université de BORDEAUX	**Année Universitaire 2024–2025, Master 1 CSI** Examen, Théorie de la complexité 16 décembre 2024, 14h30–17h30	Collège Sciences et Technologies

La rédaction doit être précise et concise. Le barème est indicatif.
Seules, les réponses correctement justifiées apporteront des points.

Exercice 1 – Applications du cours (3 points).

Répondez aux questions suivantes en quelques lignes, en justifiant clairement vos réponses.

1. Le problème suivant est-il décidable ou indécidable ? Justifier.
   Entrée : Le code d'une machine de Turing M prenant en entrée des entiers codés en base 2.
   Question : La machine M calcule-t-elle la fonction $n \mapsto n$ de $\mathbb{N}$ dans $\mathbb{N}$?
2. Le problème suivant est-il dans la classe de complexité **L** ?
   Entrée : Pas d'entrée.
   Question : Soient $u_1 = 100$, $u_2 = 0$, $u_3 = 1$ et $v_1 = 1$, $v_2 = 100$, $v_3 = 0$. Existe-t-il une suite d'indices $i_1, i_2, \ldots, i_k$ (où $k \geqslant 1$), tous entre 1 et 3, tels que $u_{i_1} u_{i_2} \cdots u_{i_k} = v_{i_1} v_{i_2} \cdots v_{i_k}$?
3. Dans cette question, on suppose **P** $\neq$ **NP**. On note $\oplus$ l'opérateur Booléen « ou exclusif ». Il se comporte comme l'addition dans $\mathbb{Z}/2\mathbb{Z}$, c'est-à-dire que $0 \oplus 0 = 1 \oplus 1 = 0$ et $0 \oplus 1 = 1 \oplus 0 = 1$. Une **XOR-formule** est une conjonction de formules de la forme $\ell_1 \oplus \ell_2 \oplus \ell_3$ où ℓ_1, ℓ_2, ℓ_3 sont des littéraux. Par exemple, la formule $(x_1 \oplus \neg x_2) \wedge (x_1 \oplus \neg x_2 \oplus x_3) \wedge (x_2)$ est une XOR-formule. Le problème XOR-SAT suivant est-il **NP**-complet ?
   Entrée : Une XOR-formule φ.
   Question : La formule φ est-elle satisfaisable ?

Exercice 2 – Complexité (7 points).

Pour chacun des problèmes ci-dessous, déterminez la plus petite classe de problèmes vue en cours qui le contient. Si vous répondez **NL**, **NP** ou **PSPACE**, montrez aussi qu'il est complet pour cette classe.

Remarques

- Les réductions sont simples en choisissant le bon problème de départ.
- Pour les réductions, vous pouvez utiliser tout problème des feuilles de cours-TD.
- Pour montrer qu'un problème est $\mathcal{C}$-complet, pensez à justifier qu'il est dans $\mathcal{C}$.

On rappelle qu'en logique propositionnelle, un littéral est une variable ou une négation de variable. Si $k \in \mathbb{N}$, une k-clause est une disjonction de k littéraux, et une formule k-CNF est une conjonction de k-clauses.

1. NAE-4-SAT
   Entrée : Une formule 4-CNF φ.
   Question : Existe-t-il une affectation des variables de φ telle que dans chaque clause de φ, il y a un littéral vrai et un littéral faux ?
2. Appartenance pour automate déterministe
   Entrée : Un automate **déterministe** $\mathcal{A}$ sur un alphabet Σ, et un mot $w \in \Sigma^*$.
   Question : Le mot w est-il accepté par $\mathcal{A}$?
3. Appartenance pour automate non-déterministe
   Entrée : Un automate **non déterministe** $\mathcal{A}$ sur un alphabet Σ, et un mot $w \in \Sigma^*$.
   Question : Le mot w est-il accepté par $\mathcal{A}$?

4. SOMME APPROCHÉE

Entrée : Des entiers strictement positifs $x_1, \ldots, x_n$ et un entier S.

Question : Existe-t-il un sous-ensemble I de $\{1, \ldots, n\}$ tel que $\left|\sum_{i \in I} x_i - S\right| \leqslant 42$?

On rappelle qu'un graphe orienté est donné par un ensemble S de sommets et un ensemble $A \subseteq S \times S$ d'arcs entre sommets. Un arc (s, t) est noté par une flèche de s à t : $s \to t$. Un *chemin* dans un graphe orienté est une suite de sommets $(s_0, s_1, \ldots, s_{n-1}, s_n)$ telle que $s_i \to s_{i+1}$ est un arc pour tout $0 \leqslant i \leqslant n-1$. La longueur de ce chemin est n (c'est son nombre d'arcs). Enfin, ce chemin est *simple* si les sommets $s_0, s_1, \ldots, s_{n-1}, s_n$ sont distincts deux à deux.

5. PLUS LONG CHEMIN SIMPLE

Entrée : Un graphe orienté G, deux sommets s, t de G et un entier $k > 0$.

Question : Existe-t-il dans G un chemin simple de longueur **supérieure ou égale** à k, allant de s à t ?

6. PLUS COURT CHEMIN

Entrée : Un graphe orienté G, deux sommets s, t de G et un entier $k > 0$.

Question : Existe-t-il dans G un chemin de longueur **inférieure ou égale** à k, allant de s à t ?

Un graphe **non orienté** est dit *connexe* si pour tous sommets s, t du graphe, il existe un chemin de s à t. Un *arbre* est un graphe non orienté, connexe, et sans circuit. Le *degré* d'un graphe est le nombre maximal de voisins que peut avoir un sommet dans le graphe.

7. **ARBRE COUVRANT DE DEGRÉ BORNÉ**

Entrée : Un graphe non orienté G et un entier $k > 0$.

Question : Existe-t-il un sous-ensemble d'arêtes de G formant un arbre de degré k qui contient tous les sommets de G ?

Exercice 3 – Mot lointain (4 points). Soit A un alphabet et deux mots $u, v \in A^*$ de *même longueur* n. On définit la distance $d(u, v) \in \mathbb{N}$ comme suit. Soit $a_1, \ldots, a_n, b_1, \ldots, b_n \in A$ les lettres telles que $u = a_1 \cdots a_n$ et $v = b_1 \cdots b_n$. On définit $d(u, v)$ comme le nombre d'indices $i \leqslant n$ tels que $a_i \neq b_i$. Par exemple, si $A = \{0, 1, \square\}$:

$$\begin{aligned} d(000111,\ & 000110) &&= 1, \\ d(0\square 0,\ & 1\square\square) &&= 2, \\ d(011\square 0,\ & 110\square\square) &&= 3. \end{aligned}$$

On s'intéresse au problème MOT LOINTAIN suivant :

Entrée : Un alphabet A, des entiers $k, m, n \in \mathbb{N}$ et m mots $u_1, \ldots, u_m \in A^*$ de longueur n.

Question : Existe-t-il un mot $v \in A^*$ de longueur n tel que $d(u_i, v) \geqslant k$ **pour tout** $i \leqslant m$?

1. Montrer que MOT LOINTAIN est dans **NP**.

On veut maintenant montrer que le problème est **NP**-complet. On cherche à réduire 3-SAT. Soit donc φ une entrée de 3-SAT. On note $\{C_1, \ldots, C_m\}$ l'ensemble de clauses et $\{x_1, \ldots, x_n\}$ les variables de φ.

On utilise l'alphabet $A = \{0, 1, \square\}$ et on construit $m + 9$ mots $u_1, \ldots, u_{m+9}$, chacun de longueur $n + 2$:

- Pour $i \leqslant m$, le mot u_i de longueur $n + 2$ est obtenu à partir de la clause C_i : ses n premières lettres sont déterminées par C_i et les deux dernières sont égales à "$\square$". Plus précisément, on a $u_i = a_1 \cdots a_n \square\square$ tel que pour tout $j \leqslant n$,

$$a_j = \begin{cases} 0 & \text{si la variable } x_j \text{ apparaît positivement dans } C_i \\ 1 & \text{si la variable } x_j \text{ apparaît négativement dans } C_i \\ \square & \text{si la variable } x_j \text{ n'apparaît pas dans } C_i \end{cases}$$

  Par exemple si $C_i = x_2 \vee \neg x_5 \vee x_3$, alors,

$$u_i = \square 00\square 1\square\square\square \cdots \square.$$

- Les mots $u_{m+1}, \ldots, u_{m+9}$ sont les neuf mots de longueur $n + 2$ dont les n premières lettres sont égales à "$\square$" et les 2 dernières parcourent les 9 combinaisons de 2 lettres de $\{0, 1, \square\}$.

2. Montrer que $C_1 \wedge \cdots \wedge C_m$ est satisfaisable si et seulement si il existe un mot $v \in A^*$ de longueur $n+2$ tel que $d(v, u_i) \geqslant n$ pour tout $i \leqslant m+9$. Que peut-on en conclure pour MOT LOINTAIN ?

On s'intéresse maintenant au problème MOT PROCHE suivant :

Entrée : Un alphabet A, des entiers $k, m, n \in \mathbb{N}$ et m mots $u_1, \ldots, u_m \in A^*$ de longueur n.
Question : Existe-t-il un mot $v \in A^*$ de longueur n tel que $d(u_i, v) \leqslant k$ **pour tout** $i \leqslant m$?

3. En utilisant une réduction similaire à celle proposée pour MOT LOINTAIN, montrer que MOT PROCHE est également **NP**-complet.

Exercice 4 – Problèmes P-complets (8 points). En logique propositionnelle, on appelle *clause de Horn* une clause qui contient ***au plus un*** littéral positif. Par exemple, les trois clauses suivantes sont des clauses de Horn : (v), $(\neg v \vee \neg x)$ et $(\neg y \vee \neg z \vee \neg x \vee \neg u \vee v)$. À l'inverse, la clause suivante n'est **pas** une clause de Horn car elle contient deux littéraux positifs (soulignés) : $(\neg y \vee \underline{z} \vee \neg x \vee \neg u \vee \underline{v})$.

1. Montrer que le problème HORN-SAT suivant est dans **P** en décrivant en français un algorithme polynomial.

   Entrée : Une conjonction $\varphi = C_1 \wedge C_2 \wedge \cdots \wedge C_n$ de clauses de Horn.
   Question : La formule φ est-elle satisfaisable ?

   Vous pourrez d'abord considérer le cas où toute clause a au moins un littéral négatif. Sinon, votre algorithme pourra initialiser toutes les variables à « Faux », puis, modifier itérativement cette affectation selon la formule φ, jusqu'à obtenir une conclusion. Justifiez précisément la complexité de l'algorithme obtenu.

2. On dit que deux formules propositionnelles sont *équivalentes* si elles ont le même ensemble de variables, et sont vraies pour les mêmes affectations de ces variables. Soit $\varphi = (x \wedge y \wedge z) \Rightarrow (u \wedge v \wedge w)$. L'ensemble des variables de φ est $\{x, y, z, u, v, w\}$. Donner une conjonction de trois 4-clauses de Horn équivalente à φ.

3. On se place sur l'alphabet $\{0,1\}$. Soit $A \subseteq \{0,1\}^*$ un problème de la classe **P**. On suppose que $A \neq \emptyset$ et $A \neq \{0,1\}^*$. Quels sont les problèmes de **P** se réduisant polynomialement à A ?

On définit maintenant la notion de **P**-complétude. On dit qu'un problème A est **P**-*complet* si les deux conditions suivantes sont vérifiées :

(a) $A \in \mathbf{P}$,

(b) tout problème de **P** se réduit à A par une réduction en espace logarithmique.

3. Montrer que le problème HORN-SAT est **P**-complet.

   Remarques.

   - Pour le point (b), observez que nous n'avons encore rencontré aucun problème **P**-complet. Cela donne une indication sur ce qu'il faut faire dans cette question.
   - Vous pouvez utiliser la question 2 pour construire une clause de Horn.

4. Montrer que le problème reste **P**-complet si les clauses de la formule d'entrée contiennent au plus 3 littéraux.
   Indication. La réduction vue en cours de SAT à 3SAT ne produit pas une formule de Horn. À la place, vous pouvez utiliser l'approche suivante : pour toute 4-clause de Horn $C = \ell_1 \vee \ell_2 \vee \ell_3 \vee \ell_4$, ajouter une variable x_C et construire une conjonction φ_C de deux 3-clauses de Horn sur les variables de C et x_C, telle que toute affectation des variables de C qui la rend vraie se prolonge en une affectation qui rend φ_C vraie, et inversement, toute affectation des variables de φ_C qui la rend vraie a sa restriction aux variables de C qui rend C vraie.

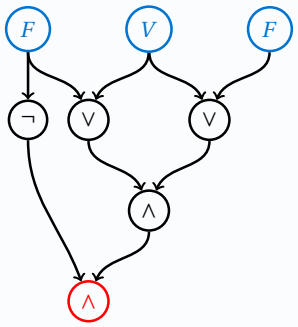
On passe maintenant à un second problème. Un ***circuit Booléen*** est un **graphe *orienté sans cycle*** dont les sommets sont appelés des *portes*. Les portes sont étiquetées soit par des valeurs Booléennes, Vrai (V) ou Faux (F), soit par des connecteurs logiques ($\wedge$ pour **Et**, $\vee$ pour **Ou** et $\neg$ pour **Non**). Un circuit doit satisfaire trois propriétés :

(a) Les portes qui n'ont pas d'arc entrant (appelées *portes d'entrée*) sont étiquetées par Vrai (V) ou Faux (F).

(b) Les autres portes sont étiquetées par des connecteurs logiques ($\wedge$, $\vee$ ou $\neg$). Une porte étiquetée par $\neg$ a un unique arc entrant, et une porte étiquetée par $\wedge$ ou $\vee$ a exactement 2 arcs entrants.

(c) Il y a une unique porte sans arc sortant, appelée la *porte résultat*. Les autres portes peuvent avoir un nombre arbitraire d'arcs sortants.

Un exemple de circuit Booléen est donné ci-dessous. Les portes d'entrée, en haut, sont indiquées en bleu. La porte résultat, en bas, est étiquetée $\wedge$ et indiquée en rouge.



Chaque porte calcule une valeur Booléenne. Sauf pour la porte résultat, cette valeur est ensuite transmise le long de tous ses arcs qui sortent de la porte. Comme les portes peuvent avoir plusieurs arcs sortants (sauf la porte résultat), une même sortie peut ainsi être utilisée plusieurs fois en entrée. Intuitivement, une porte effectue sur ses arcs entrants l'opération logique correspondant à son étiquette :

- Une porte d'entrée calcule Vrai si son étiquette est V et Faux si son étiquette est F.
- Une porte étiquetée par $\neg$, et dont l'unique arc entrant transmet la valeur val, calcule $\neg val$.
- Une porte étiquetée par $\vee$, et dont les deux arc entrants transmettent des valeurs val_1 et val_2, calcule $val_1 \vee val_2$.
- Une porte étiquetée par $\wedge$, et dont les deux arcs entrants transmettent des valeurs val_1 et val_2, calcule $val_1 \wedge val_2$.

La ***valeur calculée par le circuit*** est, par définition, celle de sa porte résultat. On considère le problème suivant :

ÉVALUATION DE CIRCUIT

Entrée : Un circuit Booléen C.
Question : Est-ce que le circuit calcule la valeur Booléenne Vrai ?

5. Montrer que ÉVALUATION DE CIRCUIT est **P**-complet. Pour montrer la **P**-difficulté, vous pouvez utiliser la question 4 même si vous ne l'avez pas faite.
6. On considère maintenant une restriction, appelée ÉVALUATION DE CIRCUIT MONOTONE :

   Entrée : Un circuit Booléen C **sans porte étiquetée** $\neg$.
   Question : Le circuit C calcule-t-il la valeur Vrai ?

   Montrer que ÉVALUATION DE CIRCUIT MONOTONE est encore **P**-complet. Vous pourrez commencer par le cas où toutes les négations portent sur des variables.
7. On considère une autre variation du problème ÉVALUATION DE CIRCUIT. Pour cela, on définit l'opérateur NAND de la façon suivante : si x, y sont deux variables Booléennes, $\mathsf{NAND}(x,y) = \neg(x \wedge y)$. On considère un nouveau type de porte : les portes NAND. Une telle porte a deux arcs entrants. Si les arcs entrants d'une porte NAND transmettent les valeurs val_1 et val_2, la porte calcule la valeur $\mathsf{NAND}(val_1, val_2)$.
   Un *NAND-circuit* est défini de la même façon qu'un circuit, mais ses portes qui ne sont pas des portes d'entrée sont toutes des portes NAND. Montrer que le problème suivant est encore **P**-complet.

   Entrée : Un NAND-circuit.
   Question : Le circuit C calcule-t-il la valeur Vrai ?
8. On considère maintenant un problème sur les grammaires hors-contexte dans une forme particulière (appelée forme normale de Chomsky). Dans une telle grammaire, les règles peuvent être de trois formes :
   (a) $X \to YZ$, où Y, Z sont des variables.
   (b) $X \to a$, où a est un terminal.
   (c) $S \to \varepsilon$, où S est la variable de départ de la grammaire.

   Montrer que le problème suivant est **P**-complet (n'oubliez pas de justifier pourquoi il est dans **P**) :

   NON-VACUITÉ POUR GRAMMAIRE HORS CONTEXTE

   Entrée : Une grammaire hors-contexte en forme normale de Chomsky.
   Question : La grammaire engendre-t-elle au moins un mot ?

   Indication. On peut partir du problème Horn-3SAT.