

## I Sous-graphe le plus dense

Soit  $G = (S, A)$  un graphe non orienté à  $n$  sommets et  $p$  arêtes. Pour  $S' \subseteq S$ , on définit la fonction de densité par :

$$\rho(S') = \frac{|A(S')|}{|S'|}$$

où  $A(S')$  est l'ensemble des arêtes de  $G$  ayant leurs deux extrémités dans  $S'$ .

1. Quelles sont les valeurs minimum et maximum de  $\rho(S')$ , en fonction de  $|S'|$  ?

On s'intéresse aux problèmes suivants :

### DENSE

Entrée : un graphe  $G = (S, A)$ .

Sortie : un ensemble  $S' \subseteq S$  tel que  $\rho(S')$  soit maximum.

### DENSE-DEC

Entrée : un graphe  $G = (S, A)$ , un entier  $k$  et un réel  $\alpha$ .

Sortie : existe-t-il un ensemble  $S' \subseteq S$  tel que  $|S'| = k$  et  $\rho(S') \geq \alpha$  ?

### CLIQUE-DEC

Entrée : un graphe  $G = (S, A)$  et un entier  $k$ .

Sortie : existe-t-il une clique de taille  $k$  ( $S' \subseteq S$  tel que  $|S'| = k$  et tous les sommets de  $S'$  sont adjacents) ?

2. En admettant que CLIQUE-DEC est NP-complet, montrer que DENSE-DEC est NP-complet.

On propose un algorithme glouton pour DENSE :

- Itérativement retirer un sommet de degré minimum (ainsi que toutes les arêtes adjacentes) jusqu'à ce qu'il n'y ait plus de sommet.
  - À chacune de ces itérations, calculer la valeur de  $\rho$  et conserver le maximum.
3. Expliquer comment on pourrait implémenter cet algorithme en complexité temporelle  $O(n + p)$ .

Soit  $S'^*$  tel que  $\rho(S'^*)$  soit maximum,  $v^* \in S'^*$  le premier sommet de  $S'^*$  retiré par l'algorithme glouton et  $S''$  l'ensemble des sommets restants juste avant de retirer  $v^*$ .

4. Montrer que  $\rho(S'') \geq \frac{\deg_{S''}(v^*)}{2}$ , où  $\deg_{S''}(v^*)$  est le degré de  $v^*$  dans  $S''$ .
5. Justifier que  $\rho(S'^*) \geq \rho(S'^* \setminus \{v^*\})$ .
6. En déduire que  $\deg_{S'^*}(v^*) \geq \rho(S'^*)$ .
7. En déduire que l'algorithme glouton est une 2-approximation pour DENSE.

## II Ensemble dominant

Soit  $G = (V, E)$  un graphe. Un ensemble dominant de  $G$  est un sous-ensemble  $D$  de  $V$  tel que tout sommet de  $V$  est soit dans  $D$ , soit adjacent à un sommet de  $D$ .

On note  $d(G)$  la taille d'un plus petit ensemble dominant de  $G$ .

1. Calculer  $d(G)$  si  $G$  est un chemin à  $n$  sommets.
2. On suppose que  $G$  est connexe et contient au moins 2 sommets. Montrer que  $d(G) \leq \left\lfloor \frac{n}{2} \right\rfloor$ .
3. On suppose que  $G$  ne contient pas de sommet isolé (sommet de degré 0). Montrer que  $d(G) \leq \left\lfloor \frac{n}{2} \right\rfloor$ .

Une couverture par sommets de  $G$  est un sous-ensemble  $C$  de  $V$  tel que toute arête de  $G$  ait au moins une extrémité dans  $C$ .

On s'intéresse aux problèmes suivants :

### DOMINANT

**Entrée :** Un graphe  $G = (V, E)$  et un entier  $k$ .

**Sortie :**  $G$  possède-t-il un ensemble dominant de taille  $k$  ?

### COUVERTURE

**Entrée :** Un graphe  $G = (V, E)$  et un entier  $k$ .

**Sortie :**  $G$  possède-t-il une couverture par sommets de taille  $k$  ?

4. Soit  $G$  un graphe sans sommet isolé. Est-ce qu'une couverture par sommets est un ensemble dominant ? Et réciproquement ?
5. On admet que COUVERTURE est NP-complet. Montrer que DOMINANT est NP-complet.
6. Décrire un algorithme efficace pour résoudre DOMINANT si  $G$  est un arbre. L'implémenter en OCaml.

## III k-Centres

### k-CENTRES

**Instance :** un entier  $k$  et un graphe complet pondéré et non orienté  $G = (S, A, d)$  dont les distances (poids) vérifient l'inégalité triangulaire ( $d(u, w) \leq d(u, v) + d(v, u)$ )

**Solution :** un sous-ensemble  $S \subseteq S$  de cardinal  $k$

**Optimisation :** minimiser  $r = \max_{v \in S} \min_{s \in S} d(v, s)$ .

Si l'on considère le cas particulier où les sommets du graphe sont des points du plan et le poids  $d(u, v)$  est la distance euclidienne entre  $u$  et  $v$ , on demande donc de recouvrir  $S$  par  $k$  cercles de même rayon  $r$  centrés en  $k$  points de  $S$ , en minimisant  $r$ .

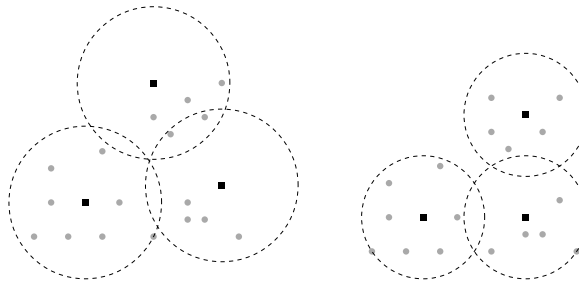


Figure 1: Une instance pour  $k = 3$ , avec deux solutions (la deuxième est meilleure).

On considère l'algorithme glouton suivant :

```
Choisir  $x$  un sommet quelconque
 $C \leftarrow \{x\}$ 
Pour  $i = 1$  à  $k - 1$  :
    Choisir  $y$  tel que  $\min_{c \in C} d(y, c)$  soit maximal.
     $C \leftarrow C \cup \{y\}$ 
return  $C$ 
```

On souhaite montrer que cet algorithme fournit une 2-approximation. On note  $C, r$  l'ensemble des centres et le rayon correspondant renvoyés par l'algorithme glouton et  $C^*, r^*$  une solution optimale.

1. Soit  $r$  la distance maximum d'un point à un centre en fin d'algorithme,  $u$  un point réalisant ce maximum et  $S = C \cup \{u\}$ . Montrer qu'il existe  $c_0 \in C^*$  et deux éléments distincts  $x$  et  $y$  de  $S$  tels que  $d(x, c_0) \leq r^*$  et  $d(y, c_0) \leq r^*$ .
2. Conclure.