

## CC2 – Complexité


**45 minutes, documents non-autorisés.** Le barème est donné à titre indicatif.

### Exercice 1.

Questions de cours (4 points)

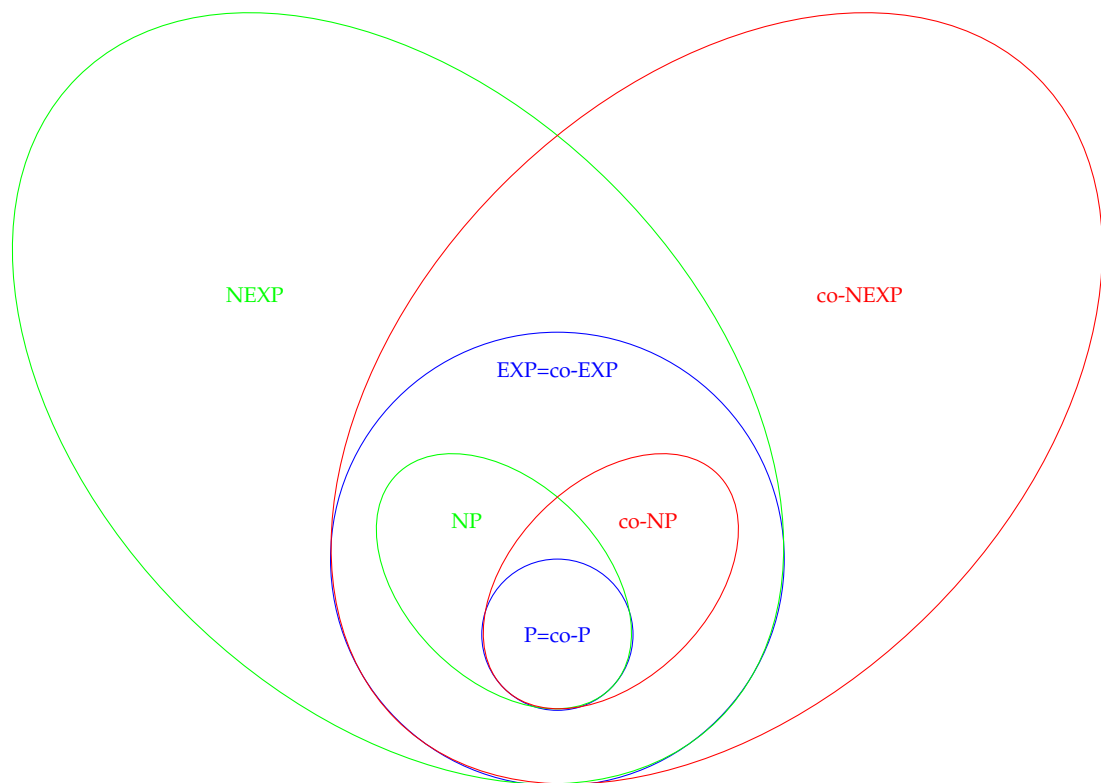
Dans cet exercice, ne justifiez pas vos réponses.

#### 1. De quoi P et NP sont les initiales ?

 *Polynomial* (temps polynomial) et *Non deterministic Polynomial* (pour temps polynomial non déterministe). L'erreur à ne pas faire est de répondre non polynomial car justement, on ne sait pas si tous les problèmes dans NP peuvent être résolus en temps polynomial ou non.


#### 2. Faites un diagramme de Venn des classes NP, EXP, NEXP, P, coNP, coEXP, coNEXP, coP






#### 3. Parmi les affirmations suivantes, lesquelles sont correctes ?

- (a) Un problème dans P est décidable.      (b) Un problème dans P est semi-décidable.  
 (c) Un problème dans NP est décidable.      (d) Un problème dans NP est semi-décidable.

 Elles sont toutes correctes. Tous les problèmes dans les classes de complexités P, NP, EXP sont décidables, toute la question est en combien de temps.

#### 4. Soit A et B deux problèmes de décision. Que signifie $A \leq_m^P B$ ?

 Il existe une fonction f telle que :

- $f : \Sigma_A^* \rightarrow \Sigma_B^*$  (des instances de A vers les instances de B);

- $f$  est calculable en temps polynomial;
- $\forall w \in \Sigma_A^*, w \in A \Leftrightarrow f(w) \in B$ .

## Exercice 2.

P, NP, coNP ou EXP (8 points)

Pour chacun des problèmes suivants, donner sa classe de complexité  $C$  (la plus petite possible) parmi P, NP, coNP ou EXP et prouver l'appartenance à  $C$ . On ne vous demande pas de prouver la  $C$ -difficulté de votre problème ou de prouver la non-appartenance à une classe de difficulté plus petite que  $C$ .

1. Soit  $G = (V, A)$  un graphe orienté et soit  $S \subseteq V$  un ensemble de sommets. Le **graphe induit**  $G[S] = (S, A')$  est un sous-graphe de  $G$  qui est formé en prenant  $S$  comme ensemble de sommets et en incluant toutes les arêtes de  $G$  qui relient des sommets dans  $S$ . Autrement dit  $A' = \{(u, v) \in A \mid u \in S \text{ et } v \in S\}$ .

### Problème FORTE\_CYCLICITÉ

*entrée* : Un graphe orienté  $G = (V, A)$  et un entier  $k \leq |V|$  codé en binaire.

*question* : Pour tout ensemble  $S \subseteq V$  avec  $|S| = k$ ,  $G[S]$  contient un cycle ?

☞ Quand  $(G, k)$  est une instance négative, on peut le prouver avec un certificat vérifiable en temps polynomial (en  $k$  et donc en  $|V|$ ) et le problème est donc dans coNP. Le certificat polynomial est un ensemble  $S \subseteq V$  tel que  $|S| = k$  mais tel que  $G[S]$  est acyclique. On peut représenter  $S$  comme une liste de sommets. Pour vérifier que ce certificat est valide :

- Vérifier que la liste est de taille  $k$  (temps linéaire en  $k$ )
- Vérifier que  $S$  ne contient pas deux fois le même sommet (temps quadratique en  $k$ )
- Vérifier que  $G[S]$  est acyclique. Plusieurs algorithmes peuvent faire ça. Le plus simple est de chercher pour chaque sommet  $u \in S$  un chemin de  $u$  vers  $u$  (avec un algorithme de recherche en largeur par exemple). Ceci prend un temps quadratique en  $k$ . Sinon il existe des algorithmes plus intelligents et linéaire (voir [Recherche de cycles dans les graphes](#) par par Adrien Panhaleux).

Notons que ce problème est (à une petite reformulation près) le complémentaire du problème Feedback vertex set qui est NP-complet.

2. Pour tout  $k \in \mathbb{N}$ , Une formule  $k$ -CNF est une formule CNF (pour *conjunctive normal form*) dont les clauses sont de taille au plus  $k$ .

Exemple de formule 3-CNF :  $(\lambda_1 \vee \bar{\lambda}_2 \vee \bar{\lambda}_3) \wedge (\bar{\lambda}_1 \vee \lambda_3 \vee \lambda_4) \wedge (\bar{\lambda}_2 \vee \lambda_3)$ .

### Problème 1-SAT :

*entrée* : Une formule 1-CNF  $\phi$ .

*question* :  $\phi$  est-elle satisfiable ?

☞ Pour toute variable  $\lambda_i$ , on vérifie qu'elle n'apparaît pas positivement et négativement dans  $\phi$  (c'est-à-dire, on n'a pas une clause  $\lambda_i$  et une autre clause  $\bar{\lambda}_i$ ). Ça prend un temps polynomial déterministe  $O(nm)$  où  $n$  est le nombre de variables et  $m$  le nombre de clauses. Le problème est donc dans P. Notons que cet algorithme fonctionne car si une variable apparaît positivement et négativement dans  $\phi$  alors peu importe la valuation de la variable, la formule ne sera pas satisfaite. À l'inverse, si cette condition est respectée, on peut valuer à vrai toutes les variables qui apparaissent positivement dans  $\phi$  et à faux toutes les autres et la formule sera satisfaite.

3. Si  $A$  est un AFI (automate fini indéterministe) alors  $L(A)$  est le langage reconnu par  $A$ .

### Problème ÉQUIVALENCE\_AFI

*entrée* : Deux AFI  $A$  et  $B$ .

*question* : Est-ce que  $L(A) = L(B)$  ?

☞ C'est dans EXP car on peut :

- Déterminiser et mettre sous forme minimale  $A$  et  $B$ . (ça prend un temps exponentiel et les deux AFD obtenus sont potentiellement de taille exponentielle en le nombre d'états de  $A$  et  $B$ )

- Comparer les deux AFD obtenus pour vérifier que ce sont bien les mêmes à un renommage des états près. (On peut le faire de manière efficace avec un parcours préfixe. Ça prend un temps simplement exponentiel car linéaire en la taille des AFD).

On n'en a pas parlé pendant ce cours, mais ce problème est en fait PSPACE-complet. Ceci signifie qu'on pourrait faire un algorithme (pas celui-là) qui résout le problème en utilisant seulement un espace polynomial.

#### Problème SOMME\_DE\_SOUS\_ENSEMBLE

4. *entrée* : Un ensemble d'entiers  $E \subseteq \mathbb{N}$  et un entier  $k$   
*question* : Est-ce qu'il existe  $S \subseteq E$  dont la somme des entiers est égale à  $k$  ?

On peut supposer que tous les entiers sont encodés en binaire et que  $E$  est donné sous la forme d'une liste d'entiers.

☞ C'est dans NP car quand l'instance est positive,  $S$  tel que la somme des entiers de  $S$  est égale à  $k$  serait un certificat valide. Pour vérifier le certificat, il suffit de calculer la somme des entiers de  $S$  et de la comparer à  $k$ .

### Exercice 3.

Half-Clique (8 points)

Rappel : dans un graphe non-orienté  $G = (V, E)$ , une clique est un sous-ensemble  $V' \subseteq V$  tel que pour tous  $v, v' \in V'$  on a l'arrête  $\{v, v'\} \in E$ . Sa taille est  $|V'|$ .

#### HALF-CLIQUE

*entrée* : un graphe non-orienté  $G' = (V', E')$  avec un nombre pair de sommet  
*question* :  $G'$  contient-il une clique de taille  $|V'|/2$  ?

1. Dessiner deux instances à 6 sommets de HALF-CLIQUE. La première instance doit-être positive et la seconde négative.
2. Montrer que le problème HALF-CLIQUE appartient à la classe NP.

☞ Pour montrer que c'est un problème dans NP, il suffit de montrer que dans le cas où une instance est positive, on peut le prouver grâce à un certificat qui peut-être vérifié en temps polynomial.

Certificat : un ensemble de sommets de taille  $|V|/2$  qui constitue une clique.

Vérification : Pour chaque pair de sommets  $\{u, v\}$ , vérifier que  $\{u, v\} \in E$ . Ça prend un temps polynomial  $o(|V|^2)$ . Selon comment l'ensemble de sommets est encodé, il faut potentiellement :

- vérifier que chaque sommet n'est présent qu'une seule fois dans la structure de donnée (si la structure de donnée est une liste par exemple). Ça prend un temps polynomial  $o(|V|^2)$ .
- vérifier que le nombre de sommets donné est bien  $|V|/2$ . Ça prend un temps polynomial  $O(|V|)$ .

Une autre solution est de faire une réduction vers CLIQUE. La réduction est simplement  $f : G' = (V', E') \mapsto (G', |V'|/2)$ . Elle est clairement polynomiale.  $G'$  une instance positive de HALF-CLIQUE  $\Leftrightarrow G'$  a une clique de taille  $|V'|/2 \Leftrightarrow f(G') = (G', |V'|/2)$  est une instance positive de CLIQUE. Comme HALF-CLIQUE  $\leq_m^P$  CLIQUE et que CLIQUE est dans NP alors HALF-CLIQUE aussi.

#### CLIQUE

*entrée* : Un graphe non-orienté  $G$  et un entier  $k$  codé en binaire.  
*question* : Le graphe  $G$  contient-il une clique de taille  $k$  ?

Rappel : CLIQUE est NP-complet.

3. Montrer que HALF-CLIQUE est NP-complet.

☞ On va montrer que CLIQUE  $\leq_m^P$  HALF-CLIQUE. Comme CLIQUE est NP-difficile, cela suffira pour montrer que HALF-CLIQUE aussi. Et comme HALF-CLIQUE est dans NP, alors cela montrera qu'il est également NP-complet.

On définit la réduction entre CLIQUE et HALF-CLIQUE suivante  $f : (G, k) \mapsto G'$  où  $G' = (V', E')$  est un graphe de taille  $2|V|$  tel que :

- $G$  est un sous graphe induit de  $G'$  ;
- en plus,  $G'$  contient  $|V| - k$  sommets supplémentaires formant une clique et chacun voisin de tous les sommets de  $V$  (on appellera cet ensemble de sommets  $X$ ),
- en plus,  $G'$  contient  $k$  sommets supplémentaires, chacun sans aucun voisin (on appellera cet ensemble de sommets  $Y$ ).

$f$  est clairement calculable en temps polynomial.

Soit  $(G, k)$  une instance de CLIQUE.

Dans un premier sens,  $(G, k)$  instance positive de CLIQUE  $\Rightarrow G$  a une clique  $C$  de taille  $k \Rightarrow G'$  a une clique  $C' = C \cup X$  de taille  $|V| \Rightarrow f(G, k) = G'$  est une instance positive de HALF-CLIQUE.

Dans l'autre sens,  $f(G, k) = G'$  est une instance positive de HALF-CLIQUE  $\Rightarrow G'$  a une clique  $C'$  de taille  $|V|$  (notons que  $Y \cap C' = \emptyset \Rightarrow C = C' \setminus X$  est une clique de taille au moins  $|V| - |X| = |V| - (|V| - k) = k$  dans  $G'$  et dans  $G$ .  $\Rightarrow G$  a une clique de taille  $k$  (n'importe quel sous-ensemble de  $C$  de taille  $k$ )  $\Rightarrow G$  est une instance positive de CLIQUE.

Donc,  $(G, k)$  instance positive de CLIQUE  $\Leftrightarrow f(G, k)$  instance positive de HALF-CLIQUE. Ceci conclue la réduction.