03/06 2015

925

(I) Définitions élémentaires, représentations

Définition 1 : un graphe (orienté) est un couple (S, A) où $A \subset S^2 \setminus \Delta(S)$
S est l'ensemble des sommets, A des arêtes

Exemple 2 : $G_1 = (S_1, A_1)$ où $S_1 = [\![0;4]\!]$ et $A_1 = \{(0,1),(1,4),(4,2),(2,0),(0,2)\}$, voir annexe

Définition 2 : $G = (S,A)$ est non orienté si $\forall (u,v) \in A, (v,u) \in A$. Il est parfois noté (G, A'), où $A' = \{\{u,v\} / (u,v) \in A\}$.

Notation : on note $u \to v$ si $(u,v) \in A$

Définition 3 : un chemin de u à v dans G est une suite $(u_i)_{0 \le i \le n}$ finie ~~telle que~~ ; $u_0 = u$, $u_n = v$ et pour $0 \le i \le n-1$, $u_i \to u_{i+1}$. n est appelé longueur du chemin.
On note $u \to^* v$ s'il existe un chemin de u à v ; $u \to^n v$ s'il en existe un de longueur n

- Implémentation des graphes : il en existe 2 principales :

A) par matrice d'adjacence $M_G = (a_{ij})_{1 \le i \le n, 1 \le j \le n}$, où $a_{ij} = 1$ si $i \to j$, 0 sinon
(en identifiant S à $[\![1, |S|]\!]$)

Ex : $M_{G_1} = \begin{pmatrix} 0&1&1&0&0 \\ 0&0&0&0&1 \\ 1&0&0&0&0 \\ 0&0&0&0&0 \\ 0&0&1&0&0 \end{pmatrix}$
accès à l'information $u \to v$ en $O(1)$
complexité spatiale $O(|S|^2)$
à privilégier pour les graphes denses ($|A|$ proche de $|S|^2$)

B) par listes d'adjacence $T_G = (\{v / u_i \to v\})_{1 \le i \le n}$
où l'ensemble est représenté par la liste de ses éléments
complexité spatiale $O(|S| + |A|)$, accès à l'info $u \to v$ en $O(|\{w / u \to w\}|)$
à privilégier pour les graphes peu denses ~~[illegible]~~

Définition 4. G est connexe si $\forall (i,j) \in S^2$, $i \to^* j$ ou $j \to^* i$
fortement connexe si — // — et — // —

Définition 5 : un cycle de G est un chemin $u \to^* u$ de longueur $n \geq 1$
G est cyclique s'il possède un cycle, acyclique sinon (cf annexe)

Définition 6 : un arbre est un graphe connexe acyclique
exemple en annexe

Définition 7 : soit $u \in S$, le degré de u est $\deg(u) := |\{v \in S / u \to v\}|$

Proposition 8 : la somme des degrés des éléments de S est paire lorsque G est non orienté

Application 9 : preuve du théorème de Brouwer en dimension 2 : si Δ est un triangle, toute application continue de Δ dans Δ a un point fixe (voir annexe

Définition 10 : un graphe pondéré est un couple (G, w) où $G = (S,A)$ est un graphe, et $w : A \to \mathbb{N}$ (parfois $\mathbb{R}$) une fonction poids

(II) Algorithmes de parcours

Définition 11 : la profondeur de $x \in S$ depuis $u \in S$ est l'entier (éventuellement $+\infty$) $\min(n \in \mathbb{N}^* / u \to^n x)$

Définition 12 : le parcours en largeur (resp. profondeur) de G est l'algorithme :
- Largeur : Entrée : $G = (S,A)$, $u \in S$; Sortie : liste des sommets accessibles depuis u par profondeur croissante
- Profondeur : Entrée : $G = (S,A)$, $s \in S$; Sortie : liste des sommets accessibles partant de s, en explorant toujours le plus profond visible

Exemple 13 : annexe

Implémentation (profondeur) :

```
PROF(S,A,u) ;
  l := liste vide
  soit PROF1(s) ; marque s ;
      dans w : s -> v et v non marqué
      l := [v, l] ; PROF1(v)
  PROF1(u).
```

Implémentation (largeur) :

```
LARG(S,A,u)
  pour tout v ≠ u faire couleur(v) := blanc ; fait.
  l := liste vide
  F := file vide
  couleur(u) := gris
  u enfile à F
  tant que F ≠ ∅
    s := tête F
    pour v : s -> v
      si couleur(v) = blanc
        couleur(v) := gris
        l := [v, l]
        enfile v F
    couleur(s) := noir
```

Applications : LARG permet de calculer facilement un plus court chemin en termes de nombre d'arêtes ; PROF les composantes fortement connexes du graphe, ou un tri topologique s'il est acyclique.

III) Arbres couvrants, optimisation

Les graphes modélisent des connexions / interactions, qui ont éventuellement un coût (graphes pondérés)

On veut simplement être sûr de pouvoir connecter les objets (transmission lents d'informations, réseaux lents de distribution, composants électroniques), ce qui revient en fait à des arbres couvrants minimaux

Définition 14 : si $G = (S, A)$ est connexe non orienté, un arbre couvrant de G est un sous-ensemble $A' \subset A$ tel que (S, A') est un arbre

Définition 15 : ~~si T CA le poids de T est w(~~
si (G, w) est un graphe connexe non orienté pondéré, le poids de $T \subset A$ est donné par $w(T) := \sum_{\{u,v\} \in T} w(\{u,v\})$

un arbre couvrant est minimal s'il est de poids minimal

les algorithmes de Kruskal et de Prim calculent un arbre couvrant minimal :

Kruskal $\begin{cases} S' = \emptyset \\ A' = \emptyset \end{cases}$ tant que $S' \neq S$, ajoute à A' une arête de poids minimal préservant l'acyclicité de (S', A'), et à S' les 2 sommets de cette arête

Prim $\begin{cases} S' = \{u\} \\ A' = \emptyset \end{cases}$ tant que $S' \neq S$, choisir $\{s, s'\} \in A$ tq $s \in S'$ et $s' \notin S'$ de poids minimal ; fais $(A', S') := (A' \cup \{s, s'\}, S' \cup \{s'\})$

Kruskal calcule à chaque étape une forêt de poids minimal qui, à la fin, est un arbre, et Prim calcule à chaque étape un ACM d'un ensemble de sommets qui est finalement S

En utilisant un tas pour simuler une file de priorité, les deux sont de complexité temporelle $O(|A| \ln(|S|))$

IV) Plus court chemin

Trouver le plus court chemin entre 2 sommets a d'innombrables applications (transport de marchandises, GPS, jeux vidéo, conduite automatisée ...)

On distingue 2 catégories de problème : calcul entre 2 sommets ou entre tous les couples

A) Plus court chemin entre 2 sommets

- Algorithme de Djikstra

Entrée : $u \in S$, $w : A \to \mathbb{R}^+$

Sortie : tableau d tq $d(v)$ = longueur d'un plus court chemin $u \to^* v$

DJIKSTRA(S, A, u, w) :
$E := \emptyset$; $F := S$ (file de priorité sur la distance d ~~...~~) ; pour $v \in S$ $d(v) := +\infty$ (père(v) := NIL) ; $d(u) := 0$
tant que $F \neq \emptyset$:
  u := extraire_min F
  E := E ∪ {u}
  pour v tq u → v fais relâcher u v w fait
fait

où RELACHER(u, v, w) :
  si $d(v) > d(u) + w(u,v)$
    $d(v) := d(u) + w(u,v)$
    père(v) := u

complexité $O(S^2)$, améliorable si G est peu dense

Inconvénient : les poids doivent être positifs

- Algorithme de Bellman-Ford : même spécification avec $w : A \to \mathbb{R}$ et G sans cycle de poids < 0

BELLMAN-FORD(S, A, u, w)
pour $v \in S$ $d(v) := +\infty$; père(v) := NIL fait ; $d(u) := 0$
itère |S| fois
  pour $s \to s' \in A$ relâcher (s, s', w) fait ;

Remarque : l'algorithme peut détecter un cycle de poids < 0

Complexité $O(SA)$, par listes d'adjacence ou matrices d'adjacence.

B) Plus court chemin entre tout couple de sommets

- Algorithme de Floyd-Warshal (DVPT)

Entrée : (G, w)

Sortie : matrice (d_{ij}) des plus courtes longueurs de chemin de i à j

principe : calculer $d_{ij}^{(k)}$ la longueur de chemin n'utilisant que les k premiers sommets de S

complexité $O(S^3)$

V) Problèmes de flots

But : maximiser la circulation d'un produit d'une source de "production" à un lieu de "consommation" (transport de marchandises, transmission d'informations, circulation de liquides ...)

Déf : un réseau de flot est un graphe pondéré $G=(S,A,c)$ $(s,t\} \subset S$; s est appelé source, t puits ; $c : A \to \mathbb{R}_+$ capacité $A \cap (S \times \{s\}) = A \cap (\{t\} \times S) = \emptyset$. On suppose que $(u \to v) \Rightarrow (v \not\to u)$ et on pose $\forall (u,v) \in S^2,\ (u,v) \notin A \Rightarrow c(u,v) = 0$

Un flot de G est une fonction $f : S^2 \to \mathbb{R}$ satisfaisant :

1- la contrainte de capacité $\forall (u,v) \in S^2 \quad 0 \leq f(u,v) \leq c(u,v)$

2- la conservation du flot $\forall u \in S \setminus \{s,t\}, \sum_{v \in S} f(v,u) = \sum_{v \in S} f(u,v)$

Définition 17 : la valeur $|f|$ de f est donnée par $\sum_{v \in S} f(s,v)$

f est un flot maximal s'il est de valeur maximale

Exemple 18 : $S = \{$Paris, Brest, Marseille, Lille$\}$, s = Brest, t = Paris ; nous représentons notés en bleu en annexe, f en rose est un flot ; f' en vert un flot maximal ; $|f| = 8$ et $|f'| = 11$

Définition 19 : si f est un flot de G, le réseau résiduel G_f est (S, A_f) où $c_f(u,v) = c(u,v) - f(u,v)$ si $c(u,v) > 0$; ~~$f(v,u)$~~

A_f pour tout $(u,v) \in A$ si $f(u,v) > 0$; $c_f(v,u) = f(u,v)$

Un chemin améliorant de f dans G est un chemin p de s à t de poids > 0 dans G_f

Proposition 20 : l'algorithme suivant

FORD FULKERSON (G, s, t)

f = flot nul

Tant qu'il existe p chemin améliorant de f

$f' = f + p$

retourner f

fonctionne si c est à valeurs entières, et retourne un flot maximal quand il termine.

La complexité (pour des capacités entières) est $O(A|f^*|)$, où f^* est maximal

On recherche le chemin améliorant par un parcours en largeur, en $O(SA^2)$: c'est l'algorithme d'Edmonds-Karp

Corollaire 21 : le problème du couplage maximal d'un graphe biparti est polynomial

Le problème de la couverture de développements d'intégration :

Entrée : l'ensemble de développements, L de leçons, $A \subset D \times L$ (de

Sortie : oui si pour tout $\ell \in L$, $\#(A \cap D \times \{\ell\}) \geq 2$

est polynômial (voir annexe)

VI Problèmes de graphes et NP-complétude

Définition 22 : $G = (S,A)$ est complet si $A = S^2 \setminus \Delta$

une clique de taille k est un sous-graphe complet à k sommets

Proposition 23 : les problèmes :

- ENS_INDEPENDANTS :
  Entrée : $G = (S,A)$ graphe non orienté, $k \in \mathbb{N}$ (*)
  Sortie : oui ssi il existe $S' \subset S$ tel que $A \cap S'^2 = \emptyset$, et $|S'| \geq k$
- CLIQUE :
  Entrée : (*)
  Sortie : oui ssi G a une k-clique
- COUVERTURE DE SOMMETS :
  Entrée : (*)
  Sortie : oui ssi ~~$\forall (u,v) \in A,\ u \in S'$~~ $\exists S' \subset S,\ |S'| \geq k$ et $A \subset (S' \times S) \cup (S \times S')$

sont NP-complets,

Proposition 24 : les problèmes :

- 3-COLOR
  Entrée : $G = (S,A)$ non orienté (**)
  Sortie : oui ssi G admet une 3-coloration
- CHEMIN_HAMILTONIEN
  Entrée : (**)
  Sortie : oui ssi G a un chemin hamiltonien ($u_1 \to u_2 \to \dots \to u_{|S|}$ u_i tq $S = \{u_1, \dots, u_{|S|}\}$)
- PVC (problème du voyageur de commerce)
  Entrée : (*), w poids positifs sur G
  Sortie : oui ssi G admet une tournée (chemin hamiltonien) de poids $\leq k$

sont NP-complets

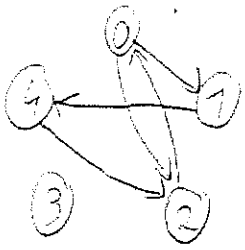
Proposition 25 : (DVPT)

- Il existe un algorithme polynômial de 2-approximation de PVC restreint aux graphes euclidiens
- Si PVC admet un algorithme de ρ-approximation (polynômial) avec $\rho > 1$, alors $P = NP$

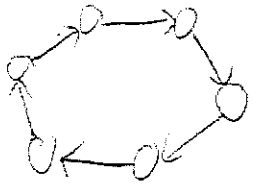
C.

Annexe

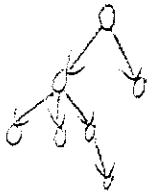
Exemple 2 :



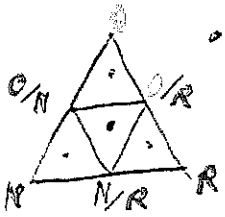
cycle

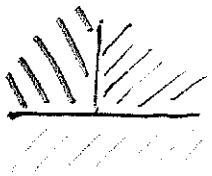


arbre



Théorème de Brouwer

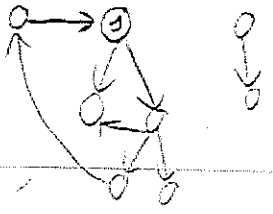


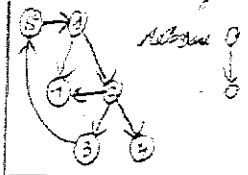


forte connexité

Exemple 13 :

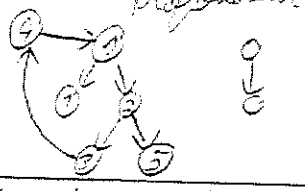
G =



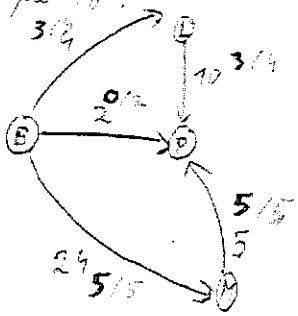


largeur :

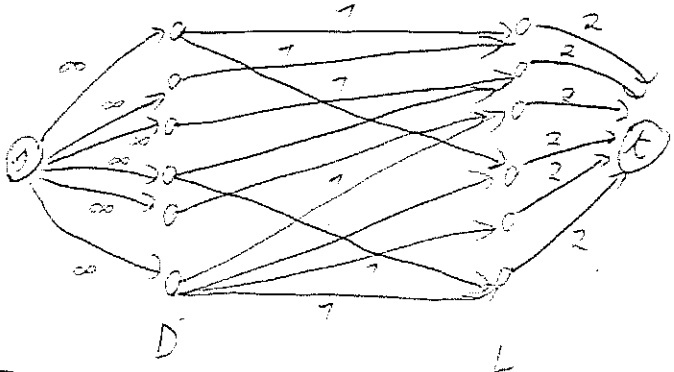
Profondeur



Exemple 14 :



Couverture de développement



REF :

- Gross / Johnson : "Computer and intractability, a guide to the theory of NP-completeness"
- Cormen