université de BORDEAUX	**Année Universitaire 2021/2022, Session 1 d'automne** Master CSI, UE 4TCY701U, Théorie de la complexité. Lundi 13 décembre 2021. 14h30–17h30. Durée : 3h. **Documents non autorisés**.	**Collège Sciences et Technologies**

Toute réponse non justifiée n'apportera pas de point.
Vous pouvez utiliser le résultat d'une question même si elle n'a pas été traitée.
Le barème est indicatif.

Exercice 1 : NP-complet, NL-complet ou L (7 points)

Pour chacun des problèmes ci-dessous, ***prouver*** qu'il est soit **NP**-complet, soit **NL**-complet, soit dans **L**, en justifiant.
Les réductions sont simples en choisissant le bon problème de départ (vous pouvez utiliser tout problème du cours, DM, DS).
Toutes les entrées des problèmes sont supposées finies, et les entiers supposés codés en binaire.

Rappels de définitions sur les graphes orientés

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). S'il existe un arc $s_n \to s_0$, ce chemin est aussi appelé un *circuit*. Enfin, ce chemin est *simple* si les sommets $s_0, s_1, \ldots, s_{n-1}, s_n$ sont distincts deux à deux.

1) 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** à k, allant de s à t ?

2) 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** à k, allant de s à t ?

3) Circuit simple de poids 42

Entrée : Un graphe orienté G et, associé à chaque arc de G un entier dans $\mathbb{Z}$ (appelé *poids* de l'arc).
Question : Existe-t-il dans G un circuit simple dont la somme des poids des arcs est 42 ?

Un graphe **non orienté** est *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 cycle. Enfin, une feuille d'un arbre est un sommet qui a un unique voisin.

4) Arbre couvrant à peu de feuilles

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 contenant tous les sommets de G, et ayant au maximum k feuilles ?

Un ***semigroupe*** est un ensemble S muni d'une multiplication associative $(x,y) \mapsto xy$: pour tous $x, y, z \in S$, on a $(xy)z = x(yz)$. On dit qu'un semigroupe S est *apériodique* si pour tout élément $s \in S$, il existe un entier $n \geqslant 1$ tel que $s^n = s^{n+1}$ (ici, s^n désigne la multiplication de n copies de s, c'est-à-dire $\underbrace{ss\cdots s}_{n \text{ fois}}$).

5) Semigroupe apériodique

Entrée : Un semigroupe S donné par sa table de multiplication.
Question : Le semigroupe S est-il apériodique ?

Si S est un semigroupe et X un sous-ensemble d'éléments de S, le *semigroupe engendré par* X est l'ensemble de tous les produits de la forme $x_1 x_2 \cdots x_n$ avec $n > 0$ et où chaque x_i est un élément de X (un même élément peut apparaître plusieurs fois dans ce produit).

6) Appartenance à un sous-semigroupe

Entrée : Un semigroupe S donné par sa table de multiplication, un sous-ensemble X de S et un élément x de S.
Question : L'élément x appartient-il au semigroupe engendré par X ?

7) Sat-42

Entrée : Une formule propositionnelle φ **en forme CNF**.
Question : Y a-t-il une affectation des variables pour laquelle chaque clause de φ a au moins 42 littéraux à « vrai » ?

8) 42Sat-42

Entrée : Une formule propositionnelle φ **en forme 42CNF** (chaque clause a donc exactement 42 littéraux).
Question : Y a-t-il une affectation des variables pour laquelle chaque clause de φ a au moins 42 littéraux à « vrai » ?

Exercice 2 : Problèmes P-complets (8 points)

Définition 1 (Rappel)

Dans l'exercice, on va montrer que plusieurs problèmes sont **P**-complets. Nous avons déjà vu *un* problème **P**-complet en cours : HORN-SAT (qui est une restriction de CNF-SAT) On en rappelle la définition ici. Une *clause de Horn* est une disjonction de littéraux qui contient *au plus* un littéral positif. Par exemple, "$x_0 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4$" est une clause de Horn. Une *formule de Horn* est une conjonction de clauses de Horn :

$$x_2 \wedge \neg x_0 \wedge x_4 \wedge (x_3 \vee \neg x_2) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (x_0 \vee \neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4).$$

Le problème HORN-SAT prend en entrée une formule de Horn et demande si celle-ci est satisfaisable. On a vu en cours que HORN-SAT est **P**-complet. Dans l'exercice, on pourra réutiliser (sans justification) le fait que HORN-SAT est **P**-difficile. Par contre, on demande dans la première question de reprouver que HORN-SAT est dans **P**.

On rappelle également qu'une fois qu'un problème à été montré **P**-complet dans une question, le résultat est réutilisable dans les autres questions (que vous ayez traité la question ou pas).

Problème A : Restriction de HORN-SAT à des 3-clauses.

On commence avec le problème HORN-3SAT : la restriction de HORN-SAT aux formules de Horn dont les clauses contiennent *au plus* trois littéraux.

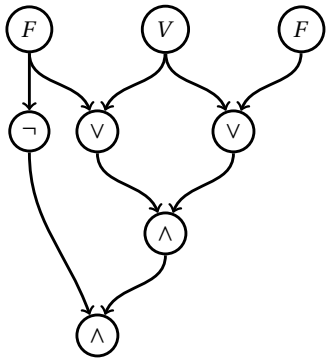
1) Donner un algorithme en temps polynomial qui résout HORN-3SAT. L'algorithme devra être direct. En particulier, il est interdit (pour cette question) de se servir du fait que HORN-SAT est **P**-complet.
2) Montrer que HORN-3SAT est **P**-complet. Dans cette question, vous pouvez utiliser la **P**-complétude de HORN-SAT.

Problème B : Circuits Booléens.

On passe maintenant à un second problème. Un ***circuit Booléen*** est un **graphe *orienté sans circuit*** 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 les propriétés suivantes :

- Les portes qui n'ont pas d'arc entrant (appelées *portes d'entrée*) sont étiquetées par une valeur Booléennes : Vrai (V) ou Faux (F).
- 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.
- 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. La porte résultat, en bas du dessin, est étiquetée $\wedge$.



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 celle de sa porte résultat. On considère le problème ÉVALUATION DE CIRCUIT suivant :

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

3) Montrer que ÉVALUATION DE CIRCUIT est **P**-complet.

4) 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.

Problème C : Appartenance au langage d'une grammaire.

On rappelle qu'une **grammaire hors-contexte** est un tuple $G = (\Sigma, V, S, R)$ où Σ est un alphabet, V est un ensemble fini de *variables*, $S \in V$ est la *variable de départ* et R est un ensemble de *règles*. Chaque règle s'écrit comme suit :

$$X \to \alpha \qquad \text{avec } X \in V \text{ et } \alpha \in (V \cup \Sigma)^*.$$

Le langage défini par la grammaire G, noté $L(G) \subseteq \Sigma^*$ contient tous les mots qui peuvent à partir de la variable de départ. On considère le problème APPARTENANCE suivant :

Entrée : Une grammaire hors-contexte $G = (\Sigma, V, S, R)$ et un mot $w \in \Sigma^*$.
Question : Est-il vrai que w appartient au langage $L(G)$?

5) Montrer que APPARTENANCE est **P**-complet.

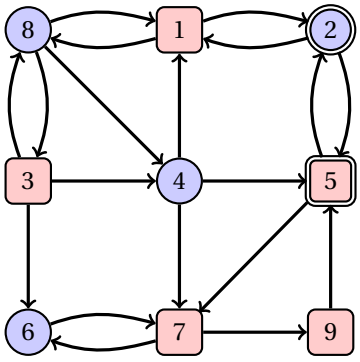
Problème D : Jeux d'accessibilité.

On s'intéresse maintenant à la résolution algorithmique des jeux d'accessibilité, et à leur complexité. Un tel jeu se joue à deux joueurs, nommés J_0 et J_1. Une partie se joue sur un graphe orienté $G = (S, A)$ (S est l'ensemble des sommets, A celui des arcs) ayant les propriétés suivantes :

— L'ensemble S est partitionné en deux sous-ensembles disjoints S_0 et S_1, c'est-à-dire que $S = S_0 \cup S_1$, avec $S_0 \cap S_1 = \emptyset$.

— Tout sommet de G a au moins un arc sortant.

Un exemple de tel graphe avec les sous-ensembles S_0, S_1 est donné sur la figure suivante. Les sommets de $S_0 = \{2,4,6,8\}$ sont dessinés dans des cercles en bleu, et ceux de $S_1 = \{1,3,5,7,9\}$ dans des carrés en rouge.



On se donne aussi un sous-ensemble de sommets But $\subseteq S$. Sur la figure, But $= \{2,5\}$ (sommets doublement entourés).

Une partie se déroule de la façon suivante. Un jeton est initialement placé sur un des sommets de G. Le jeu se déroule en tours. À chaque tour, si le jeton est sur un sommet s de S_0, alors J_0 joue, et sinon J_1 joue. Le joueur qui joue doit déplacer le jeton sur un sommet t tel que (s,t) est un arc. Dans l'exemple, si le jeton est sur le sommet 8, c'est J_0 qui joue (car $8 \in S_0$) et il doit déplacer le jeton sur 1, 3 ou 4 (car les arcs partant de 8 sont $(8,1)$, $(8,3)$ et $(8,4)$).

Le joueur J_0 gagne la partie si le jeton arrive, à n'importe quel moment, en un sommet appartenant à But. Si J_0 n'arrive pas à guider le jeton sur un sommet de But, alors J_1 gagne. Par exemple, à partir du sommet 1, J_0 a une stratégie gagnante : si J_1 déplace le jeton sur 2, J_0 gagne directement. Sinon, J_1 déplace le jeton sur 8, et J_0 le déplace ensuite sur 4, puis sur 5.

6) On considère les ensembles de sommets suivants (on rappelle que A est l'ensemble des arcs du graphe) :

$$X_0 = \text{But}, \qquad \text{et} \qquad X_{n+1} = X_n \cup \left\{ s \in S_0 \mid \text{il existe } t \text{ tel que } (s,t) \in A \text{ et } t \in X_n \right\} \cup \left\{ s \in S_1 \mid \text{pour tout } t \text{ tel que } (s,t) \in A, \text{ on a } t \in X_n \right\}.$$

Calculer les ensembles X_n pour $n = 0,1,2,3,4,5$ sur l'exemple précédent.

7) On note $X = \bigcup_{n\geqslant 0} X_n$. Montrer que X se calcule en temps polynomial (en fonction de la taille de G) et que J_0 a une stratégie pour gagner toute partie commençant sur un sommet de l'ensemble X.

8) Montrer que le joueur J_1 a une stratégie pour gagner toute partie commençant sur un sommet de l'ensemble $S \setminus X$.

9) On considère le problème JEU D'ACCESSIBILITÉ suivant :

Entrée : Un graphe G satisfaisant les conditions ci-dessus avec des sous-ensembles de sommets S_0, S_1, But, et un sommet s de G.

Question : Le joueur J_0 a-t-il une stratégie gagnante si le jeu débute avec le jeton sur s ?

Montrer que ce problème est dans **P**.

10) Montrer que le problème JEU D'ACCESSIBILITÉ est **P**-complet.

Exercice 3 : Automates enrichis (7 points)

Un *automate enrichi* manipule une mémoire. Il possède des transitions spéciales qui permettent de lire soit le début, soit la fin de la mémoire, et d'écrire au début de la mémoire. Formellement, un *automate enrichi* est un tuple $\mathscr{A} = (\Sigma, \Gamma, Q, I, F, \delta, \pi, \rho, \beta)$, où Σ est l'alphabet d'entrée, Γ est l'alphabet de mémoire, Q est un ensemble d'états, $I \subseteq Q$ est l'ensemble d'états initiaux et $F \subseteq Q$ est l'ensemble d'états initiaux finaux. De plus $\delta \subseteq Q \times \Sigma \times Q$ contient les *transitions de lecture*, $\pi \subseteq Q \times \Gamma \times Q$ est les *transitions de filage*, $\rho \subseteq Q \times \Gamma \times Q$ les *transitions de défilage* et $\beta \subseteq Q \times \Gamma \times Q$ les *transitions de dépilage*.

Il reste à définir le langage reconnu par un automate enrichi $\mathscr{A}$. Une *configuration* de $\mathscr{A}$ est un triplet $(q, u, x) \in Q \times \Sigma^* \times \Gamma^*$. Intuitivement, u est le mot restant à lire, et x est le contenu courant de la mémoire. On définit une relation "$\to$" entre les configurations :

— **Lecture :** Si $(q, a, r) \in \delta$ alors pour tout $u \in \Sigma^*$ et tout $x \in \Gamma^*$, on a $(q, a\,u, x) \to (r, u, x)$.

— **Filage :** Si $(q, d, r) \in \pi$ alors pour tout $u \in \Sigma^*$ et tout $x \in \Gamma^*$, on a $(q, u, x) \to (r, u, d\,x)$.

— **Défilage :** Si $(q, d, r) \in \rho$ alors pour tout $u \in \Sigma^*$ et tout $x \in \Gamma^*$, on a $(q, u, x\,d) \to (r, u, x)$.

— **Dépilage :** Si $(q, d, r) \in \beta$ alors pour tout $u \in \Sigma^*$ et tout $x \in \Gamma^*$, on a $(q, u, d\,x) \to (r, u, x)$.

Enfin, on note $\xrightarrow{*}$ la clôture reflexive transitive de $\to$. C'est-à-dire que $(q, u, x) \xrightarrow{*} (r, v, y)$ si et seulement si il existe $n \in \mathbb{N}$ et $(q_0, u_0, x_0), \ldots, (q_n, u_n, x_n) \in Q \times \Sigma^* \times \Gamma^*$ tels que,

$$(q, u, x) = (q_0, u_0, x_0) \to (q_1, u_1, x_1) \to \cdots \to (q_n, u_n, x_n) = (r, v, y).$$

Le langage reconnu par $\mathscr{A}$, noté $L(\mathscr{A}) \subseteq \Sigma^*$ contient exactement les mots $w \in \Sigma^*$ tels qu'il existe $q \in I$ et $r \in F$ qui satisfont $(q, w, \varepsilon) \xrightarrow{*} (r, \varepsilon, \varepsilon)$.

1) Soit $\Sigma = \{a, b, c\}$. Donnez un automate enrichi $\mathscr{A}$ tel que $L(\mathscr{A}) = \{a^n b^n c^n \mid n \in \mathbb{N}^*\}$. Il est recommandé de décrire cet automate à haut niveau. Une liste de transitions sans explication ne sera pas évaluée. À l'inverse, une description en français, si elle est à la fois concise, précise et correcte, suffira pour obtenir les points de cette question.

2) Un *automate à file* est un automate enrichi $\mathscr{F} = (\Sigma, \Gamma, Q, I, F, \delta, \pi, \rho, \beta)$ tel que $\beta = \emptyset$ (il n'y a *pas* de transitions de dépilage). Montrer que pour tout automate enrichi $\mathscr{A}$, il existe un automate à file $\mathscr{F}$ tel que $L(\mathscr{A}) = L(\mathscr{F})$.

3) Soit Σ un alphabet quelconque et $\mathscr{M}$ une machine de Turing déterministe ayant Σ pour alphabet d'entrée. Montrez qu'il existe un automate enrichi $\mathscr{A}$ tel que $L(\mathscr{A}) = L(\mathscr{M})$.

4) Un *automate à pile* est un automate enrichi $\mathscr{P} = (\Sigma, \Gamma, Q, I, F, \delta, \pi, \rho, \beta)$ tel que $\rho = \emptyset$ (il n'y a *pas* de transitions de défilage). Montrer que pour tout *automate à pile* $\mathscr{P}$, il existe une *grammaire hors-contexte* $\mathscr{G}$ telle que $L(\mathscr{G}) = L(\mathscr{P})$.

Indication

Pour toute paire d'états $(q, r) \in Q^2$ de $\mathscr{P}$, $\mathscr{G}$ contient une variable $X_{q,r}$, telle qu'un mot $w \in \Sigma^*$ peut être dérivé depuis celle-ci si et seulement si $(q, w, \varepsilon) \xrightarrow{*} (r, \varepsilon, \varepsilon)$.

5) Inversement, montrer que pour toute *grammaire hors-contexte* $\mathscr{G}$, il existe un *automate à pile* $\mathscr{P}$ tel que $L(\mathscr{G}) = L(\mathscr{P})$.

6) Pour chacune des affirmations suivantes, dites si elle est vraie ou fausse en justifiant brièvement la réponse :

a) Étant donnés deux automates à pile $\mathscr{P}_1$ et $\mathscr{P}_2$, le problème de savoir si $L(\mathscr{P}_1) \cap L(\mathscr{P}_2) = \emptyset$ est **NP**-complet.

b) Étant donné un automate enrichi $\mathscr{A}$, le problème de savoir si $L(\mathscr{A})$ est un langage régulier est indécidable. *Rappel :* un langage est régulier si il peux être défini par un automate "classique" comme ceux vus en cours.

c) Étant donné un automate à pile $\mathscr{P}$, le problème de savoir si $\varepsilon \in L(\mathscr{P})$ est dans **P** (où "ε" est le mot vide).

d) Étant donné un automate à file $\mathscr{F}$, le problème de savoir si $\varepsilon \in L(\mathscr{F})$ n'est *pas* dans **P** (où "ε" est le mot vide).