Master I d'informatique
INFO0809 : info. théorique
Pascal Mignot

Université de Reims Champagne-Ardenne

Examen terminal

session de juin 2011

Notes :
– 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.

Questions élémentaires

1. Expliquer pourquoi, en informatique, il est important à quel langage appartient la classe d'un problème que l'on essaie de résoudre. On donnera en particulier des exemples concrets pour les langages réguliers, libres de contexte, récursivement énumérable, et non récursivement énumérable.
2. Expliquer pourquoi, en informatique, il est important de savoir dans quelle classe de complexité un problème se trouve ? On donnera deux exemples concrets (un dans **P** et un dans **NP**), avec des exemples de temps d'exécution.
3. Est-ce-que l'écriture $O(2^n) = 3^{O(n)}$ a un sens ? On justifiera.
4. Donner un exemple d'un $w \in$ `SUBSET-SUM`. Puis donner son certificat et le code de son vérificateur.
5. Donner le code de la machine de Turing non déterministe à temps polynomial qui décide `SAT3`. Puis, donner un exemple d'un $w \in$ `SAT3`, ainsi que le résultat de son exécution sur la machine de Turing non déterministe.

Exercice 1 : classe d'un langage (1)

Soit le langage $L_1 = \{w \in (0|1)^* \mid$ il y a le même nombre de 01 et de 10$\}$.

1. Ce langage est-il régulier ? Si oui, donner son automate déterministe fini, si non, le prouver en utilisant le lemme de l'étoile pour les langages réguliers.
2. Ce langage est-il libre du contexte ? Si oui, donner sa grammaire libre du contexte, si non, le prouver en utilisant le lemme de l'étoile pour les grammaires libres du contexte.
3. Ce langage est-il récursivement énumérable ? Si oui, donner sa machine de Turing, si non, le démontrer.

Exercice 2 : classe d'un langage (2)

Soit le langage $L_2 = \{0^n1^n0^n1^n \mid n \geq 0\}$.

1. Ce langage est-il régulier ? Si oui, donner son automate déterministe fini, si non, le prouver en utilisant le lemme de l'étoile pour les langages réguliers.
2. Ce langage est-il libre du contexte ? Si oui, donner sa grammaire libre du contexte, si non, le prouver en utilisant le lemme de l'étoile pour les grammaires libres du contexte.
3. Ce langage est-il récursivement énumérable ? Si oui, donner sa machine de Turing, si non, le démontrer.

Exercice 3 : contrôle d'exécution

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.

1. Expliquer le principe pour assurer ce comportement.
2. Écrire la machine de Turing M' correspondante.
3. Est-ce-que la machine M' qui vérifie cette contrainte est décidable ? Si oui, on le démontrera. Si non, on donnera un contre-exemple.

Exercice 4 : langage non récursivement énumérable

Soit $(0|1)^*$ l'ensemble des mots binaires. Montrer qu'il existe des langages $L \subset (0|1)^*$ qui ne sont pas récursivement énumérables.

Exercice 5 : langage décidable

Soit le langage suivant : $E_{\text{GLC}} = \{\langle G \rangle \mid G \text{ est une grammaire libre de contexte et } L(G) = \emptyset\}$.

1. Comment vérifier que G est une grammaire libre de contexte ?
2. Comment montrer $L(G) = \emptyset$ si G est une grammaire libre de contexte ?
3. En déduire la machine de Turing correspondante. La décidabilité devra être clairement établie.

Exercice 6 : fermeture des ensembles récursivement énumérables

Démontrer que les langages récursivement énumérables sont fermés :

1. par concaténation.
2. par l'opérateur étoile.

Exercice 7 : 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 8 : complexité temporelle

On rappelle qu'un chemin simple dans un graphe non orienté est un chemin sans répétition de sommet.

1. Soit $\texttt{SPATH} = \{\langle G, a, b, k \rangle \mid G$ contient un chemin simple de a à b de longueur **au plus** $k\}$.
   Montrer que $\texttt{SPATH} \in \mathbf{P}$.
2. Soit $\texttt{HAMPATH} = \{\langle G, a, b \rangle \mid$ il existe un chemin Hamiltonien dans le graphe **orienté** G entre a et $b\}$
   et $\texttt{UHAMPATH} = \{\langle G, a, b \rangle \mid$ il existe un chemin Hamiltonien dans le graphe **non orienté** G entre a et $b\}$.
   On veut montrer que UHAMPATH est **NP**-complet, en utilisant le fait que HAMPATH est **NP**-complet.
   (a) Montrer que $\texttt{UHAMPATH} \in \mathbf{NP}$.
   (b) Proposer une fonction de transformation f qui transforme un graphe orienté $< G, a, b > \in \texttt{HAMPATH}$ en un graphe non orienté $< G', a, b > \in \texttt{UHAMPATH}$ (indice : ajouter des sommets supplémentaires de façon à forcer le parcours des sommets à avoir lieu dans un sens particulier), et vérifiant en plus les questions (c) à (e) suivante.
   (c) Montrer qu'avec cette transformation f, $w \in \texttt{HAMPATH} \Rightarrow f(w) \in \texttt{UHAMPATH}$.
   (d) Montrer qu'avec cette transformation f, $f(w) \in \texttt{UHAMPATH} \Rightarrow w \in \texttt{HAMPATH}$.
   (e) Montrer que f s'exécute en temps polynomial.
   (f) Que peut-on en conclure ? On justifiera précisément la conclusion.
3. Soit $\texttt{LPATH} = \{\langle G, a, b, k \rangle \mid G$ contient un chemin simple de a à b de longueur **au moins** $k\}$.
   On veut montrer que LPATH est **NP**-complet en se basant sur le fait que UHAMPATH est **NP**-complet.
   (a) montrer que $\texttt{LPATH} \in \mathbf{NP}$.
   (b) Proposer une fonction de transformation f qui transforme un graphe orienté $< G, a, b > \in \texttt{UHAMPATH}$ en un graphe non orienté $< G', a, b, k > \in \texttt{LPATH}$, et vérifiant en plus les questions (c) à (e) suivante.
   (c) Montrer qu'avec cette transformation f, $w \in \texttt{UHAMPATH} \Rightarrow f(w) \in \texttt{LPATH}$.
   (d) Montrer qu'avec cette transformation f, $f(w) \in \texttt{LPATH} \Rightarrow w \in \texttt{UHAMPATH}$.
   (e) Montrer que f s'exécute en temps polynomial.
   (f) Que peut-on en conclure ? On justifiera précisément la conclusion.