université de BORDEAUX	**Année Universitaire 2022–2023, Master CSI** DS, Théorie de la complexité (UE 4TCY701U) 16 novembre 2022, 9h30–12h30	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. Pour chacun des deux problèmes suivants, dire s'il est décidable et s'il est semi-décidable.

   (a) **Problème 1 : accessibilité d'un état.**
   Entrée : Une machine de Turing M (donnée par son code) et un état q de M.
   Question : La machine M a-t-elle un calcul sur le mot vide qui atteint l'état q ?

   (b) **Problème 2 : inclusion dans a^*.**
   Entrée : Une machine de Turing M sur l'alphabet d'entrée $\{a, b\}$.
   Question : La machine M accepte-t-elle uniquement des mots n'ayant que des a ?

2. Pour une machine de Turing M, on note $c(M)$ son code et on considère le langage suivant :

$$L = \{\, c(M) \mid M \text{ est une machine non déterministe qui } \textbf{n'accepte pas } c(M) \text{ en } 2^{|c(M)|} \text{ pas de calcul ou moins} \,\}.$$

   Le langage L est-il dans **NEXPTIME** ? Dans **NP** ? Que pouvez-vous en conclure ?

Exercice 2 : P vs. NP-complet (6 points)

Pour chacun de ces problèmes, démontrer soit qu'il est **NP**-complet, soit qu'il est dans **P**.

Remarques

- Les réductions sont *simples* en choisissant le bon problème de départ.
- Pour les réductions, vous pouvez utiliser *tout* problème **NP**-complet vu en cours ou TD.
- Pour montrer qu'un problème est **NP**-complet, pensez à vérifier qu'il est dans **NP**.

On rappelle qu'en logique propositionnelle, un littéral est une variable ou une négation de variable, une clause est une disjonction de littéraux, et une formule CNF est une conjonction de clauses. Dans les problèmes suivants, les entiers donnés en entrée sont codés en binaire.

1. **SAT**$_{\geqslant 42}$

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

2. **FULL SAT**

Entrée : Une formule propositionnelle φ en forme CNF.
Question : Existe-t-il une affectation des variables pour laquelle chaque clause de φ a tous ses littéraux à « vrai » ?

3. **SAT MODIFIÉ**

Entrée : Une formule propositionnelle φ en forme **DNF** (c'est-à-dire, une disjonction de conjonctions de littéraux).
Question : Existe-t-il une affectation des variables qui rend la formule φ vraie et une autre affectation des variables qui rend la formule φ fausse ?

Dans un graphe non orienté dont les sommets sont coloriés, on dit qu'une arête est **mal coloriée** lorsque les deux sommets à ses extrémités ont la même couleur.

4. **3-COLORATION À UNE ERREUR PRÈS**

Entrée : Un graphe fini non-orienté G.
Question : Existe-t-il une coloration des sommets avec trois couleurs telle qu'il existe au plus une arête mal coloriée ?

5. **COUVERTURE PAR SOUS-ENSEMBLES**

Entrée : Un ensemble fini E, des sous-ensembles $E_1, \ldots, E_n$ de E, et un entier k.
Question : Existe-t-il k sous-ensembles parmi $E_1, \ldots, E_n$ dont l'union est égale à E ?

Exercice 3 : Machines à piles et à file (12 points)

On s'intéresse à des machines utilisant une mémoire et acceptant des mots. On considère d'abord les *machines à deux piles*. Le mot d'entrée d'une telle machine est initialement écrit dans une partie de sa mémoire. Cette mémoire est constituée de deux mots, qu'elle peut modifier en fonction de ses transitions. Chacun de ces deux mots représente une pile : les lettres de chaque mot sont les symboles stockés dans la pile correspondante, le haut de pile étant la première lettre du mot.

Formellement, une machine à deux piles est un tuple $\mathscr{P} = (\Sigma, \Gamma, Q, I, F, \alpha_1, \beta_1, \alpha_2, \beta_2)$ où où Σ est l'alphabet d'entrée, Γ l'alphabet de la mémoire, Q un ensemble fini d'états, $I \subseteq Q$ l'ensemble d'états initiaux et $F \subseteq Q$ l'ensemble d'états finaux. De plus, $\alpha_1, \beta_1, \alpha_2, \beta_2$ sont des sous-ensembles de $Q \times \Gamma \times Q$. Intuitivement, α_1 et β_1 sont les transitions agissant sur la première pile, et α_2 et β_2 sont celles agissant sur la seconde. De plus, α_1, α_2 sont les transitions de dépilement, et β_1, β_2 celles d'empilement.

On définit à présent le langage accepté par une machine à deux piles $\mathscr{P}$. Une *configuration* de $\mathscr{P}$ est un triplet $(q, x_1, x_2) \in Q \times \Gamma^* \times \Gamma^*$. Intuitivement, x_1 est le contenu courant de la première pile et x_2 est celui de la seconde. On définit une relation "$\to$" entre les configurations :

Dépilement 1 : Si $(q, a, r) \in \alpha_1$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, a\,x_1, x_2) \to (r, x_1, x_2)$.

Dépilement 2 : Si $(q, a, r) \in \alpha_2$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, x_1, a\,x_2) \to (r, x_1, x_2)$.

Empilement 1 : Si $(q, a, r) \in \beta_1$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, x_1, x_2) \to (r, a\,x_1, x_2)$.

Empilement 2 : Si $(q, a, r) \in \beta_2$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, x_1, x_2) \to (r, x_1, a\,x_2)$.

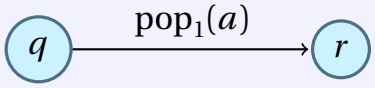
Enfin, on note $\xrightarrow{*}$ la relation définie par $(q,u,x)\xrightarrow{*}(r,v,y)$ si et seulement si il existe une suite de configurations deux à deux reliées par $\to$ allant de (q,u,x) à (r,v,y). Autrement dit, $(q,u,x)\xrightarrow{*}(r,v,y)$ lorsqu'il existe $n\in\mathbb{N}$ et $(q_0,u_0,x_0),\dots,(q_n,u_n,x_n)\in Q\times\Gamma^*\times\Gamma^*$ tels que,

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

Notez qu'on a toujours $(q,u,x)\xrightarrow{*}(q,u,x)$: c'est le cas $n=0$. Le langage accepté par $\mathscr{P}$, noté $L(\mathscr{P})$, est l'ensemble des mots $w\in\Sigma^*$ tels qu'il existe $q\in I$ et $r\in F$ qui satisfont $(q,w,\varepsilon)\xrightarrow{*}(r,\varepsilon,\varepsilon)$. En particulier, les deux piles doivent être vides à la fin de l'exécution.

I. Machines à deux piles sans restriction.

1. Construire une machine à deux piles (en donnant la liste de ses transitions) sur l'alphabet d'entrée $\Sigma=\{a,b\}$ acceptant le langage $\{a^nb^n \mid n\geqslant 0\}$, ainsi qu'une explication des phases de son exécution.

> **Remarque**
>
> Vous pouvez utiliser une notation graphique indiquant la nature de l'opération et la lettre concernée. Par exemple, vous pouvez indiquer que $(q,a,r)\in\alpha_1$ par le dessin :
>
> 
>
>
> et utiliser des notations similaires pour α_2,β_1 et β_2 (comme pop_2, push_1 et push_2).

2. Expliquer en français, sans donner les transitions, comment construire une machine à deux piles M sur l'alphabet d'entrée $\Sigma=\{a,b,c\}$ telle que $L(M)=\{a^nb^nc^n \mid n\geqslant 0\}$.
3. Montrer que toute machine à deux piles peut être simulée par une machine de Turing. Autrement dit, on demande d'expliquer comment, à partir du code source d'une machine à deux piles, on peut construire une machine de Turing acceptant le même langage.

> **Indication**
>
> Pensez à utiliser une machine de Turing à plusieurs bandes.

4. Montrer que toute machine de Turing à une bande peut être simulée par une machine à deux piles. Autrement dit, on demande d'expliquer comment, à partir du code source d'une machine de Turing, on construit une machine à deux piles acceptant le même langage.

II. Machines restreintes à deux piles.

On dit qu'une machine à deux piles $\mathscr{P}=(\Sigma,\Gamma,Q,I,F,\alpha_1,\beta_1,\alpha_2,\beta_2)$ est *restreinte* lorsque β_1 est vide : la machine n'empile jamais sur la première pile. Cette première pile ne sert donc qu'à lire le mot d'entrée (par des opérations « Dépilement 1 »).

5. Étant donnée une *grammaire hors-contexte* $\mathscr{G}$, décrire une *machine restreinte à deux piles* $\mathscr{P}$ telle que $L(\mathscr{G})=L(\mathscr{P})$. Vous pouvez utiliser les symboles de $\mathscr{G}$ dans la pile 2.
6. Inversement, étant donnée une *machine restreinte à deux piles* $\mathscr{P}$, décrire une *grammaire hors-contexte* $\mathscr{G}$ telle que $L(\mathscr{G})=L(\mathscr{P})$.

Indication

Pour toute paire d'états (q, r) de $\mathscr{P}$, la grammaire $\mathscr{G}$ contient une variable $X_{q,r}$, telle qu'un mot $w \in \Sigma^*$ peut être dérivé depuis celle-ci si et seulement si $(q, w, \varepsilon) \xrightarrow{*} (r, \varepsilon, \varepsilon)$. Vous pouvez distinguer deux cas, selon que le calcul revient de façon intermédiaire à une pile 2 vide, ou non.

III. Machines à file.

On définit maintenant une autre sorte de machine : les machines à file. Formellement, une machine à file est donnée par un tuple $\mathscr{F} = (\Sigma, \Gamma, Q, I, F, \alpha_1, \alpha_2, \gamma_2)$ où Σ, Γ, Q, I, F, α_1, α_2 sont définis comme pour les machines à deux piles, et $\gamma_2 \subseteq Q \times \Gamma \times Q$ représente les *transitions d'enfilement*. La machine peut donc lire (par dépilement) sur la mémoire 1 ou 2 (via α_1 ou α_2). Grâce à γ_2, elle peut enfiler sur le second mot de la mémoire. Formellement, une configuration de $\mathscr{F}$ est à nouveau un triplet $(q, x_1, x_2) \in Q \times \Gamma^* \times \Gamma^*$, et la relation "$\to$" entre configurations est maintenant définie ainsi (les dépilements sont exactement les mêmes que précédemment) :

Dépilement 1 : Si $(q, a, r) \in \alpha_1$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, a\,x_1, x_2) \to (r, x_1, x_2)$.

Dépilement 2 : Si $(q, a, r) \in \alpha_2$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, x_1, a\,x_2) \to (r, x_1, x_2)$.

Enfilement 2 : Si $(q, a, r) \in \gamma_2$, alors pour tous $x_1, x_2 \in \Gamma^*$, on a $(q, x_1, x_2) \to (r, x_1, x_2 a)$.

Notez que, comme les machines *restreintes* à pile, une machine à file ne peut pas ajouter de lettre sur le premier mot de la mémoire, x_1. Notez aussi que contrairement aux opérations d'empilement des machines à deux piles, la lettre insérée par enfilement sur le second mot de la mémoire, x_2, est ajoutée en **fin** de mot.

On définit $\xrightarrow{*}$ comme précédemment, et le langage accepté par $\mathscr{F}$, noté $L(\mathscr{F})$, est l'ensemble des mots $w \in \Sigma^*$ tels qu'il existe $q \in I$ et $r \in F$ qui satisfont $(q, w, \varepsilon) \xrightarrow{*} (r, \varepsilon, \varepsilon)$.

7. Expliquer à haut niveau le principe d'une machine à file acceptant l'ensemble des mots sur l'alphabet d'entrée $\Sigma = \{a, b, c\}$ dont les nombres de a, de b et de c sont égaux (sans donner explicitement la liste des transitions).
8. Montrer que toute machine à deux piles (même non restreinte) peut être simulée par une machine à file. Autrement dit, expliquer comment, à partir du code source d'une machine à deux piles, on peut construire une machine à file acceptant le même langage.

IV. Questions récapitulatives.

Pour chacune des affirmations suivantes, dites si elle est vraie ou fausse en justifiant brièvement la réponse. Vous pouvez utiliser les résultats des questions précédentes, même si vous ne les avez pas faites.

9. Tout langage accepté par une machine restreinte à deux piles est régulier.
10. Étant donnée une machine à deux piles $\mathscr{P}$, le problème de savoir si un mot d'entrée appartient à $L(\mathscr{P})$ est dans **NP**.
11. Étant données deux machines restreintes à deux piles $\mathscr{P}_1$ et $\mathscr{P}_2$, le problème de savoir si $L(\mathscr{P}_1) \cap L(\mathscr{P}_2) = \emptyset$ n'est pas semi-décidable.
12. Étant donnée une machine à file $\mathscr{F}$, le problème de savoir si $\mathscr{F}$ accepte le mot vide est dans **P**.