Graphes

1 Définitions

1.1 Graphes

Définitions (*graphes non orientés et vocabulaire*) :

Un graphe non orienté est la donnée d'un couple $\mathcal{G} = (V, E)$ où V est un ensemble fini non vide et $E \subseteq \left\{\{x, y\} \,\middle|\, (x, y) \in V^2\right\}$.
· Les éléments de V sont appelés sommets du graphe.
· Les éléments de E sont appelés arêtes du graphes.

Si $e = \{x\} \in E$ avec $x \in V$, on dit que e est une boucle sur le sommet x.

Pour $(x, y) \in V^2$, on dit que x et y sont voisins ssi $\{x, y\} \in E$ et on appelle voisinage de x, noté $\mathcal{V}(x)$, l'ensemble de ses voisins :

$$\mathcal{V}(x) = \{y \in V \,|\, \{x, y\} \in E\}$$

Le degré de x est alors son nombre de voisins, on le note $d(x)$ ou $\deg(x)$:

$$\deg(x) = \text{Card}(\mathcal{V}(x))$$

Définitions (*graphes orientés et vocabulaire*) :

Un graphe orienté est la donnée de $\mathcal{G} = (S, A)$ où S est un ensemble fini non vide et $A \subseteq S^2$.
· Les éléments de S sont appelés sommets du graphe.
· Les éléments de A sont appelés arcs du graphes.

Si $a = (x, x) \in A$ avec $x \in S$, on dira de même que a est une boucle sur le sommet x.

Pour $(x, y) \in V^2$, on dit que : · x est prédécesseur de y ssi $(x, y) \in A$
· x est successeur de y ssi $(y, x) \in A$

Dans la suite de ce cours, on notera respectivement, pour $x \in S$ donné, $\mathcal{P}\text{red}(x)$ et $\mathcal{S}\text{ucc}(x)$ les ensembles des prédécesseurs et des successeurs de x. Ainsi :

$$\mathcal{P}\text{red}(x) = \{z \in S \,|\, (z, x) \in A\} \qquad \mathcal{S}\text{ucc}(x) = \{y \in S \,|\, (x, y) \in A\}$$

Le degré entrant de x (noté $d_+(x)$ ou $\deg_+(x)$) est alors son nombre de prédécesseurs, et son degré sortant (noté $d_-(x)$ ou $\deg_-(x)$) est son nombre de successeurs :

$$\deg_+(x) = \text{Card}(\mathcal{P}\text{red}(x)) \qquad \deg_-(x) = \text{Card}(\mathcal{S}\text{ucc}(x))$$

Remarque : Il convient de ne pas mélanger la terminologie des graphes non orientés avec celle des graphes orientés qui, contrairement à la première, tend à rendre apparente la notion de sens, ce qui se verra davantage encore dans les définitions qui suivront.

Propriété (*somme des degrés*) :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté sans boucle et $\mathcal{G}' = (S, A)$ un graphe orienté sans boucle. Alors, on a les propriétés suivantes :

i. $\displaystyle\sum_{x\in V} \deg(x) = 2\,\mathrm{Card}(E)$

ii. $\displaystyle\sum_{z\in S} \deg_-(z) = \sum_{z\in S} \deg_+(z) = \mathrm{Card}(A)$

Preuve :

i. Puisque par hypothèse $\mathcal{G}$ ne comporte pas de boucle, on a :

$$S_1 := \sum_{x\in V} \deg(x) = \sum_{x\in V}\sum_{y\in V} \mathbb{1}_{\{x,y\}\in E} \overset{hyp.}{=} \sum_{x\in V}\sum_{\substack{y\in V\\ x\neq y}} \mathbb{1}_{\{x,y\}\in E} = \sum_{\substack{(x,y)\in V^2\\ x\neq y}} \mathbb{1}_{\{x,y\}\in E}$$

Or en notant $\widetilde{E} = \left\{(x,y)\in V^2 \,|\, \{x,y\}\in E\right\}$, on a $\{x,y\}\in E$ ssi $(x,y),(y,x)\in\widetilde{E}$. D'où :

$$S_1 = 2\sum_{\substack{\{x,y\}\\ (x,y)\in V^2, x\neq y}} \mathbb{1}_{\{x,y\}\in E} = 2\sum_{\{x,y\}\in E} 1 = 2\,\mathrm{Card}(E)$$

ii.

1.2 Accessibilité, connexité

Pour cette section, on fixe $\mathcal{G} = (V, E)$ un graphe non orienté sans boucle et $\mathcal{H} = (S, A)$ un graphe orienté sans boucle.

Définition (*chaîne, chemin, cas élémentaire*) :

Soient $s = (s_i)_{i\in[0..n]} \in V^{n+1}$ et $t = (t_j)_{j\in[0..m]} \in S^{m+1}$. On dit que :

- s est une chaîne (de longueur n) de $\mathcal{G}$ ssi $\forall\, i \in [0..n-1]$, $\{s_i, s_{i+1}\} \in E$
- t est un chemin (de longueur m) de $\mathcal{H}$ ssi $\forall\, j \in [0..m-1]$, $(t_j, t_{j+1}) \in A$

Dans les deux cas, on dira que s_n ou t_m est accessible depuis s_0 ou t_0.
Plus précisément, dans le premier cas, s relie s_0 et s_n tandis que dans le second, t relie t_0 à t_m.

Si de plus $\forall\,(i,i') \in [0..n-1]^2 \cup [1..n]^2 = [0..n]^2 \backslash \{(0,n),(n,0)\}$ avec $i \neq i'$, on a $t_i \neq t_{i'}$, alors on dit que t est élémentaire. Cette définition s'adapte au cas des graphes non orientés.

Définition (*cycle, circuit*) :

Une chaîne $s = (s_i)_{i\in[0..n]}$ de $\mathcal{G}$ est appelée un cycle ssi $s_0 = s_n$.
De même, un chemin $t = (t_j)_{j\in[0..n]}$ de $\mathcal{H}$ sera appelé un circuit si $t_0 = t_m$.

Remarque : Il y a toujours un nombre fini de chaînes ou chemins élémentaires, mais dans le cas où le graphe admet un cycle, il y a un nombre infini de chaînes ou de cycles puisque l'on peut "tourner en rond" autant de fois que l'on veut.

Propriété :

On définit la relation binaire $\leftrightarrow$ sur V par :

$$\forall\,(x,y)\in V^2,\ x\leftrightarrow y \iff \exists\, s \text{ une chaîne reliant } x \text{ et } y$$

On définit de même une relation $\overleftrightarrow{}$ sur S par :

$$\forall\,(x,y)\in S^2,\ x\overleftrightarrow{}\,y \iff \begin{cases}\exists\, t \text{ un chemin reliant } x \text{ à } y\\ \exists\, t' \text{ un chemin reliant } y \text{ à } x\end{cases}$$

Alors, $\leftrightarrow$ et $\overleftrightarrow{}$ sont des relations d'équivalence : plus précisément, ce sont les clôtures réflexives transitives respectives de E et de A.

Preuve :

Définition (*composante connexe, composante fortement connexe*) :

On appelle composante connexe de $\mathcal{G}$ toute classe d'équivalence pour la relation $\leftrightarrow$.
On appelle de même composantes fortement connexes de $\mathcal{H}$ les classes d'équivalence pour $\overleftrightarrow{}$.

Définition (*connexité, connexité forte*) :

Si $\mathcal{G}$ admet une seule composante connexe, on dit que $\mathcal{G}$ est connexe.
Si $\mathcal{H}$ admet une seule composante fortement connexe, on dit que $\mathcal{H}$ est fortement connexe.

Remarque : Les composantes connexes d'un graphe forment une partition de l'ensemble de ses sommets, en tant qu'éléments de son espace quotient par une relation d'équivalence (*cf.* chapitre 8 – "ordre et induction").

Remarque : La définition de la connexité simple (*i.e.* non forte) pour les graphes orientés est ambiguë. Ainsi, le graphe suivant peut être connexe ou non selon la définition adoptée :

Définition (*partie connexe de l'ensemble des arêtes*) :

Pour $W\subseteq V$ non vide, on dit que W est connexe ssi $\forall\,(x,y)\in W^2,\ x\leftrightarrow y$.

Propriété (*caractérisation des composantes connexes*) :

Soit $W\subseteq V$, $W\neq\emptyset$. W est une composante connexe de $\mathcal{G}$ ssi W est connexe maximal, *c-à-d* :

$$\forall\, W'\subseteq V,\ W\subsetneq W' \Longrightarrow W' \text{ n'est pas connexe}$$

Définition (*sous-graphe*) :

Soit $\mathcal{G}'=(V',E')$ un graphe. $\mathcal{G}'$ est un sous-graphe de $\mathcal{G}$ ssi $V'\subseteq V$ et $E'\subseteq E$.

Définition (*graphe induit*) :

Soit $V'\subseteq V$. On appelle graphe induit par $\mathcal{G}$ sur V' le graphe :

$$\mathcal{G}'=\left(V',\left\{\{x,y\}\in E\,\middle|\,(x,y)\in (V')^2\right\}\right)$$

Ce graphe est alors facilement un sous-graphe de $\mathcal{G}$.

Propriété (*caractérisation de la connexité*) :

Soit $W\subseteq V$. W est connexe ssi le graphe qu'il induit est connexe.

1.3 Types de graphes

Définition (*graphe acyclique*) :

Un graphe non orienté (resp. orienté) est dit acyclique s'il ne contient aucun cycle (resp. circuit) de longueur supérieure ou égale à 2.

Définition (*arbres et forêts*) :

Un arbre est un graphe non orienté connexe acyclique.
Une forêt est un graphe acyclique décomposé en ses composantes connexes, qui sont des arbres.

Propriétés : *cf.* TD n°13, exercice 2 pour caractérisations.

Définition (*graphe biparti*) :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté. $\mathcal{G}$ est dit biparti ssi il existe une partition $\{W_1, W_2\}$ de V telle que toutes les arêtes de E ont une extrémité dans W_1 et l'autre dans W_2, c'est-à-dire :

$$E \subseteq \left\{\{x, y\} \in V^2 \,\middle|\, (x, y) \in W_1 \times W_2 \cup W_2 \times W_1\right\}$$

Illustration :

2 Parcours

2.1 Définitions

Définition (*bordure*) :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté. Pour $W \subseteq V$, on définit la bordure de W par :

$$\mathcal{B}(W) = \{v \in V \backslash W \mid \exists\, w \in W,\, v \in \mathcal{V}(w)\}$$

Pour $\mathcal{G}' = (S, A)$ un graphe orienté, on définit de même la bordure de $T \subseteq S$ par :

$$\mathcal{B}(T) = \{s \in S \backslash T \mid \exists\, t \in T,\, s \in \mathcal{S}\mathrm{ucc}(t)\}$$

Remarque : On obtient des définitions équivalentes en remplaçant $v \in \mathcal{V}(w)$ et $s \in \mathcal{S}\mathrm{ucc}(t)$ respectivement par $\{w, v\} \in E$ et $(t, s) \in A$.

Remarque : En particulier, $\mathcal{B}(\emptyset) = \emptyset$.

Définition (*parcours, points de regénération*) :

Soit $\mathcal{G} = (V, E)$ un graphe quelconque (non orienté ou orienté).
Soit $(L_i)_{i \in [1..n]} \in V^n$ une permutation des sommets de V (qui est donc de cardinal n).

On dit que L est un parcours de $\mathcal{G}$ ssi

$$\forall\, i \in [1..n],\, L_i \in \mathcal{B}\Big(\Big\{L_j \,\Big|\, j \in [1..i-1]\Big\}\Big) \text{ ou } \mathcal{B}\Big(\Big\{L_j \,\Big|\, j \in [1..i-1]\Big\}\Big) = \emptyset$$

De plus, on dit que L_i est un point de regénération du parcours si $\mathcal{B}(\{L_j \mid j \in [1..i-1]\}) = \emptyset$.

Définition (*forêt d'arborescences associée à un parcours*) :

Soit $\mathcal{G} = (V, E)$ un graphe et L un parcours de ce graphe. En notant $\mathcal{R}$ l'ensemble des points

de regénération de L, on dit que le graphe orienté $\mathcal{F} = (W, P)$ est une forêt d'arborescences associée au parcours L ssi :

- $\forall\, i \in [1..n]$, $L_i \in \mathcal{R}$ ou $(\exists\, j \in [1..i-1]$ tel que $L_i \in \mathcal{V}(L_j)$ (ou $\mathcal{S}\text{ucc}(L_j))$ et $(L_j, L_i) \in P)$
- $\forall\, u \in V \backslash \mathcal{R}$, $\exists !\, w \in V$ tel que $(w, u) \in P$: on dit alors que w est le père de u
- P est minimal parmi les ensembles vérifiant la première propriété.

Remarque : Dans la première propriété, on peut englober les cas non orienté et orienté dans une seule proposition en écrivant tout simplement "$L_i \in \mathcal{B}(\{L_j \,|\, j \in [1..i-1]\})$".

Remarque : La troisième propriété rend un peu redondante la deuxième.

Exemples :

Propriété :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté et $L = (L_i)_{i \in [1..n]}$ un parcours de $\mathcal{G}$ où $n = \text{Card}(V)$.

Si l'ensemble des points de regénération s'écrit $\{L_{r_k} \,|\, k \in [1..K]\}$ avec $(r_k)_{k \in [1..K]}$ croissante, alors $\mathcal{G}$ admet K composantes connexes, à savoir les $(C_k)_{k \in [1..K]}$ définies par :

$$\forall\, k \in [1..K],\ C_k = \{L_i \,|\, i \in [r_k..r_{k+1} - 1]\}$$

où on a noté $r_{K+1} = n + 1$.

La preuve de cette propriété va nécessiter le lemme suivant.

Lemme : Soit $\mathcal{G} = (V, E)$ un graphe non orienté et $W \subseteq V$. Si $\mathcal{B}(W) = \emptyset$, alors il n'existe aucune chaîne reliant un sommet de W et un sommet de $V \backslash W$.

Preuve :

- Si $W = \emptyset$, c'est évident.
- Sinon, supposons par l'absurde qu'il existe une chaîne γ de longueur ℓ telle que : $\begin{cases} \gamma_0 \in W \\ \gamma_\ell \in V \backslash W \end{cases}$.

On a clairement $\gamma_0 \neq \gamma_\ell$ donc $\ell > 0$. On peut alors définir $i_0 = \min\{i \in [1..\ell] \,|\, \gamma_i \notin W\}$ puisque cet ensemble contient au moins ℓ, donc est non vide.

Par définition de i_0, $\gamma_{i_0 - 1} \in W$ et par définition d'une chaîne, $\{\gamma_{i_0-1}, \gamma_{i_0}\} \in E$ c'est-à-dire $\gamma_{i_0} \in \mathcal{V}(\gamma_{i_0-1})$. Ainsi on a $\gamma_{i_0} \in \mathcal{B}(W)$ ce qui est absurde.

Preuve de la propriété :
Soit $k \in [1..K]$, montrons que C_k est connexe maximal.

Considérons donc $u \in V \backslash C_k$ et montrons que $C_k \sqcup \{u\}$ n'est pas connexe.
Par définition d'un parcours, il existe $i_u \in [1..n]$ tel que $u = L_{i_u}$. Comme $u \notin C_k$, $i_u \notin [r_k..r_{k+1}[$. Alors, il faut distinguer deux cas :

- Si $i < r_k$, posons $W = \{L_i \,|\, i \in [1..r_k[\}$. On a $L_{i_u} \in W$ et $\mathcal{B}(W) = \emptyset$ puisque L_{r_k} est un point de regénération. D'après le lemme, il n'existe aucun chemin reliant L_{i_u} et $L_{r_k} \notin W$, donc par caractérisation de la connexité $C_k \sqcup \{u\}$ n'est pas maximal.

- Si $i \geq r_{k+1}$, on pose $W = \{L_i \,|\, i \in [1..r_{k+1}[\}$. On a clairement $C_k \subseteq W$ donc $L_{r_k} \in W$. De plus, $\mathcal{B}(W) = \emptyset$: en effet :
  - si $k < K$, alors $L_{r_{k+1}}$ est un point de regénération donc $\mathcal{B}(W) = \emptyset$
  - si $k = K$, $\mathcal{B}(W) = \mathcal{B}(\{L_i \,|\, i \in [1..n+1[\}) = \mathcal{B}(V) = \emptyset$ puisque $\mathcal{B}(V) \subseteq V \backslash V = \emptyset$.

D'après le lemme, on ne peut donc pas trouver de chemin reliant L_{r_k} et $L_{i_u} \notin V \backslash W$.

Ainsi, s'il est connexe, C_k est nécessairement connexe maximal.

Montrons à présent que C_k est bien connexe : on suppose par l'absurde qu'il ne l'est pas. Par transitivité, on peut alors dire qu'il existe $i \in]r_k..r_{k+1}[$ tel que $L_{r_k} \not\leftrightarrow L_i$. Soit alors :

$$i_0 = \min\left\{i \in]r_k..r_{k+1}[\,\middle|\, L_{r_k} \not\leftrightarrow L_i\right\}$$

Par minimalité de i_0, $\forall\, j \in]r_k..i_0[$, $L_{r_k} \leftrightarrow L_j$, ce qui implique que $\forall\, j \in [r_k..i_0[$, $L_{i_0} \not\leftrightarrow L_j$ (sinon, on aurait $L_{i_0} \leftrightarrow L_{r_k}$ par transitivité). De plus comme $L_{r_k} \in \mathcal{R}$, $\mathcal{B}(\{L_j \,|\, j \in [1..r_k[\}) = \emptyset$ donc d'après le lemme, $\forall\, j \in [1..r_k[, L_{i_0} \not\leftrightarrow L_j$.

Ainsi, $\forall\, j \in [1..i_0[$, $L_{i_0} \not\leftrightarrow L_j$, en particulier $\{L_j, L_{i_0}\} \notin E$. Donc $L_{i_0} \notin \mathcal{B}(\{L_j \,|\, j \in [1..r_k-1]\})$.

Par définition d'un parcours, on a alors nécessairement $\mathcal{B}(\{L_j \,|\, j \in [1..r_k - 1]\}) = \emptyset$ donc $L_{i_0} \in \mathcal{R}$, ce qui est absurde puisque par hypothèse L_{r_k} et $L_{r_{k+1}}$ sont des points de regénération consécutifs.

Définition (*sommets ouverts, parcours en largeur, parcours en profondeur*) :

Soit $\mathcal{G} = (V, E)$ un graphe orienté et $L = (L_i)_{i \in [1..n]}$ un parcours de $\mathcal{G}$.

Pour $k \in [1..n]$, on appelle sommet ouvert à l'étape k (dans L) un sommet de l'ensemble

$$\mathcal{O}_k = \left\{L_j \,|\, j \in [1..k[\text{ et } \mathcal{V}(L_j) \not\subseteq \{L_i \,|\, i \in [1..k[\}\right\}$$

Alors, on dit que :

- L est un parcours en largeur de $\mathcal{G}$ ssi
  $$\forall\, k \in [1..n],\ L_k \in \mathcal{R} \text{ ou bien } L_k \in \mathcal{V}(L_{i_0}) \text{ où } i_0 = \min\{i \in [1..n] \,|\, L_i \in \mathcal{O}_k\}$$
- L est un parcours en profondeur de $\mathcal{G}$ ssi
  $$\forall\, k \in [1..n],\ L_k \in \mathcal{R} \text{ ou bien } L_k \in \mathcal{V}(L_{i_0}) \text{ où } i_0 = \max\{i \in [1..n] \,|\, L_i \in \mathcal{O}_k\}$$

Remarque : Autrement dit, L est un parcours en largeur ssi chaque sommet du parcours est point de regénération ou bien voisin du premier sommet ouvert, et c'est un parcours en profondeur ssi chaque sommet est point de regénération ou voisin du dernier sommet ouvert.

En fait, on observe une similarité entre ces notions de parcours pour les graphes et pour les arbres (*cf.* chapitre 9 – "structures de données arborescentes").

Remarque : Lorsqu'on s'intéresse à la forêt d'arborescences associée à un parcours en largeur/profondeur, on choisira comme père d'un sommet L_k le premier/dernier sommet ouvert à l'étape k. La forêt justifiera donc que le parcours est en largeur/profondeur.

Exemple :

2.2 Algorithmes de parcours

2.2.1 Détection de composantes connexes

On propose ici un algorithme qui consiste à associer à chaque sommet du graphe pris en entrée un numéro qui représentera la composante connexe à laquelle il appartient.

Il nous faudra pour cela introduire un ensemble $\mathcal{O}$ des sommets ouvert à l'étape courant, que l'on n'a pas fini de traiter, ainsi qu'un ensemble $\mathcal{F}$ des fermés, sommets que l'on a fini de traiter et qu'on ne traitera donc plus.

Algorithme – Comp_connexe

entrée : $\mathcal{G} = (V, E)$ avec $V = [1..n]$ un graphe non orienté
sortie : T_{res} un tableau contenant pour chaque sommet, le numéro de sa composante connexe

```
n = nb_sommets(G)
O, F = ensembles de sommets initialement vides
T_res = tableau indicé par [1..n], initialisé à −1
n_c = 0    // numéro de la composante connexe courante
racine = 1
T_res[racine] ← n_c
O.ajouter(racine)
i = 0
Tant que i < n
    Si O est vide
        racine ← un sommet s tel que T_res[s] = −1
        n_c ← n_c + 1
        O.ajouter(racine)
        T_res[racine] ← n_c
    Sinon
        u = un sommet extrait de O    // comprendre : il n'y est plus après l'instruction
        Pour tout v voisin de u
            Si T_res[v] = −1 alors
                O.ajouter(v)
                T_res[v] ← n_c
        F.ajouter(u)
        i ← i + 1
Retourner T_res
```

Remarque : En réalité, l'ensemble $\mathcal{F}$ n'est pas indispensable à l'algorithme. Néanmoins, il s'avère utile lors que l'on cherche à établir des invariants en vue d'en montrer la correction.

Ce constat pourra également se reproduire dans les algorithmes qui suivront.

2.2.2 Détection de graphes bipartis

2.2.3 Plus court chemin en nombre d'arcs

Définition (*distance dans un graphe*) :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté et $(a, b) \in V^2$. La distance entre a et b, notée $\text{dist}(a, b)$, est la longueur minimale d'une chaîne reliant a et b, ou bien $+\infty$ s'il n'en existe pas.

Pour $\mathcal{G}' = (S, A)$ un graphe orienté et $(a', b') \in S^2$, on définit de même la distance entre (a', b'), notée toujours $\text{dist}(a, b)$, comme la longueur minimale d'un chemin reliant a' à b' et $+\infty$ s'il n'existe pas de telle chaîne.

Exemple :

Remarque : Attention, la distance dans un graphe orienté n'est pas une distance au sens mathématique puisqu'on n'a pas toujours $\text{dist}(a, b) = \text{dist}(b, a)$.

Voici maintenant un algorithme qui permet d'obtenir un tableau contenant les distances de chaque sommet d'un graphe orienté par rapport à un sommet quelconque choisi.

Comme précédemment, nous stockerons à chaque étape les ouverts et les fermés, mais les ouverts nécessiteront cette fois-ci une structure de file, que l'on "implémente" à l'aide de deux piles.

Algorithme – PCCNA

entrée : $\mathcal{G} = (S, A)$ un graphe orienté avec $S = [1..n]$
$(a, b) \in S^2$

$n = \mathsf{nb\_sommets}(\mathcal{G})$
$\mathcal{O}_1, \mathcal{O}_2 =$ piles d'entiers initialement vides
$\mathcal{F} =$ ensemble initialement vide
$D, P =$ tableaux d'entiers indicés par $[1..n]$, initialisés respectivement à $+\infty$ et -1
prof $= 0$
$\mathcal{O}_2.\mathsf{ajouter}(a)$
$D[a], P[a] \leftarrow 0, a$
Tant que ("$\mathcal{O}_1 \cup \mathcal{O}_2$" $\neq \emptyset$)
  Si $\mathcal{O}_1$ est vide alors
    transvaser $\mathcal{O}_2$ dans $\mathcal{O}_1$
    prof $=$ prof $+ 1$
  $u = \mathcal{O}_1.\mathsf{extraire\_sommet}$
  Pour tout v successeur de u
    Si $D[v] = +\infty$ alors
      $D[v] \leftarrow D[u] + 1$
      $P[v] \leftarrow u$
      $\mathcal{O}_2.\mathsf{ajouter}(v)$
  $\mathcal{F}.\mathsf{ajouter}(u)$
Retourner $D[b]$

Remarque : On effectue dans cet algorithme un parcours en largeur de la composante connexe contenant le sommet de départ, ce qui évite, lorsque l'on cherche à affecter une distance à un sommet, de devoir effectuer des comparaisons à des distances déjà existantes. Cela justifie que l'on ait eu besoin d'une file pour les ouverts.

Grâce au tableau P stockant les pères, il est également possible de construire et de renvoyer la liste d'entiers correspondant à un chemin possible de longueur minimale entre a et b :

Si $D[b] < +\infty$ alors
  $w = b$
  $\gamma =$ liste d'entiers initialement vide
  Tant que $P[w] \neq w$ *// ou bien $w \neq a$*
    $\gamma.\mathsf{ajoute\_début}(w)$
    $w \leftarrow P[w]$
  $\gamma.\mathsf{ajoute\_début}(a)$
Retourner γ

3 Plus courts chemins

3.1 Graphes pondérés

Définition (*pondération*) :

Soit $\mathcal{G} = (V, E)$ un graphe non orienté. On dit que w est une pondération des arêtes ssi $w \in \mathcal{F}(E, \mathbb{R})$ et on dira alors que $\mathcal{G} = (V, E, w)$ est un graphe pondéré.

Pour $\mathcal{G}' = (S, A)$ un graphe orienté, w est une pondération des arcs ssi $w \in \mathcal{F}(A, \mathbb{R})$ et dans ce cas, $\mathcal{G}' = (S, A, w)$ sera aussi appelé un graphe pondéré.

Définition (*longueur et distances dans un graphe pondéré*) :

Soit $\mathcal{G} = (V, E, \omega)$ un graphe pondéré non orienté.

Si $\gamma = (\gamma_i)_{i \in [0..n]}$ est une chaîne de $\mathcal{G}$, sa longueur est :

$$\text{long}(\gamma) = \sum_{i=0}^{n-1} \omega(\{\gamma_i, \gamma_{i+1}\})$$

On définit alors la distance entre deux sommets de V par :

$$d(a, b) = \min \{\text{long}(\gamma) \,|\, \gamma \text{ est une chaîne reliant } a \text{ et } b\}$$

Remarque : Bien sûr, ces deux définitions se généralisent aussi au cas des graphes orientés.

3.2 Algorithme de Dijkstra

Algorithme – Dijkstra

entrée : $\mathcal{G} = (V, E, \omega)$ un graphe orienté, pondéré positivement
$s \in S$ où $S = [1..n]$

$n = \mathsf{nb\_sommets}(\mathcal{G})$
$D, P =$ tableaux indexés par $[1..n]$, initialisés à $+\infty$ et -1 respectivement
$D[s], P[s] \leftarrow 0, s$
$\mathcal{O}, \mathcal{F} = \{s\}, \emptyset$
Tant que $\mathcal{O} \neq \emptyset$
  $u =$ extraire de $\mathcal{O}$ un élément minimisant D *// donc* $u = \mathbf{argmin}_{i \in \mathcal{O}} D[i]$
  Pour $v \in \mathcal{S}\text{ucc}(u)$
    Si $D[v] = +\infty$ alors *// accessible en* $\Theta(1)$
      $\mathcal{O}.\mathsf{ajouter}(v)$
    Si $D[u] + \omega(u, v) < D[v]$ alors
      $D[v] \leftarrow D[u] + \omega(u, v)$
      $P[v] \leftarrow u$
  $\mathcal{F}.\mathsf{ajouter}(u)$
Retourner D (ou P)

Propriété :

Soit $\mathcal{G} = (S, A, \omega)$ un graphe orienté pondéré positivement et $s \in S$. Le tableau D retourné par $\mathsf{Dijkstra}(\mathcal{G}, s)$ est tel que :

$$\forall\, u \in S,\ D[u] = d_\omega(s, u)$$

d_ω étant la distance pour la pondération ω.

Preuve :
cf. annexe "scan Dijkstra".

3.3 Une première résolution par programmation dynamique

Soit $\mathcal{G} = (S, A, \omega)$ un graphe orienté pondéré positivement avec $S = [1..n]$.

Pour $(i,j) \in S^2$ et $\ell \in \mathbb{N}$, on note :

- $M_{i,j}^{\ell} = \min\{\text{long}_\omega(\gamma) \,|\, \gamma \text{ est un chemin de } i \text{ à } j \text{ ayant au plus } \ell \text{ arcs}\}$
- $N_{i,j}^{\ell} = \min\{\text{long}_\omega(\gamma) \,|\, \gamma \text{ est un chemin de } i \text{ à } j \text{ ayant exactement } \ell \text{ arcs}\}$

Propriétés :

On a alors les quatre propriétés suivantes :

i. $\forall (i,j) \in S^2,\ M_{i,j}^0 = N_{i,j}^0 = \begin{cases} 0 \text{ si } i = j \\ +\infty \text{ sinon} \end{cases}$

ii. $\forall (i,j) \in S^2,\ N_{i,j}^1 = \begin{cases} \omega(i,j) \text{ si } (i,j) \in A \\ +\infty \text{ sinon} \end{cases}$

iii. $\forall (i,j) \in S^2,\ M_{i,j}^1 = \begin{cases} 0 \text{ si } i = j \\ \omega(i,j) \text{ si } (i,j) \in A \\ +\infty \text{ sinon} \end{cases}$

iv. Pour $\ell \geq 2$ et $(i,j) \in S^2$, on a :

- $N_{i,j}^{\ell} = \min\limits_{\substack{s \in S \\ k \in [1..\ell-1]}} N_{i,s}^k + N_{s,j}^{\ell-k} = \min\limits_{u \in \mathcal{S}\text{ucc}(i)} \omega(i,u) + N_{u,j}^{\ell-1} = \min\limits_{u \in [1..n]} N_{i,u}^1 + N_{u,j}^{\ell-1}$
- $M_{i,j}^{\ell} = \min\left(M_{i,j}^{\ell-1}, \min\limits_{u \in \mathcal{S}\text{ucc}(i)} \omega(i,u) + M_{u,j}^{\ell-1}\right) = \min\left(M_{i,j}^{\ell-1}, \min\limits_{v \in \mathcal{P}\text{red}(j)} M_{i,v}^{\ell-1} + \omega(v,j)\right)$

Définissons pour $(A,B) \in \left((\mathbb{R}^+ \cup \{+\infty\})^{n^2}\right)^2$ la loi interne $\otimes$ par :

$$\forall (i,j) \in [1..n]^2,\ (A \otimes B)_{i,j} = \min_{k \in [1..n]} A_{i,k} + B_{k,j}$$

Alors, la première partie de la propriété ***iv.*** se reformule comme suit :

$$\forall \ell \geq 2,\ \forall (i,j) \in S^2,\ N_{i,j}^{\ell} = (N^{\ell})_{i,j} = (N^1 \otimes N^{\ell-1})_{i,j}$$

Remarque : Si l'on acceptait de définir des matrices à coefficients dans un ensemble qui n'est pas un corps (ni même un anneau), on pourrait écrire "$\mathcal{M}_n(\overline{\mathbb{R}^+})$" à la place de $(\mathbb{R}^+ \cup \{+\infty\})^{n^2}$ mais on perd dans ce cas la stucture d'algèbre pour $(\mathcal{M}_n(\overline{\mathbb{R}^+}), +, \otimes, \cdot)$.

Néanmoins, il est intéressant de voir le parallèle que l'on peut faire avec $(\mathcal{M}_n(\mathbb{K}), +, \times, \cdot)$ si $(\mathbb{K}, +, \times)$ est un corps puisque dans $(\overline{\mathbb{R}^+}, \min, +)$:

- on conserve la propriété de distributivité puisque $a + \min(b,c) = \min(a+b, a+c)$
- le neutre pour min, $+\infty$, est absorbant pour $+$
- la définition de $\otimes$ dans $\mathcal{M}_n(\overline{\mathbb{R}^+})$ est alors analogue à celle de $\times$ dans $\mathcal{M}_n(\mathbb{K})$
- on a enfin une matrice "identité"

3.4 Algorithme de Floyd-Warshall