# Notions fondamentales de la théorie des graphes

**1. Notions préliminaires**

**1. 1 Application multivoque et relation binaire**

Considérons deux ensembles E et E’, distincts ou non. Soit ℜ une relation donnant à certains couples (e, e’) ∈ E×E’ une caractéristique telle que la famille des couples possédant la propriété d’être reliés par cette relation soit une partie parfaitement définie U de E×E’. ℜ est dite relation binaire.

Dans le cas particulier où E = E’ = X, on dit que ℜ est définie sur X.

Soit Γ un procédé parfaitement définie permettant d’associer à chaque élément e d’un ensemble (source) E, un ou plusieurs (éventuellement aucun) élément(s) d’un ensemble (cible) E’, appliquant en quelque sorte e sur chacun des éléments du sous-ensemble – éventuellement vide – de la cible ainsi défini ; on donne à un tel procédé le nom d’application multivoque de E dans E’.

On note Γ(e) l’ensemble des éléments de E’ ainsi associés à un élément quelconque e ∈ E. Si e’∈ Γ(e) on dit encore que e’ est une image de e, Γ(e) étant par définition l’ensemble des images de e. Cet ensemble peut être vide, réduit à un seul élément, en contenir un nombre quelconque d’éléments.

Dans le cas particulier où E = E’ = X, on dit encore que l’application multivoque Γ est définie sur X.

Il est utile d’en déduire une application dite application multivoque réciproque notée Γ−, admettant E’ comme ensemble source et E comme ensemble cible.

**1. 2 Graphe et ses constituants de base**

Soit X = {x1, x2, …, xn} un ensemble de points sur le plan et Γ une application définie sur X telle que à chaque élément de X, elle fait correspondre soit le même élément ou un autre élément de X lui même. Le couple (X, Γ) définit un graphe noté G = (X, Γ).

Les éléments de X sont dits **sommets**.

Un sommet xi est relié à un autre sommet xj par une flèche dite **arc**.

Donc, un arc est un couple (xi, xj) tel que xj = Γ(xi). xj est dit, dans ce cas, **successeur** de xi, et xi est dit **prédécesseur** de xj .

Un graphe G peut-être aussi noté par G = (X, U) où U est l’ensemble des arcs défini sur (X×X). On dit dans ce cas que le graphe est un sous-ensemble du produit cartésien X×X.

**1.3 Graphe en tant que représentation d’une relation binaire ou d’une application multivoque**

Schématiquement, un graphe G = (X, U) apparaît comme une figure simplifiée visant à représenter non pas la forme, mais les relations des objets symbolisés par X. Conformément à cette conception schématique, G peut toujours être regardé comme une représentation graphique d’une relation binaire ℜ de X à X. U représentant l’ensemble des couples ordonnés de X2 pour lesquels ℜ est vérifiée.

Ce qui vient d’être dit en termes de relations binaires se transpose sans aucune difficulté en termes d’applications multivoques étant donné ce qui a été exposé à propos des rapport qui lient ces deux concepts. Nous parlons de la représentation graphique d’une application multivoque pour désigner le graphe G, dont :

* les sommets représentent les éléments de la réunion des ensembles cible et source ;
* les arcs relient chaque élément de la source à ceux de la cible sur lesquels il est appliqué par Γ.

1. **Que représente un graphe ?**

Un graphe représente un grand nombre de situations ou de problèmes réels :

* L’exemple le plus classique est la représentation d’un réseau de communication : réseau de routes, réseaux de chemins de fer, de téléphone, de relais de télévision, réseau électrique, réseau des informations dans un organisme quelconque, … etc.
* La famille d’exemples la plus générale est la représentation d’une relation binaire, qu’elle soit algébrique, mécanique, chimique, sociologique, …etc, comme les règles de certains jeux (dames ou échecs), la supériorité des participants à un tournoi, les relations de parenté d’un groupe d’individus, l’ordonnancement des opérations de montage ou de démontage d’un ensemble technologique, … etc.
* Un cas particulier important est la représentation d’une évolution, comme le passage d’un certain état à un autre : chaînes de Markov, évolution d’une population dans un phénomène démographique, processus génétique, …etc.

1. **Représentation d’un graphe**
2. **Représentation sagitale**

Soit X = {a, b, c, d, e}, on définit Γ(a) = {b, d}, Γ(b) = {c, e}, Γ(d) = {b, e}, Γ(c) = {e}, Γ(e) = φ.

Donc, la représentation sagitale d’un tel graphe est :

b c

a

e

d

On déduit Γ − (a) = φ , Γ − (b) = {a, d}, Γ − (c) = {b}, Γ − (d) = {a}, Γ − (e) = {b, c, d}.

Le sommet i qui est tel que Γ − (i) = φ est dit **entrée du graphe**, et le sommet j qui est tel que Γ(j) = φ est dit **sortie du graphe**. (Par Γ − (i), on entend la fonction inverse de Γ qui donne l’ensemble des prédécesseurs de i)

1. **Représentation matricielle**
2. **A l’aide d’une matrice booléenne**

Au graphe précédent, on peut faire correspondre la matrice booléenne suivante, où la présence d’un 1 en case (i, j) indique une correspondance entre le sommet i et le sommet j.

a b c d e

a 0 1 0 1 0

b 0 0 1 0 1

## M = c 0 0 0 0 1

d 0 1 0 0 1

e 0 0 0 0 0

Cette matrice est aussi dite **matrice d’adjacence**.

Dans cette représentation, si on fait la somme des éléments d’une ligne i, on obtient le nombre d’arcs incidents extérieurement au sommet i, ou le **demi-degré extérieur** du sommet i.

Respectivement, la somme des éléments d’une colonne j donne le nombre d’arcs incidents intérieurement au sommet j, ou le **demi-degré intérieur** du sommet j.

Le sommet dont le demi-degré intérieur est nul est dit entrée du graphe.

De même, le sommet dont le demi-degré extérieur est nul est dit sortie du graphe.

**Matrice d’incidence sommets-arcs**

De la même manière on définit la matrice d’incidence sommet-arc.

En ligne, on trouve les sommets x1, x2, …, xn du graphe et en colonne, ses arcs u1, u2, …, um. Si l’arc uj est incident extérieurement au sommet xi, alors la case (i, j) de la matrice est à −1. Par contre, si il est incident intérieurement, alors elle est à +1. Elle est à 0 si le sommet xi n’est ni une extrémité initiale ni une extrémité terminale de l’arc uj.

1. **à l’aide d’une matrice littérale**

A l’intersection de la ligne i et de la colonne j de cette matrice, on inscrit ij si l’arc (i,j) existe ; autrement, on laisse la case vide. A titre d’exemple la matrice littérale correspondante à l’exemple des paragraphes précédents peut s’écrire sous la forme :

a b c d e

a ab ad

b bc be

## M = c ce

d db de

e

**Remarque**

Dans la représentation matricielle (booléenne particulièrement), la matrice d’adjacence (sommet – sommet) permet d’indiquer s’il y a une liaison entre un sommet et un autre. On voit que la quantité d’information nécessaire pour représenter une telle matrice est N2 (N étant le nombre de sommets). Dans le cas des graphes peu denses où M (nombre d’arcs) << N2 (M<< ½ N(N+1) pour les graphes non orientés), il y a une perte considérable d’informations. Il serait, donc, avantageux de décrire uniquement les termes non nuls de la matrice d’adjacence. D’où, on fait appel au mode suivant de représentation.

1. **Représentation par tableaux**

On utilise deux tableaux A de dimension N et B de dimension M dans le cas orienté, et de dimension 2M dans le cas non orienté. Pour chaque sommet i, la liste des successeurs de i est contenue dans B à partir de la case numéro A(i). On voit donc que l’ensemble des informations relatives au sommet i est situé entre les cases A(i) et A(i +1)−1 du tableau B.

**Exemple**

Au graphe décrit précédemment, on fait correspondre les deux tableaux qui suivent :

A a b c d e

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 6 | 8 |

B

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| B | d | c | e | e | b | e | Φ |

1 2 3 4 5 6 7 8

**4. Vocabulaire de base associé aux graphes**

1. **Chemin**: c’est une suite d’arcs dont l’extrémité terminale de chacun, sauf le dernier, est l’extrémité initiale du suivant.
2. **Circuit**: un chemin qui se ferme sur lui même est dit circuit.
3. **Chemin simple** : un chemin est dit simple s’il ne passe qu’une et une seule fois par chacun de ses arcs.
4. **Chemin élémentaire** : c’est tout chemin qui ne rencontre pas plus d’une fois chacun de ses sommets.
5. La **longueur d’un chemin** est le nombre d’arcs qui le composent.
6. Un circuit de longueur 1 est une **boucle**.

**Exemple**

La figure suivante représente un graphe dans lequel :

B

- DAEAEA est un chemin de longueur 5

A C - ACDA est un circuit

- En B existe une boucle

- DAEAEA n’est pas simple

- DAEA est simple, mais non élémentaire

- DAE est élémentaire et simple.

E D

1. Un **chemin** est dit **hamiltonien** s’il passe une fois et une seule par tous les sommets du graphe
2. Dans le cas des graphes non orientés, un arc devient une **arête**, un chemin une **chaîne** et un circuit un **cycle**.
3. Le nombre d’arêtes incidentes à un sommet détermine son **degré**. Ce nombre peut être pair ou impair. De façon générale, le nombre d’arêtes dans un graphe est la moitié de la somme des degrés de ses sommets. Donc, la somme des degrés de tous les sommets est un nombre pair.

Problème du plus court chemin

1. **Introduction**

Les problèmes de cheminement sont parmi les problèmes les plus anciens de la théorie des graphes. On les trouve dès l’antiquité dans les problèmes célèbres de la ‘’traversée du batelier : un batelier doit faire traverser un loup, une chèvre et un chou d’une rive à une autre. Le bateau est si petit qu’il ne peut contenir que deux passagers à la fois’’.

Le problème du plus court chemin est parmi ces problèmes les plus typiques. Il se rencontre soit directement, soit comme sous-problème dans de nombreuses applications, entre autres :

* Les problèmes de tournées,
* Les problèmes d’optimisation de réseaux : routiers, de télécommunication, … etc,
* Certains problèmes de l’intelligence artificielle et de la reconnaissance des formes,
* Certains problèmes d’investissements et de gestion de stocks,
* De nombreux problèmes de programmation dynamique à états discrets et temps discret,
* Certaines méthodes de traitement numérique du signal, de codage et de décodage de l’information.
* …etc.

1. **Définitions**

Etant donné un graphe G= (X, U), on associe à chaque arc u∈U un nombre l(u)∈ℝ appelé ‘’longueur de l’arc’’ ; on dit que G est valué par les longueurs l(u). Si u= (i, j), on utilisera également la notation lij pour désigner la longueur de l’arc u.

Le problème du plus court chemin entre deux sommets i et j est de trouver un chemin μ(i, j) de i à j dont la longueur totale :

l(μ )= ∑u∈μ(i, j) l(u) est minimale.

Ce problème a de nombreuses applications pratiques ; car, la longueur l(u) peut s’interpréter aussi bien comme un coût de transport sur l’arc u, comme les dépenses de construction de l’arc u, comme le temps nécessaire pour parcourir l’arc u, …etc.

1. **Conditions d’existence**

Considérons un chemin μ(i, j) de i à j comprenant un circuit ω.

j

k

i

ω

On notera μ’ le chemin de i à j associé à μ et n’empruntant pas le circuit ω. On a :

l(μ ) = l(μ’) + l(ω)

où l(ω) représente la longueur du circuit ω.

Si l(ω)<0, il n’existe pas de plus court chemin de i à j (on peut utiliser une infinité de fois le circuit ω).

Si l(ω)≥0, alors l(μ’)≤ l(μ). Dans la recherche d’un plus court chemin, on peut donc se restreindre aux chemins élémentaires entre i et j.

Dans toute la suite, on supposera donc que le graphe n’admet pas de circuit de longueur négative.

Comme les chemins élémentaires forment un ensemble fini, il existe alors toujours un plus court chemin de i à j si j∈Γ^(i) (fermeture transitive de i).

1. **Les algorithmes**

Les algorithmes de résolution de ce problème sont différents selon les propriétés du graphe :

* l(u) ≥ 0 ∀u∈ U,
* l(u) = 1 ∀u∈ U,
* G et l(u) quelconques,
* G sans circuit,

et suivant le problème considéré :

* recherche du plus court chemin d’un sommet à un autre,
* recherche du plus court chemin d’un sommet à tous les autres,
* recherche du plus court chemin entre tous les couples de sommets.

**La rapidité des algorithmes dépendra souvent de la représentation en machine du graphe.**

**4.1. Cas où l(u)** **≥ 0, ∀u∈ U**

**Algorithme de Moore-Dijkstra :** il consiste à calculer le plus court chemin d’un sommet numéro 1 à tous les autres.

Posons lij la longueur de l’arc (i, j) si (i, j)∈U.

Définissons π\*(i) comme la longueur minimale des chemins de 1 à i ; en particulier, π\*(1)=0.

L’algorithme procède en N−1 itérations (si N est le nombre de sommets). Au début de chaque itération, l’ensemble des sommets est partitionné en deux sous-ensembles S et S= X−S, avec 1∈S.

Chaque sommet i de X est affecté d’une étiquette π(i) qui vérifie la propriété :

* si i∈S , π(i)= π\*(i) ,
* si i∈S , π(i)= mink∈S, k∈Γ−(i)  (π(k) + lki)

La valeur de π(i) (i∈S ) donne la longueur minimale des chemins de 1 à i, soumis à la condition que tous ses sommets, expté i, sont dans S.

**Lemme**

Soit j ∈ S tel que π(j) = mini∈S (π(i)), alors π\*(j) = π(j).

(Le chemin qui mène à j est le plus court).

**Démonstration**

Il existe évidemment un chemin de 1 à j de longueur π(j). Pour montrer qu’il n’existe pas de chemin plus court de 1 à j, considérons un chemin μ de 1 à j et divisons le en deux parties : μ1 la partie commençant du chemin allant de 1 à h, où h est le premier sommet de S rencontré, et μ2 le reste du chemin (allant de h à j). Alors, la longueur de μ1 est ≥ π(h) ≥ π(j) et la longueur de μ2 est positive. On en déduit que la longueur de μ ≥ π(j).

L’algorithme de Moore et de Dijkstra peut alors être formulé comme suit :

**Algorithme 1**

{Recherche du plus court chemin du sommet 1 aux autres dans un graphedont les longueurs sont positives.}

1. Initialisation :

S = {2, 3, …, N},

π(1)= 0,

π(i) = l1i si i∈Γ(1)

+ ∞ sinon

1. Sélectionner j∈ S tel que π(j)= mini∈S (π(i)) et faire :

S ←⎯ S − {j} (⇒ S←⎯ S + {j})

Si | S | = 0 alors Fin.

Sinon :

1. Pour tout i∈Γ(j) et i∈S faire : π(i) ←⎯ min ( (π(i), (π(j) + lji) )

retourner en (b).

L’étape (c) revient à ajouter la valeur π(i) (i∈S) afin de tenir compte du fait que j est maintenant dans S.

L’algorithme définit donc l’un après l’autre les sommets les plus proches de 1.

Il conduit à la construction d’une arborescence de racine 1 qui détermine les plus courts chemins de 1 à tous les autres sommets i∈X.

Remarquons, cependant, que si on veut déterminer explicitement le plus court chemin et non pas seulement sa longueur, nous devons, dans l’algorithme précédent, tenir à jour un vecteur P(i)=j si π(i)= π(j) + lji . Ce vecteur sera mis à jour à l’étape (c) chaque fois que les π(i) diminueront. (Au départ P(i)= 1, ∀ i∈Γ(1)).

**Exemple**

On désire appliquer l’algorithme précédent sur le graphe suivant :

4

7 2 5

5 1

1 2

3

7

**4.2. Cas où l(u) = 1, ∀u∈U**

Un cas plus particulier que le cas des longueurs positives, et pourtant assez courant, est celui où toutes les longueurs des arcs sont égales. On peut alors améliorer l’algorithme précédent pour le calcul du plus court chemin du sommet 1 à tous les autres.

**Algorithme 2**

{Recherche du plus court chemin du sommet 1 aux autres sommets dans un graphe dont les longueurs sont toutes égales.}

1. Poser

π(1) = 0, π(i) = + ∞ ∀i ≥2

k= 0, S= {1}, S0= {1}

{π(i) est une étiquette qui, dès qu’elle n’est plus égale à +∞, sera égale à la

longueur du plus court chemin de 1 à i.}

1. A l’itération k, soit Sk={i / π(i) = k}, S={i / π(i) ≤ k}, S = X−S.

Poser

Sk+1= Γ( Sk)∩S

π(i) = k+1 pour i∈ Sk+1

S←⎯ S∪Sk

1. Si |S|= |X| alors Fin.

Sinon aller en (b) avec k ←⎯ k+1.

**L’exploration du graphe défini par l’algorithme 2 est une exploration en largeur d’abord.**

**Exemple**

Reprenons le graphe précédent avec des longueurs toutes identiques.

* 1. **Graphe G et longueur l(u) quelconques**

Notons par π(j) la longueur du plus court chemin de 1 à j. Considérons l’arc (i, j) de longueur lij.

Le plus court chemin de 1 à j pouvant passer par i doit vérifier :

π(j)≤ π(i) + lij

ou bien : π(j) − π(i) ≤ lij.

Cette dernière relation nous invite à considérer le vecteur π comme un potentiel (la différence des potentiels entre i et j doit être inférieure ou égale à lij). Un algorithme associé à ce cas sera alors le suivant :

**Algorithme 3**

{Recherche du plus court chemin du sommet 1 à tous les autres dans un graphe quelconque représenté par la liste de ses arcs.}

1. Poser π(1) = 0 et π(i) = + ∞ pour les autres sommets.
2. Chercher un arc (i, j) tel que π(j) − π(i) > lij . S’il n’en existe pas alors Fin.
3. Sinon poser π(j) = π(i) + lij et retourner en (b).

Cet algorithme est aussi connu sous le nom de ‘’algorithme de FORD’’.

A chaque étape de l’algorithme, π(i) est un majorant du plus court chemin de 1 à i.

Puisque l’on suppose le graphe sans circuit de longueur négative, π(i) est borné inférieurement par la longueur du plus court chemin de 1 à i.

Comme π(i) décroît et est borné inférieurement, il admet une limite et l’algorithme converge.

Pratiquement, on n’utilise pas l’algorithme 3, mais une amélioration qui tient compte du fait que le graphe peut être donné par les successeurs de chaque sommet. Le principe est le suivant :

On ne cherche dans (b) un arc (i, j) améliorant que parmi les sommets i dont l’étiquette π(i) a été précédemment modifiée. On cherche alors à améliorer tous les successeurs du sommet i.

* 1. **G peut admettre un circuit**

**Algorithme 4**  (Bellman 1957).

{Recherche du plus court chemin du sommet 1 aux autres dans un graphe quelconque (ou détection d’un circuit négatif.}

1. Poser π0(1) = 0 et π0(i) = + ∞ pour les autres sommets i.

K= 1

1. A l’itération k, faire πk(1) = 0 et pour tous les sommets i, poser :

πk(i) = min (πk−1(i) , minj∈Γ−(i) (πk−1(j) + lji))

1. Si πk(i) = πk−1(i) pour tout i, alors Fin.

Si k ≤ N−1 alors aller en (b) avec k ←⎯ k+1.

Si k= N alors il existe un circuit de longueur négative .

L’algorithme 4 peut être amélioré au niveau de l’itération (b) :

Supposons que nous calculons les πk(i) de l’étape (b) dans un ordre donné, par exemple 2, 3, …, N. Pour calculer ces coefficients, nous pouvons utiliser les valeurs πk(j) pour j<i calculés au début de l’itération (b). L’équation est alors la suivante :

πk(i) = min(πk−1(i) , minj∈Γ−(i), j<i (πk(j) + lji), minj∈Γ−(i), j>i (πk−1(j) + lji))

**Les algorithmes matriciels**

On définit la matrice L = [ lij ] comme suit :

longueur de l’arc (i, j) si (i, j)∈U

lij =

+ ∞ sinon

**Algorithme 5**

{Recherche de la matrice des plus courts chemins ou détection d’un circuit de longueur négative}.

Pour k allant de 1 à N faire

Pour i allant de 1 à N faire

Si lik = + ∞ alors passer au sommet i suivant.

Si lik + lki < 0, alors arrêt : circuit négatif.

Pour j allant de 1 à N faire

lij ←⎯ min (lij , lik + lkj ).

**Exemple**

3

2

−2 2 4

2 3

1

4

Il est clair que L0 peut être formulée comme suit (l’inexistence des boucles a été interprétée par des 0 dans la diagonale) :

1 2 3 4

1 0 3 +∞ 3

L0  = 2 2 0 2 2

3 −2 +∞ 0 1

4 +∞ 4 4 0

Les autres matrices s’obtiennent facilement en exécutant les boucles de l’algorithme.

Dans certaines applications, les graphes sont appelés à être partiellement modifiés au cours de la résolution, ou en d’autres termes à subir des extensions. L’algorithme suivant construit les plus courts chemins sur ce type de graphes. La méthode est particulièrement intéressante lorsque l’on est amené à modifier le graphe pour lequel on connaît déjà la matrice des plus courts chemins.

**Algorithme 6** (Dantzig 1966).

{Recherche de la matrice des plus courts chemins ou détection d’un circuit de longueur négative}.

Pour k allant de 1 à N faire

Pour i allant de 1 à K faire

1. li,k+1←⎯ min 1≤j≤k ( li,j + lj,k+1 )

lk+1,i ←⎯ min 1≤j≤k ( lk+1,j + lj,i )

1. t ←⎯ min 1≤i≤k ( lk+1,i + li,k+1 )

Si t<0 alors arrêt : circuit négatif.

Sinon

Pour tout i et j allant de 1 à k faire

li,j ←⎯ min ( li,j ; li,k+1 + lk+1,j ).

Lorsque l’algorithme se termine, on a lij = longueur du plus court chemin allant de i à j si le graphe est sans circuit de longueur négative.

Il est clair que pour k=N, l’algorithme ne fournit pas le résultat attendu (nous devons mentionner qu’il a été pris de [Gon et al 1985] ).

**NB** : Mentionnons, à la fin, que les algorithmes de Ford, de Bellmann, matricielles, s’appliquent au cas de recherche d’un chemin maximal, moyennant certaines modifications appropriées (max à la place de min, +∞ ou 0 (selon les cas) à la place de −∞, …)

Exercices

# Exercice 1

Déterminer les chemins de longueur minimale entre x1 et x8 dans le graphe présenté ci-dessous par :

* la méthode de Ford,
* la méthode de Bellman,
* la méthode de Dijkstra,
* l’une des méthodes matricielles.

Reconsidérer le problème avec des valeurs unitaires associées aux arcs.

x4 1 x5 9

2128 x6

1

x1 6x2 4 2

x3

823x8

x7  7

**Exercice 2**

Déterminer les chemins de longueur maximale entre x1 et x8 dans le graphe présenté ci-dessous par :

* la méthode de Ford,
* la méthode de Bellman,
* la méthode de Dijkstra,
* l’une des méthodes matricielles.

Reconsidérer le problème avec des valeurs unitaires associées aux arcs.

x2 3 x5 2

545  x7

7

x1 3x3 8 6

x6 5

723x8

x4  9