Master Sciences-Technologies-Santé / Mention Informatique
Info0809 : Informatique théorique / Pascal Mignot
Année universitaire 2013-2014
Session anticipée d'avril 2014 / Examen terminal

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Notes :
– Seul l'aide-mémoire est autorisé. Tout autre document est interdit.
– Toute propriété de l'aide mémoire utilisée devra être citée (même brièvement). Si une propriété utilisée n'est pas dans l'aide-mémoire, alors cette propriété devra être démontrée.

Questions indépendantes

1. Donner un exemple d'un langage non régulier, et lui appliquer le lemme de l'étoile pour démontrer son irrégularité.
2. 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.
3. Donner un exemple d'un langage régulier L qui a 3 L-classes d'équivalence.
4. Pourquoi existe-t-il des langages qui ne sont pas récursivement énumérables ?
5. En quoi le théorème de Rice démontre-t-il que la capacité des ordinateurs à résoudre certains problèmes est très limitée ? On donnera un exemple concret.
6. Donner un exemple d'un $w \in \texttt{SAT3}$ contenant au moins 3 clauses et 4 littéraux différents. Puis donner son certificat et le code de son vérificateur.
7. Donner l'arbre d'exécution de la machine de Turing non déterministe à temps polynomial qui décide le w donné à la question précédente.
8. Quelle est l'intérêt de savoir qu'il existe des langages **NP**-complet pour la classe **NP** ?
9. Y-a-t-il un rapport entre le fait qu'un langage soit décidable, et la classe de complexité spatiale dans laquelle il se trouve ?
10. Donner un exemple de langage non trivial dans **L**.

Exercice 1 : Primalité et complexité

1. Soit un entier n écrit sur la bande d'entrée d'une machine de Turing. Quel est la taille de cette entrée en supposant que l'on utilise en codage raisonnable de n ?
2. Donner la fonction totalement calculable permettant d'effectuer la multiplication deux entiers.
3. Quelle est la complexité temporelle de cette multiplication ?
4. On dit que deux nombres a et b sont premiers entre eux si aucun diviseur $d > 1$ de a ne divise b, et vice-versa. Soit $\texttt{RELPRIME} = \{\langle a, b\rangle \mid a \text{ et } b \text{ sont premiers entre eux}\}$. Montrer $\texttt{RELPRIME} \in \mathbf{P}$.
5. Donner la fonction totalement calculable permettant de calculer la factorielle d'un nombre n.
6. Quelle est la complexité temporelle de cette fonction ? Si besoin, on rappelle que $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ (Stirling).
7. Soit $\texttt{PRIME} = \{\langle n\rangle \mid n \text{ est un nombre premier}\}$. Donner une machine de Turing qui décide $\texttt{PRIME}$.
8. Quelle est la complexité temporelle de l'algorithme proposé à la question précédente ?
9. Il a été démontré en 2002 que $\texttt{PRIME} \in \mathbf{P}$. Si la complexité trouvée à la question précédente est différente, qu'est-ce-que cela prouve ?
10. Que peut-on en déduire sur la classe de complexité spatiale de $\texttt{PRIME}$?
11. Est-ce que la machine de Turing décidant $\texttt{PRIME}$ donnée à la question 7 est dans **NL** ?

Exercice 2 : Fermeture de NP par concaténation

1. Donner en terme de langages, de machines de Turing, et de vérificateur ce que signifie "démontrer que **NP** est fermé par concaténation" en utilisant des vérificateurs.
2. Effectuer la démonstration en utilisant des vérificateurs.
3. Donner en terme de langages, de machines de Turing (déterministe ou pas) ce que signifie "démontrer que **NP** est fermé par concaténation" en utilisant des machines de Turing non déterministe.
4. Le démontrer en utilisant des machines de Turing non déterministes.

Exercice 3 : Machine de Turing à mémoire bornée

On souhaite faire en sorte d'arrêter le fonctionnement d'une machine de Turing M (à une bande) si elle utilise plus que les p premières cases de sa bande.
On définit l'ensemble $\texttt{MEM}_n$ comme l'ensemble des machines de Turing qui s'exécutent sur une bande de taille de taille n au plus.

1. Montrer que $\texttt{MEM}_n$ est récursivement énumérable.
2. $\texttt{MEM}_n$ est-il décidable ?

On se place maintenant sur l'ensemble des machines présentes dans $\texttt{MEM}_n$.

3. Rappeler ce qu'est une configuration pour une machine de Turing.
4. Que peut-on dire d'une machine de Turing qui, lors de son exécution, passe par deux configurations identiques ?
5. Le nombre de configurations possibles pour une machine de Turing dans $\texttt{MEM}_n$ est-il fini, dénombrable ou infini ? S'il est fini, on donnera alors le nombre de configurations possibles. S'il est dénombrable, on le mettra en bijection avec $\mathbb{N}$. S'il est infini, on utilisera la méthode de diagonalisation.
6. Soit M, une machine de Turing quelconque de $\texttt{MEM}_n$. M s'arrête-t-elle toujours ?
7. Le problème de l'arrêt sur les machines de $\texttt{MEM}_n$ est-il décidable ? (*i.e.* soit M est machine quelconque de $\texttt{MEM}_n$, puis-je décider si M s'arrête ?).
8. Quelles conséquences sur le fonctionnement d'un ordinateur réel peut-on déduire des résultats des questions précédentes ?

Exercice 4 : 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 5 : Tautologie, satisfiabilité et NP-complétude

Une formule booléenne ϕ est une tautologie si elle est vraie quelque soit la valeur de vérité de ses littéraux.
Soit le langage TAUTOLOGY :

$$\texttt{TAUTOLOGY} = \{\langle\phi\rangle \text{ tel que } \phi \text{ est une tautologie }\}$$

1. Donner un exemple d'une formule booléenne $\phi \in \texttt{TAUTOLOGY}$.
2. Est-ce-que $\texttt{TAUTOLOGY} \in \textbf{NP}$?
3. Soit FALSIFIABLE, l'ensemble complémentaire de TAUTOLOGY. Décrire le langage FALSIFIABLE, et donner un exemple d'une formule booléenne $\phi \in \texttt{FALSIFIABLE}$.
4. Est-ce-que $\texttt{FALSIFIABLE} \in \textbf{NP}$?
5. Est-ce-que $\texttt{SAT} \leq_p \texttt{FALSIFIABLE}$ en temps polynomial ?
6. Que peut-on en déduire sur FALSIFIABLE ?
7. Que peut-on en déduire sur TAUTOLOGY ?

Exercice 6 : P-space complétude

Montrer que le langage INPLACE= $\{\langle M, w\rangle \mid M$ accepter w en restant dabs les $|w|$ premières cellules de la bande$\}$ est **P**-space complet. On pourra effectuer la démonstration en suivant les étapes suivantes :

1. Montrer que INPLACE est décidable (nécessaire car ceci n'est pas évident *a priori*).
2. Montrer que $\texttt{INPLACE} \in \textbf{PSPACE}$.
3. Montrer que INPLACE est **PSPACE**-difficile.
4. Conclure.