 sont : 1 et 2.
On examine :
$1 \to 3 \to 2 < 1 \to 2$? ou encore $5 + 1 < 3$? non
$1 \to 3 \to 4 < 1 \to 4$? ou encore $5 + 1 < 3$? non
$2 \to 3 \to 1 < 2 \to 1$? ou encore $2 - 2 < 2$? OUI donc MAJ
$2 \to 3 \to 4 < 2 \to 4$? ou encore $2 + 1 < 2$? non
$4 \to 3 \to 1 < 4 \to 1$? ou encore $4 - 2 < 6$? OUI donc MAJ
$4 \to 3 \to 2 < 4 \to 2$? ou encore $4 + 1 < 4$? non

$$M_4 = \begin{pmatrix} 0 & 3 & 5 & 3 \\ 0 & 0 & 2 & 2 \\ -2 & 1 & 0 & 1 \\ 2 & 4 & 4 & 0 \end{pmatrix}$$

L'examen de la **quatrième** colonne de M_4 nous donne :
Les predecesseurs sont : 1, 2 et 3.
Les successeurs de 1 sont : 2 et 3
Les successeurs de 2 sont : 3.
Les successeurs de 3 sont : 1 et 2.
On examine :
$1 \to 4 \to 2 < 1 \to 2$? ou encore $3 + 4 < 3$? non
$1 \to 4 \to 3 < 1 \to 4$? ou encore $3 + 4 < 5$? non
$2 \to 4 \to 3 < 2 \to 3$? ou encore $2 + 4 < 5$? non
$3 \to 4 \to 1 < 3 \to 1$? ou encore $1 + 2 < -2$? non
$3 \to 4 \to 2 < 3 \to 2$? ou encore $1 + 4 < 1$? non
$3 \to 4 \to 2 < 3 \to 2$? ou encore $1 + 4 < 1$? non

$$M_5 = \begin{pmatrix} 0 & 3 & 5 & 3 \\ 0 & 0 & 2 & 2 \\ -2 & 1 & 0 & 1 \\ 2 & 4 & 4 & 0 \end{pmatrix}$$

On connait les poids, mais pas le détail du chemin.