Master Sciences-Technologies-Santé / Mention Informatique
Info0809 : Informatique théorique / P. Mignot
Année universitaire 2012-2013 / Deuxième session

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Instructions :

- La durée de l'examen est de deux heures.
- Seul l'aide-mémoire est autorisé. Tout autre document est interdit.
- L'utilisation de propriétés autres que celles de l'aide-mémoire devront être démontrées.

Faculté des Sciences Reims

Questions indépendantes

1. Soit le langage $L \subset \{0,1\}^*$, qui accepte les chaines de longueur exactement égale à 4. Donner un ADF et une ER qui reconnaissent ce langage.
2. Est-il possible de donner un autre ADF reconnaissant exactement le même langage qu'à la question précédente ? Si oui, on le donnera. Si non expliquera pourquoi.
3. Donner un langage régulier L qui a p L-classes d'équivalence. La description devra clairement faire intervenir l'entier p, et devra être autre chose qu'une tautologie, et il devra être démontré pourquoi ce langage a p classes d'équivalence.
4. Donner un exemple d'un langage non régulier, et lui appliquer le lemme de l'étoile pour démontrer son irrégularité.
5. Donner un exemple d'une grammaire qui n'est pas libre du contexte, et lui appliquer le lemme de l'étoile pour démontrer son irrégularité.
6. Si on démontre qu'un langage **NP** est dans **P**, qu'aura-t-on montré ?
7. Y-a-t-il un rapport entre le fait qu'un langage soit en espace linéaire et la classe de complexité dans laquelle il se trouve ?
8. Donner la machine de Turing non déterministe qui décide `SAT`.
9. Donner l'arbre d'exécution de la machine de Turing non déterministe à temps polynomial qui décide si $(x \vee y \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee y)$ appartient à `SAT`.
10. Quelles classes A de complexité avons-nous vu en cours pour lesquelles $A = co\,A$?
11. Soit $A \in$ **NP**. Soit M la machine de Turing qui décide A. La connaissance de M aide-t-elle à déterminer si $\overline{A} \in$ **NP** ? Si oui, indiquer comment ; si non, indiquer pourquoi.
12. Donner un exemple d'un $w \in$ `HAMPATH`. Puis donner son certificat et le code de son vérificateur.
13. On considère un langage constitué d'un nombre infini de chaines incluses dans $\{0,1\}^*$. Existe-t-il toujours une machine de Turing qui reconnait ce langage ? On justifiera.
14. Montrer que **NL** est fermé par intersection.
15. Donner un exemple de langage non trivial dans **L**.
16. On veut construire un modèle représentant un protocole P, afin d'en étudier les propriétés (le langage reconnu étant l'ensemble des mots conformes ou engendrables par le protocole). On aimerait être en mesure de vérifier si deux protocoles sont équivalents (à savoir si deux protocoles différents reconnaissent le même langage). Parmi quelle classe de langages faut-il choisir ce protocole si l'on veut que ce problème soit décidable ?

Exercice 1 : Fermeture de $\mathcal{RE}$

1. Que peut-on dire de L si $L \in \mathcal{RE}$?
2. $\mathcal{RE}$ est-il fermé par union ? Si non, on donnera un contre-exemple ; si oui, on le démontrera.
3. $\mathcal{RE}$ est-il fermé par concaténation ? Si non, on donnera un contre-exemple ; si oui, on le démontrera.
4. $\mathcal{RE}$ est-il fermé par l'opérateur $*$? Si non, on donnera un contre-exemple ; si oui, on le démontrera.
5. $\mathcal{RE}$ est-il fermé par complémentation ? Si non, on donnera un contre-exemple ; si oui, on le démontrera.
6. Si la réponse aux trois premières questions est oui, quelle en est la conséquence vis-à-vis des langages réguliers ?
7. Si une propriété est vérifiée par $\mathcal{RE}$, peut-on en déduire quelque chose en informatique. On donnera un exemple.

Exercice 2 : `DECIDABLE` est-il décidable ?

On s'intéresse dans cet exercice à établir si le problème de déterminer si un problème est décidable est lui-même décidable. Autrement dit, soit `DECIDABLE` le langage contenant l'ensemble des descriptions des machines de Turing M qui reconnaissent des langages décidables, `DECIDABLE` est-il décidable ?

1. Soit L un langage décidable, et M la machine de Turing qui reconnait L. Que signifie pour M le fait que L soit décidable ?
2. En déduire l'expression de `DECIDABLE` sous la forme $\texttt{DECIDABLE} = \{\langle M\rangle \mid \forall w \ldots\}$.
3. Montrer, par diagonalisation, que `DECIDABLE` est indécidable.
4. Montrer, par réduction, que `DECIDABLE` est indécidable.
5. Quel théorème du cours permettrait d'arriver aussi à cette conclusion ? On appliquera ce théorème **rigoureusement**.
6. Quelle aurait été la conséquence pour l'informatique si `DECIDABLE` avait été décidable ?

Exercice 3 : Langages réguliers

1. Montrer que tout langage régulier est récursivement énumérable.
2. Montrer que tout langage régulier est décidable.
3. Soit $A_{\text{ADF}} = \{\langle M, w\rangle \mid M$ est un automate déterministe fini qui accepte $w\}$. Montrer que $A_{\text{ADF}} \in \mathbf{P}$.
4. Soit $A_{\text{ANF}} = \{\langle M, w\rangle \mid M$ est un automate non-déterministe fini qui accepte $w\}$. Pourquoi, alors qu'il existe toujours un automate déterministe fini pouvant reconnaître le même langage qu'un automate non-déterministe fini, avons-nous de bonne raison de supposer que $A_{\text{ANF}} \notin \mathbf{P}$.
5. Montrer que $A_{\text{ANF}} \in \mathbf{NP}$.
6. En terme informatique, quelle est la signification de $A_{\text{ADF}} \in \mathbf{P}$ et $A_{\text{ANF}} \in \mathbf{NP}$?
7. A la lumière des questions précédentes, comment expliquer une telle différence de complexité ?
8. Montrer que $A_{\text{ANF}} \in \mathsf{L}$.

Exercice 4 : Chemin dans un graphe

On considère dans cette exercice uniquement des graphes non orientés et non valués. Un chemin simple est un chemin sans répétition de sommet.
On rappelle que `UHAMPATH` est le langage contenant l'ensemble des chemins hamiltoniens pour les graphes non orientés, et que ce langage est **NP**-complet.

1. Soit $\texttt{SPATH} = \{\langle G, a, b, k\rangle \mid G$ contient un chemin simple de longueur **au plus** k de a à $b\}$.
   Montrer que $\texttt{SPATH} \in \mathbf{P}$.
2. Soit $\texttt{LPATH} = \{\langle G, a, b, k\rangle \mid G$ contient un chemin simple de longueur **au moins** k de a à $b\}$.
   Montrer que `LPATH` est **NP**-complet.

Exercice 5 : Problème de l'arrêt et NP-complétude

On rappelle que $\texttt{H}_{\texttt{TM}} = \{\langle M, w\rangle \mid M$ s'arrête sur $w.\}$

1. Quel théorème du cours permet de démontrer que $\texttt{H}_{\texttt{TM}}$ est indécidable ? On l'appliquera à $\texttt{H}_{\texttt{TM}}$.
2. Est-ce-que $\texttt{H}_{\texttt{TM}} \in \mathbf{NP}$?
3. Est-ce-que $\texttt{H}_{\texttt{TM}}$ est **NP**-difficile ? A savoir, est-ce-que $\forall A \in \mathbf{NP}$, $A \leq_P \texttt{H}_{\texttt{TM}}$?
4. En déduire si $\texttt{H}_{\texttt{TM}}$ est **NP**-complet.

Exercice 6 : Couverture

Soit $\texttt{SET-COVER} = \left\{\left\langle A_1, \ldots, A_p, k\right\rangle \mid \text{ il existe un ensemble } C \text{ de taille } k \text{ tel que } \forall 1 \leq i \leq n, A_i \cap C \neq \emptyset\right\}$.
Montrer que `SET-COVER` est **NP**-complet.
On pourra utiliser une réduction de `VERTEX-COVER` à `SET-COVER`. On rappelle que :

- la couverture d'un graphe est un sous-ensemble de nœuds tel que chaque arête possède au moins l'un de ces nœuds.
- $\texttt{VERTEX-COVER} = \{\langle G, k\rangle \mid G$ est un graphe non-orienté qui a une couverture de sommets à k noeuds$\}$.
- `VERTEX-COVER` est **NP**-complet (la démonstration a été faite en TD).