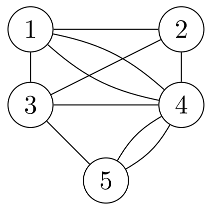
I Coupe minimum

Soit $G = (S, A)$ un multigraphe (G peut avoir plusieurs arêtes entre deux sommets), $n = |S|$ et $p = |A|$.
Une coupe de G est une partition $C = (S_1, S_2)$ de S telle que $S_1 \neq \emptyset$ et $S_2 \neq \emptyset$. Sa taille $|C|$ est le nombre d'arêtes entre S_1 et S_2 : $|C| = \{\{u, v\} \in A \mid u \in S_1 \text{ et } v \in S_2\}$. C est une coupe minimum si elle minimise $|C|$.

Soient $u, v \in S$. La contraction $G/\{u, v\}$ de $\{u, v\}$ dans G est le multigraphe obtenu à partir de G en fusionnant u et v en un nouveau sommet uv, en supprimant les arêtes entre u et v, et en remplaçant chaque arête $\{u, x\}$ ou $\{v, x\}$ par une arête $\{uv, x\}$.

1. Dessiner $G/\{3, 4\}$ si G est le multigraphe suivant :



2. On tire aléatoirement et uniformément une coupe C. Montrer que $\mathbb{P}(C \text{ est une coupe minimum}) \geq \frac{1}{2^n}$.

On propose l'algorithme suivant, où on note $S(G)$ l'ensemble des sommets et $A(G)$ l'ensemble des arêtes d'un graphe G :

> **Entrée :** Graphe connexe G
> **Sortie :** Coupe de G
> $H \leftarrow G$
> **Tant que** $|S(H)| > 2$ **:**
>   Choisir aléatoirement une arête $\{u, v\}$ de H
> $H \leftarrow H/\{u, v\}$
> **Renvoyer** $S(H)$

3. Montrer l'invariant : « H est connexe ».
4. On note $c(G)$ la taille minimum d'une coupe de G. Montrer l'invariant : « $c(H) \geq c(G)$ ». Peut-on avoir $c(H) > c(G)$?
5. Expliquer comment implémenter cet algorithme en $O(n^2)$.

Soit C une coupe minimum et $k = |C|$.

6. Montrer que $p \geq \frac{nk}{2}$.
7. Montrer que la probabilité de ne pas choisir une arête de C lors de la première contraction est au moins $1 - \frac{2}{n}$.
8. Montrer $\mathbb{P}(S(H) = C) \geq \frac{2}{n(n-1)}$.
9. En déduire un algorithme probabiliste pour trouver une coupe minimum de G avec une probabilité au moins $1 - \frac{1}{n}$.
   On pourra utiliser l'inégalité : $(1 - \frac{1}{x})^x \leq \frac{1}{e}$.
10. Montrer qu'un multi-graphe connexe à n sommets peut avoir au plus $\frac{n(n-1)}{2}$ coupes minimums. Donner un cas d'égalité.

II Ensemble indépendant

Soit $G = (S, A)$ un graphe non-orienté. Un ensemble indépendant (ou : stable) de G est un sous-ensemble $I \subseteq S$ tel que $\forall u, v \in I, \{u, v\} \notin A$. On note $\alpha(G)$ la taille du plus grand ensemble indépendant de G.

On considère les problèmes de décision suivants :

`INDEPENDANT`

Entrée : un graphe G et un entier k.
Sortie : est-ce que $\alpha(G) \geqslant k$?

`CLIQUE`

Entrée : un graphe G et un entier k.
Sortie : est-ce que G possède une clique de taille k, c'est-à-dire un sous-ensemble $C \subseteq S$ tel que $\forall u, v \in C, \{u, v\} \in A$?

1. On admet que `CLIQUE` est NP-complet. Montrer que `INDEPENDANT` est NP-complet.
2. Décrire un algorithme efficace pour calculer $\alpha(G)$ si G est un arbre.

On considère l'algorithme suivant pour `INDEPENDANT`, où $p \in [0, 1]$:

Algorithme 1

```
I ← ensemble obtenu en prenant chaque sommet de S avec probabilité p
Pour {u, v} ∈ A :
    Si u ∈ I et v ∈ I :
        Supprimer aléatoirement u ou v de I
Renvoyer I
```

3. Quelle est l'espérance de $|I|$ juste avant de rentrer dans la boucle Pour ?
4. Montrer que l'espérance de $|I|$ renvoyé par l'algorithme 1 est au moins $p|S| - p^2|A|$.
5. Montrer que $\alpha(G) \geqslant \dfrac{|S|^2}{4|A|}$.

On considère un autre algorithme pour `INDEPENDANT`, pour un graphe $G = (S, A)$:

Algorithme 2

```
S ← sommets de S ordonnés selon un ordre quelconque
I ← ∅
Pour v ∈ S :
    I ← I ∪ {v}
    Supprimer v et ses voisins de S
Renvoyer I
```

6. Montrer que l'algorithme 2 donne une $\dfrac{1}{\Delta + 1}$-approximation de $\alpha(G)$, où Δ est le degré maximum des sommets de G.
7. On ordonne les sommets de S suivant une permutation uniformément aléatoire.

   Calculer l'espérance de $|I|$ puis montrer que $\alpha(G) \geqslant \displaystyle\sum_{i=1}^{|S|} \frac{1}{d_i + 1}$, où d_i est le degré du i-ème sommet.