# Algorithmique et Complexité
# TD 5/7 – Théorie de la complexité

CentraleSupélec – Gif

ST2 – Gif

# Introduction

Ce TD est l'occasion de s'exercer sur le cours de la complexité et de pratiquer la réduction polynomiale.

# Plan

# Plan

# Présentation

Nous connaissons le problème de $k$-stable (*Independent Set*) vu en cours. Nous savons qu'il est NP-complet.

**Instance :**

- Un graphe non-orienté connexe $G = (V, E)$,
- $k \in \mathbb{N}^+$

**Question :** Existe-t-il un **stable** $S$ de $G$, c'est-à-dire un ensemble de sommets qui ne sont pas reliés entre eux par une arête, tel que $|S| \geq k$ ?

# Présentation

En revanche, nous ne connaissons pas encore le problème de $k$-couverture par sommets (*Vertex Cover*). Sa définition est la suivante :

**Instance :**

- Un graphe non-orienté connexe $G = (V, E)$,
- $k \in \mathbb{N}^+$.

**Question :** Existe-t-il un ensemble $C \subseteq V$ de taille $|C| \leq k$ tel que toute arête $\{u, v\} \in E$ a au moins une de ses extrémités dans $C$ $(u \in C \vee v \in C)$ ?

## Question 1

Démontrez que :

$S$ est un stable $\Longleftrightarrow C = V - S$ est une couverture par sommets.

## Question 1 : correction

$\Rightarrow$

Supposons que $S$ est un stable et $e = (u, v)$ une arête.

- Au maximum une extrémité de $(u, v)$ peut être dans $S$.
- Donc au moins une est dans $V - S$.
- Donc $C = V - S$ est une couverture par sommets.

# Question 1 : correction

$\Rightarrow$

Supposons que $S$ est un stable et $e = (u, v)$ une arête.
- Au maximum une extrémité de $(u, v)$ peut être dans $S$.
- Donc au moins une est dans $V - S$.
- Donc $C = V - S$ est une couverture par sommets.

$\Leftarrow$

Supposons que $C$ est une couverture par sommets et $u, v \in S = V - C$.
- Il ne peut pas y avoir d'arête $(u, v) \in E$ (sinon $C$ ne serait pas une couverture par sommets).
- Donc $S$ est un stable.

# Question 2

Démontrez que le problème de $k$-couverture par sommets est NP-complet.

# Question 2 : correction

**Étape 1.** Il faut montrer qu'il est dans NP : on peut vérifier en temps polynomial une solution à une instance positive du problème.

**Étape 2.** Il faut montrer qu'il est NP-difficile : il existe une **(2.1.)** réduction polynomiale **depuis** un problème NP-complet (générique) **vers** un problème de couverture par sommets (éventuellement particulier) telle que **(2.2.)** une instance positive de ce problème NP-complet correspond à une instance positive du problème de couverture par sommets, et **(2.3.)** réciproquement.

## Question 2 : correction

**Étape 1.** On peut écrire un algorithme en Python qui vérifie qu'un sous-ensemble $C$ est bien une solution à une instance $(G = (V, E), k)$ du problème de couverture par sommets :

```
def verifier(V,E,k,C):
```

## Question 2 : correction

**Étape 1.** On peut écrire un algorithme en Python qui vérifie qu'un sous-ensemble $C$ est bien une solution à une instance $(G = (V, E), k)$ du problème de couverture par sommets :

```
def verifier(V,E,k,C):

  if len(C) > k:
    return False
  for v in C:
    if not v in V:
      return False
  for (u, v) in E:
    if (u not in C) and (v not in C):
      return False
  return True
```

## Question 2 : correction

- La validation de *couverture* consiste à parcourir toutes les arêtes en vérifiant si au moins une de leurs extrémités est dans $C$.

-Cela se fait en $\mathcal{O}(|E| * |C| + |C|) \approx \mathcal{O}(|E| * |C|)$ si on utilise une liste pour $C$ et en $\mathcal{O}(|E| + |C|)$ si on utilise un *hashSet* pour $C$.

-Peu importe, du moment que la complexité est polynomiale en taille de l'instance.

# Question 2 : correction

**Étape 2.** Nous allons maintenant montrer que le problème de couverture par sommets est NP-difficile en utilisant une réduction polynomiale depuis le problème stable.

Soient $\mathcal{I}_S = \langle G = (V, E), k \rangle$ une instance quelconque du problème de $k$-Stable.

On construit de manière triviale une instance $\mathcal{I}_C = \langle G = (V, E), k_C \rangle$ du problème de couverture par sommets : le graphe ne change pas et $k_C = |V| - k$.

L'algorithme de construction d'une telle instance est clairement **polynomial** !

## Question 2 : correction

Il faut montrer que :
$\mathcal{I}_S$ est une instance positive $\Longleftrightarrow$ $\mathcal{I}_C$ est une instance positive

$\Longrightarrow$) Supposons que $\mathcal{I}_S$ est un instance positive de Stable, elle admet une solution $S \subseteq V$ avec $|S| \geq k$.

D'après la question précédente, $C = V - S$ est une couverture et on a $|C| = |V| - |S| \leq |V| - k = k_C$.

Donc $\mathcal{I}_C$ est une instance positive de couverture par sommets qui admet $C$ comme solution.

## Question 2 : correction

$\Longleftarrow$) Réciproquement, si $\mathcal{I}_C$ est une instance positive de couverture par sommets qui admet $C$ ($C \subseteq V \wedge |C| \leq k_C$) comme solution.

D'après la question précédente, $S = V - C$ est un stable et on a $|S| = |V| - |C| \geq |V| - k_C = k$.

Donc $\mathcal{I}_S$ est une instance positive de Stable qui admet $S$ comme solution.

## Question 2 : correction

**Conclusion** Il existe bien une réduction polynomiale de stable (qui est NP-complet et donc NP-difficile) vers couverture par sommets, donc couverture par sommets est NP-difficile. Or comme il est NP, il est NP-complet.

# Plan

# Présentation

On dit qu'un élément $e$ est couvert par un ensemble $U$ si $e$ appartient à $U$.

Étant donnés un ensemble fini $U$ et une famille $S = \{S_i, i \in I\} \subset \mathscr{P}(U)$, le problème consiste à couvrir tous les éléments $U$ avec une sous-famille de $S$.

Par exemple, considérons $U = \{0, 1, 2, 3, 4\}$ et
$S = \{\underbrace{\{0, 1\}}_{S_0}, \underbrace{\{2, 3\}}_{S_1}, \underbrace{\{3, 4\}}_{S_2}, \underbrace{\{0, 1, 2\}}_{S_3}\}$.

On peut couvrir $U$ avec $\{S_0, S_1, S_2\}$ mais la couverture qui utilise le moins de sous-ensembles est $\{S_2, S_3\}$.

# Question 1

Formalisez le problème de décision correspondant.

# Question 1 : correction

Données :

- un ensemble d'éléments $U$
- une famille $S \subset \mathscr{P}(U)$ de sous-ensembles de $U$.
- $k \in \mathbb{N}^+$, un nombre naturel positif

Question : Existe-t-il une sous-famille $S' \subseteq S$ telle que :

- $S'$ est une couverture de $U$,
  c.-à-d. : $U = \cup_{S_i \in S'} S_i$
- la sous-famille $S'$ est de taille inférieure ou égale à $k$,
  c.-à-d. : $card(S') \leq k$.

# Question 2

Montrer que le problème appartient à la classe NP.

# Question 2 : correction

On va écrire en Python une fonction qui renvoie vrai si et seulement si $S'$ est bien une couverture de $U$ de taille inférieure ou égale à $k$.

## Question 2 : correction

$U$ et $S$ seront représentés par des listes Python :

```
# U = {0, 1, 2, 3, 4}
U   = [0, 1, 2, 3, 4]
# S = { {0, 1}, {2, 3}, {3, 4}, {0, 1, 2}}
S   = [ [0, 1], [2, 3], [3, 4], [0, 1, 2]]
```

Une sous-famille $S' \subseteq S$ sera représentée par une liste $SS$ de $m = |S|$ booléens $[ss_0, \ldots, ss_{m-1}]$ telle que $ss_j = True$ si et seulement si $S[j] \in S'$.

```
# S' = { {0, 1}, {2, 3}, {3, 4}            }
# S  = [ [0, 1], [2, 3], [3, 4], [0, 1, 2]]
SS =   [ True,   True,   True,   False     ]
# S' = {                 {3, 4}, {0, 1, 2}}
# S  = [ [0, 1], [2, 3], [3, 4], [0, 1, 2]]
SS =   [ False,  False,  True,   True      ]
```

# Question 2 : correction

```
def SetCover_verif(U, S, SS, k):
```

# Question 2 : correction

```
def SetCover_verif(U, S, SS, k):

    # verification de la taille de la sous-famille
    if sum(SS) > k:
        return False

    # verification de la couverture
    for i in range(len(U)):
        for j in range(len(S)):
            if SS[j] and U[i] in S[j]:
                break
        else:
            return False
    return True
```

# Question 2 : correction

-Pour commencer, il faut déterminer la taille d'une instance du problème qui est ici la somme de taille de toutes les entrées, $|S| + |U|$.

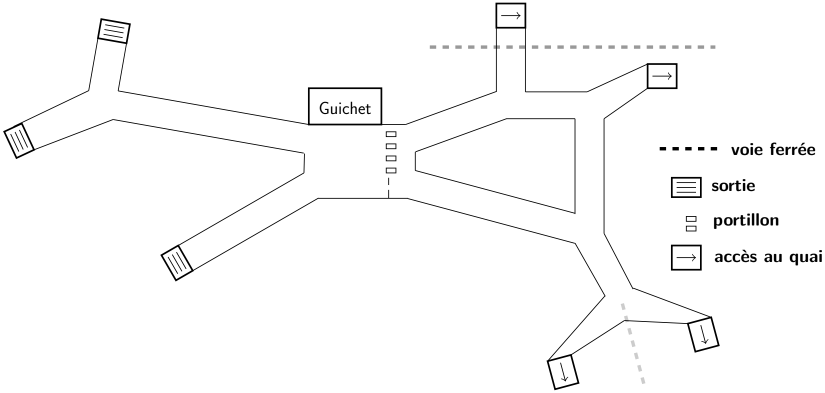
-On a une double boucle $|U| \times |S|$ avec à chaque fois une recherche en $\mathcal{O}(|U|)$ (`U[i] in S[j]`), on obtient une complexité polynomiale en $O(|U|^2|S|)$. Donc on peut vérifier une solution en temps polynomial : le problème est dans NP.

# Des caméras dans le métro

Dans une station de métro, des caméras 360° peuvent être installées à chaque intersection de couloir pour surveiller la circulation des usagers. Pour des raisons évidentes de coût et d'entretien de matériel, l'entreprise qui gère la station souhaite installer le moins de caméras mais elle veut que tous les couloirs soient surveillés par au moins une caméra.

# Des caméras dans le métro

Nous considérons le plan de la station ci-dessous :



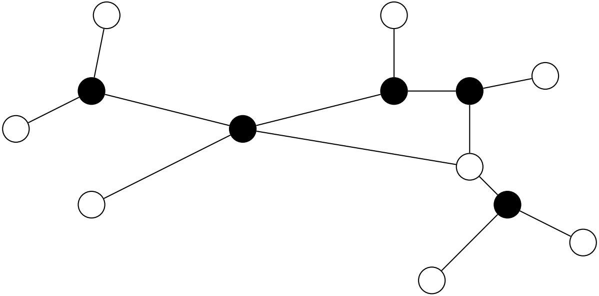
# Question 3

Modélisez la station de métro à l'aide d'un graphe dont les nœuds représentent les intersections, sorties ou accès aux quais et les arêtes représentent les couloirs.

Est-il possible de couvrir l'ensemble de la station avec 8 caméras ? Et avec seulement 5 caméras ?

## Question 3 : réponse

Avec 8, c'est l'embarras du choix ! Mais il n y a que deux solutions avec 5 caméras :



## Question 4

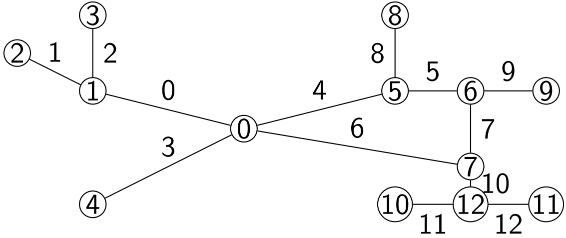
Il est naturel de modéliser le problème de pose de caméras dans la station de métro sous la forme d'une instance de Vertex Cover vu au premier exercice. On considère le graphe $G = (V, E)$ construit pour la station de métro et on se pose la question si avec $k$ sommets (où on positionnera les caméras) on peut couvrir toutes les arêtes (couloirs).

Modélisez maintenant le problème de pose de caméras dans la station de métro sous la forme d'une instance de Set Cover vu au deuxième exercice.

**Indication** : Numérotez tous les couloirs.

## Question 4 : correction

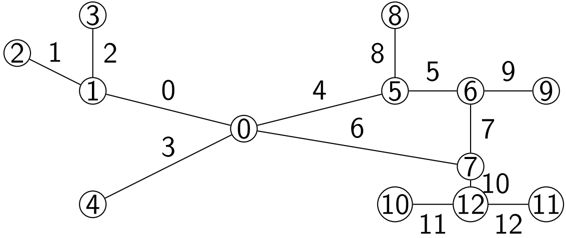
Les couloirs sont numérotés de la façon suivante :




On considère donc l'ensemble $U = \{0, \ldots, 12\}$ correspondant aux numéros de couloirs et la famille de sous ensembles :
$S_0 = \{0, 3, 4, 6\}$, $S_1 = \{0, 1, 2\}$, $S_2 = \{1\}$, ..., $S_6 = \{5, 7, 9\}$, ..., correspondant à chaque sommet (intersection, sortie ou quai).

## Question 4 : correction

Les couloirs sont numérotés de la façon suivante :




On considère donc l'ensemble $U = \{0, \ldots, 12\}$ correspondant aux numéros de couloirs et la famille de sous ensembles :
$S_0 = \{0, 3, 4, 6\}$, $S_1 = \{0, 1, 2\}$, $S_2 = \{1\}$, …, $S_6 = \{5, 7, 9\}$, …, correspondant à chaque sommet (intersection, sortie ou quai).

la question est donc : existe-t-il une sous-famille $S' \subseteq S$ telle que $|S'| \leq k$ et $S'$ est une couverture de $U$ ?

## Question 5

En généralisant le passage de Vertex Cover à Set Cover que vous avez réalisé pour cette instance, montrez que le problème de Set Cover est NP-difficile.

- Qu'en déduisez-vous ?

# Question 5 : correction

Pour montrer NP-difficile, il faut proposer une réduction polynomiale de Vertex Cover vers Set Cover.

Étant donné une instance de couverture par sommets $\mathcal{I}_{VC} = \langle G = (V, E), k \rangle$, nous allons construire une instance du problème Set Cover $\mathcal{I}_{SC} = (U, S = (S_i), k)$ telle que :

- $U = E$, les éléments sont les arêtes du graphe
- On numérote/confond les sommets $V$ de 1 à $n$ ($I = V = [1, n]$). Puis on définit $n$ sous-ensembles $S_i$ de $U$ tels que $S_i$ est l'ensemble des arêtes pour lesquelles le sommet $i$ est une extrémité, $S_i = \{(u, v) \in E | u = i \vee v = i\}$.

Cette transformation est clairement polynomiale en taille de l'instance $\mathcal{O}(|V| + |E|)$.

# Question 5 : correction

L'algorithme est le suivant :

```
def construire(V,E,k):
  U = E
  S = []
  for v in V:
    s = []
    for e in E:
      if (e[0]==v) or (e[1]==v):
        s.append(e)
    S.append(s)
  return (U,S,k)
```

Cet algorithme naïf est en $\mathcal{O}(|V| \times |E|)$ : il est polynomial.

## Question 5 : correction

Il faut ensuite montrer que :

$\Rightarrow$ Si $V' \subseteq V$ est une solution de couverture par sommets de taille au plus $k$ alors $C = \{S_i, i \in V'\}$ est une famille de taille inférieure ou égale à $k$.

- Et il s'agit bien d'une couverture par ensembles, car tout élément $e = (u, v) \in U = E$, étant une arête de $G$ est forcément couverte par au moins un sommet, soit $u$, soit $v$, soit les deux, par conséquent un/des sommet(s) couvrant(s) est/sont dans $V'$.

- Supposons que $u$ soit un sommet couvrant, on obtient par construction que $e \in S_u \in C$.

## Question 5 : correction

$\Leftarrow$ Si $C = \{S_i, i \in I'\}$ est une couverture par ensembles de taille inférieure ou égale à $k$, l'ensemble des sommets $V' = I'$ dont les numéros indexent $S_i$ de $C$ est de taille au plus $k$.

- L'ensemble $V'$ est une couverture par sommets dans le graphe : si il existe une arête $(u, v) = e \in E = U$, on sait par construction de $C$ que $S_u \in C \vee S_v \in C$ car $e$ est couverte par $C$.

- $e$ est donc couverte dans le graphe soit par $u \in V'$, soit par $v \in V'$, soit par ces deux sommets.

# Question 5 : correction

Conclusion : Set Cover est NP et aussi NP-difficile, donc il est NP-complet.