925

Graphes : représentations et algorithmes.

Intérêt : modéliser des réseaux, des intéractions

I. Divers types de graphes

Déf 1 (Graphe orienté). Un graphe G est un couple $(\mathcal{S}, \mathcal{A})$ où $\mathcal{S}$ est un ensemble fini de sommets, et $\mathcal{A} \subset \mathcal{S}^2$ est son ensemble d'arrêtes.

Ex 1 $\mathcal{S} = \{1,2,3,4\}$, $\mathcal{A} = \{(1,2),(1,3),(2,4),(4,1)\}$ (Fig. 1) $:= G_1$

Déf 2 (Graphe transposé) Soit $G = (\mathcal{S},\mathcal{A})$. On définit ${}^TG = (\mathcal{S}, {}^T\mathcal{A})$ avec ${}^T\mathcal{A} = \{(v,u), (u,v) \in \mathcal{A}\}$

Ex ${}^TG_1 = (\{1,2,3,4\}, \{(2,1),(3,1),(4,2),(1,4)\})$

Déf 3 (Graphe non orienté) $G = (\mathcal{S},\mathcal{A})$ est non orienté si $G = {}^TG$

Notations : pour $(u,v) \in \mathcal{S}^2$, on note $u \xrightarrow{G} v$ s'il existe $x_1 = u, x_2, \dots, x_{n-1}, x_n = v$ / $(x_i, x_{i+1}) \in \mathcal{A}$ $\forall i \in [\![1,n-1]\!]$

Représentations : soit $G = (\mathcal{S},\mathcal{A})$, supposons $\mathcal{S} = [\![1,n]\!]$. Il en existe deux principales :

1) Par matrice d'adjacence : $M_G = (a_{ij})_{1 \le i,j \le n}$ où $a_{ij} = \mathbb{1}_{\mathcal{A}}((i,j))$ **Ex** $M_{G_1} = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$, $M_{{}^TG} = {}^TM_G$

place en mémoire : $\Theta(|\mathcal{S}|^2)$

2) Avec un tableau T où $T[i] = \{j \in \mathcal{S} / i \xrightarrow{G} j\}$ (mémoire : $\Theta(|\mathcal{S}| + |\mathcal{A}|)$)

Ex : $T[1] = [2;3]$, $T[2] = [4]$, $T[3] = []$, $T[4] = [1]$

Déf 4 (Graphe pondéré) : Soit $G = (\mathcal{S},\mathcal{A})$ et $w : \mathcal{A} \to \mathbb{R}$. Alors $G_w = (\mathcal{S},\mathcal{A},w)$ est un graphe pondéré par w.

Ex G_w (fig. 2) $\mathcal{S} = \{$Lyon, Metz, Reims, Rennes$\}$, $w(L,M) = 4h, \dots$

Déf 5 (Graphe connexe) : $G = (\mathcal{S},\mathcal{A})$ est connexe si $\forall (i,j) \in \mathcal{S}^2$, $i \xrightarrow{G} j$ ou $j \xrightarrow{G} i$

- fortement connexe si $\forall (i,j)$, $i \xrightarrow{G} j$ et $j \xrightarrow{G} i$

Déf 6 (Cycle) Si $x_1 \to x_2 \to \dots \to x_p \to x_1$, on dit que G possède un cycle

Déf 7 (Arbre). Un arbre est un graphe connexe acyclique (ie sans cycle) Ex : Fig 3

Déf 8 (Graphe d'une chaîne de Markov) Soit $(\Omega, \mathbb{P})$ une tribu probabilisée, X_n une variable aléatoire à valeurs dans $[\![1,n]\!]$ /

$$\mathbb{E}(X_{n+1} | \sigma(X_1 \dots X_n)) = \mathbb{E}(X_{n+1} | \sigma(X_n))$$

On définit $M(X) = (m_{ij})_{1 \le i,j \le n}$ où $m_{ij} = \mathbb{P}(X_{n+1} = j | X_n = i)$

Ex Fig 4 : $M(X) = \begin{pmatrix} 0 & 1/2 & 1/2 \\ 1/3 & 0 & 2/3 \\ 1/10 & 1/10 & 4/5 \end{pmatrix}$

II. Algorithmes de parcours.

Déf (1) (profondeur) Soit $s \in \mathcal{S}$, $x \in \mathcal{S}$. La profondeur de x depuis s est le plus petit entier n / $s \to x_1 \to \dots \to x_n = x$ ($+\infty$ si $s \not\to x$)

Déf 2 (parcours en —)

- Largeur Entrée $G = (\mathcal{S},\mathcal{A})$ $s \in \mathcal{S}$
  Sortie : liste des sommets accessibles depuis s, par profondeur croissante
- Profondeur : Entrée $G = (\mathcal{S},\mathcal{A})$, $s \in \mathcal{S}$
  Sortie : liste des sommets successifs sur les branches issues de s.

```
PL (G, s)
Pour tout u ≠ s, faire  couleur[u] ← blanc
                        d[u] ← +∞
                        père[u] ← ∅
couleur s ← gris
d[s] ← 0
père[s] ← ∅
F ← {s}                 (on enfile s)
Tant que F ≠ ∅ faire
  u ← tête (F)
  Pour tout v / u → v
    | Si couleur[v] = blanc
    |   | alors couleur[v] ← gris
```

$d[v] \leftarrow d[u]+1$
$père[v] \leftarrow u$
$Enfile(F, v)$

Défiler F

Ex: Fig 5

couleur $[u] \leftarrow$ noir

Complexité : $\Theta(|\mathcal{S}|+|\mathcal{A}|)$ avec liste d'adjacence, $\Theta(|\mathcal{S}|^2)$ avec matrice

Application : plus court chemin en terme de nombre d'arrêtes

Visiter PP(u) : couleur $(u) \leftarrow$ gris
date $\leftarrow$ date+1
date$[u] \leftarrow$ date
Pour tout v / $u \to v$
  Si couleur $[v]$ = blanc alors
    père $[v] = u$
    Visiter - PP(v)
couleur$[u] \leftarrow$ noir

Applications : - tri - topologique, lorsque G est acyclique (DVPT)

- Composantes fortement connexes

III : Recherche du plus court chemin

Problème : étant donné un graphe $G_w = (\mathcal{S}, \mathcal{A}, w)$, un couple $(u,v) \in \mathcal{S}^2$, trouver un chemin $u \to x_2 \to \dots \to x_n = v$ / $\sum_{i=0}^{n-1} w(x_i, x_{i+1})$ soit minimal (si cela est possible).

1] Algorithme de Dijkstra

Entrée $s \in \mathcal{S}$ $w : \mathcal{A} \to \mathbb{R}^+$

Sortie un tableau d / $d[v] = \inf \left\{ \sum_{i=1}^{n} w(x_i, x_{i+1}) \middle/ \underset{x_1}{\overset{\shortparallel}{s}} \to x_2 \to \dots \to \underset{x_n}{\overset{\shortparallel}{u}} \right\}$

$E \leftarrow \emptyset$
$F \leftarrow \mathcal{S}$
tant que $F \neq \emptyset$ faire
  u = extraire-min (F)
  $E \leftarrow E \cup \{u\}$
  Pour chaque v / $u \to v$
    relâcher (u, v, w)

Où relâcher (u, v, w) = si $d[v] > d(u) + w(u,v)$
  alors $d[v] \leftarrow d(u) + w(u,v)$
  père $(v) \leftarrow u$

Complexité : a priori $O(|\mathcal{S}|^2 + |\mathcal{A}|) = O(|\mathcal{S}|^2)$, mais si le graphe est peu dense, on peut affiner.

2] Algorithme de Bellman-Ford

(→ prog linéaire voir Cormen)

Entrée : $G = (\mathcal{S}, \mathcal{A}, w)$ $w : \mathcal{A} \to \mathbb{R}$, $s \in \mathcal{S}$.

Sortie : vrai ssi il n'existe pas de cycle contenant s de poids strictement négatif. Dans ce cas, on connaît $d[v]$ la distance de s à v $\forall v \in \mathcal{S}$.

Principe : - on applique $|\mathcal{S}|$ fois le relâchement à tous les sommets (en initialisant à $d[s]=0$ $d[u] = +\infty$, père$[u]$ = Nil)
- Si l'on peut encore effectuer un relâchement, alors il y a un cycle de poids négatif (s est dans ce cycle).
- Sinon : il n'y en a pas et la table d donne les distances depuis s.

Complexité : $\Theta(|\mathcal{S}|\,|\mathcal{A}|)$ (avec listes d'adjacence)

3] Algorithme de Floyd-Warshal (DVPT)

Entrée : $G = (\mathcal{S}, \mathcal{A}, w)$

Sortie : matrice $(d_{ij})_{1 \leq i,j \leq n}$ où d_{ij} est la distance minimal entre i et j.

Principe : on réalise $|\mathcal{S}|$ étapes avec la propriété suivante :
À l'étape k, d_{ij}^k correspond au poids minimal d'un chemin de i vers j n'empruntant que les k premiers sommets de $\mathcal{S}$.

Complexité : $\mathcal{O}(|\mathcal{S}|^3)$

IV. Arbres couvrants minimaux

Déf 1 (Arbre couvrant) $G=(\mathcal{S},\mathcal{A})$ connexe.
$T\subset\mathcal{A}$ est couvrant si $\forall x\in\mathcal{S}, \exists y\in\mathcal{S} / (x,y)$ ou $(y,x)\in T$.
Ex : Fig 6 : T est un arbre couvrant.

Déf 2 (Poids d'un sous-ensemble) Pour $T\subset\mathcal{A}$ on note
$$w(T)=\sum_{e\in T} w(e)$$

Déf 3 (Arbre couvrant minimal) : il s'agit d'un arbre couvrant de poids minimal.

Algorithme de Kruskal : (glouton) ajouter des arrêtes de poids minimaux qui ne créent pas de cycles, jusqu'à obtenir un arbre couvrant.

Généralisation : si $V\subset\mathcal{S}$, un arbre de Steiner est un ensemble $T\subset\mathcal{A}$ / T recouvre V.

V. Maximisation de flots

Déf 1 (Réseau de transport). $G=(\mathcal{S},A)$ on note c (Si $(u,v)\notin\mathcal{A}$, $c(u,v)=0$.) $\mathcal{S}$ contient deux sommets particuliers
- s : la source.
- t : le puit.

Déf 2 (flot). C'est une fonction $f:\mathcal{S}^2\to\mathbb{R} / \forall(u,v)\in\mathcal{S}^2$
- $f(u,v)\leq c(u,v)$ (capacité)
- $\forall u\notin\{s,t\}, \sum_{v\in\mathcal{S}} f(u,v)=0$
- $\forall(u,v)$ $f(u,v)=-f(v,u)$. Ex Fig 7

Prop : tout réseau à puits et sources multiples est équivalent à un réseau à puit et source unique.

Déf (Réseau résiduel). $G_f=(\mathcal{S},\mathcal{A}_f)$ où f est un flot.
$$c_f(u,v)=c(u,v)-f(u,v) \quad \mathcal{A}_f=\{(u,v) / c_f(u,v)>0\}$$

Algorithme de Ford-Fulkerson

```
Faire f[u,v]=0
 tant qu'il existe un chemin p / s -f-> t faire
   cf(p) <- min { cf(u,v), (u,v) ∈ p }
   pour chaque arc (u,v) ∈ p
        faire f[u,v] <- f[u,v] + cf(p)
              f[v,u] <- - f[u,v]
```

Prop (admise) l'algorithme termine pour des capacités entières.

VI. Problèmes N.P-complets

Déf (Clique) Soit $G=(\mathcal{S},\mathcal{A})$ et $k\in\mathbb{N}$. Une clique de taille k est un sous-ensemble $V\subset\mathcal{S}$ / Card $V=k$ et $\forall(u,v)\in V^2, (u,v)\in\mathcal{A}$.
Ex : Fig 8 : clique de taille 4.

Déf (Ensemble indépendant) un ensemble indépendant de taille k est un sous-ensemble $V\subset\mathcal{S}$ / Card $V=k$ et $\forall(u,v)\in V^2, (u,v)\notin\mathcal{A}$.

Prop : V est un ensemble indépendant de taille k $\Leftrightarrow$ V est une clique de taille $|\mathcal{S}|-k$ dans ${}^{T}G$.

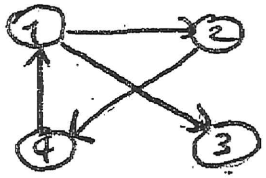
Prop : ensemble indépendant et Clique sont N.P complets.

Problèmes de coloriages : si $G=(\mathcal{S},\mathcal{A})$ un k-coloriage de G est une fonction $f:\mathcal{S}\to[\![1,k]\!]$ /
$\forall(u,v)\in\mathcal{A}, f(u)\neq f(v)$

Prop :
- pour $k=2$ le problème "existe-t-il un k-coloriage" est simple
- pour $k\geq 3$: il est NP-complets

Thm (4 couleurs) si G est un graphe planaire, alors il existe un 4-coloriage de G.

Fig 1



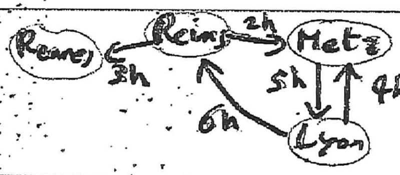


Fig 2

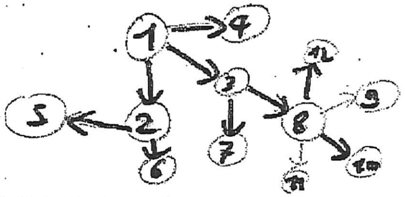


Fig 3

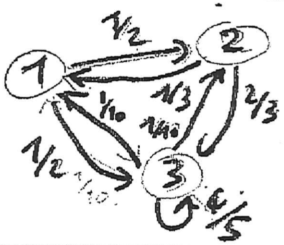


Fig 4

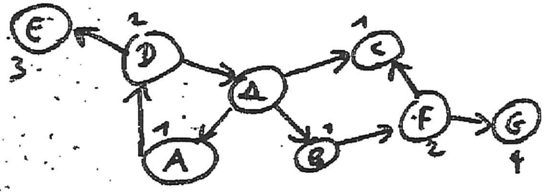


fig 5.

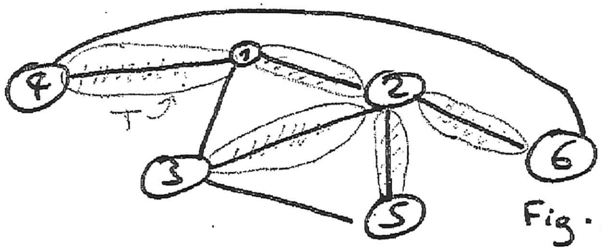


Fig. 6

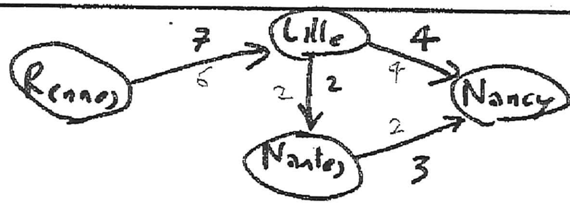


Fig 7.

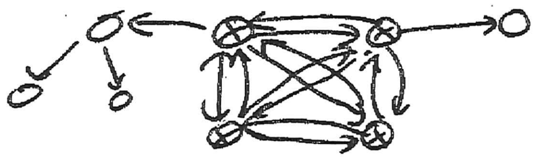


Fig 8.