Université Louis-Pasteur
UFR de Mathématiques et d'Informatique

6 juin 2005
Licence d'informatique

Examen de théorie des langages et calculabilité

Durée : 3h00
Tous documents autorisés. Appareils électroniques non autorisés.
Les exercices sont donnés dans l'ordre du cours mais sont indépendants et peuvent être traités dans n'importe quel ordre.
Les automates et machines de Turing seront exprimés soit sous forme formelle, soit sous forme d'un diagramme sagittal.

1 – rationalité

1.1. Montrez que le langage $L = \{s\ a^n \mid s \in \{a, b\}^*, |s| = n, n \geq 0\}$ est non rationnel.

1.2. On considère l'affirmation :

Tout sous ensemble d'un langage rationnel est un langage rationnel.

Cette affirmation est-elle vraie ? Prouvez votre réponse.

1.3. Soient L_1 un langage non rationnel et L_2 un langage fini quelconque.

a) Montrez que $L_1 \cup L_2$ est non rationnel.
b) Montrez que $L_1 - L_2$ est non rationnel.
c) Montrez que les conclusions des questions a) et b) ne sont pas vraies si L_2 n'est pas supposé fini.

2 – grammaires algébriques

Soit la grammaire $G = (V, \Sigma, R, S)$ définie par :

$V = \{S\}$,
$\Sigma = \{a, b\}$,
$R = \{S \to aSb \mid bSa \mid aSa \mid bSb \mid e\}$.

a) Quel est le langage engendré par G ?
b) Prouvez votre réponse.

3 – machines de Turing

3.1. On souhaite convertir des nombres écrits dans une base B_1 vers une base B_2.
Rappelons qu'un nombre écrit dans une base B peut être décomposé sous forme de puissances de B. Par exemple, $845_{10} = 8.10^2 + 4.10^1 + 5.10^0$.
Un nombre binaire est exprimé par un mot sur l'alphabet $\Sigma = \{0, 1\}$. Un nombre unaire est exprimé par un mot sur l'alphabet $\Sigma' = \{I\}$, chaque 'I' d'un tel mot représentant une unité. Par exemple, le nombre binaire 101 correspond au nombre unaire IIIII.

a) Construisez une machine de Turing qui effectue la conversion d'un mot représentant un nombre binaire en un mot représentant ce même nombre en base 1. Pour simplifier la conception, cette machine pourra être décomposée en sous-machines.
b) Expliquez comment modifier la machine construite en a) pour qu'elle fasse des conversions de nombres exprimés en une base B_1 quelconque vers un nombre unaire.
c) Construisez une machine de Turing qui effectue la conversion d'un mot représentant un nombre unaire en un mot représentant ce même nombre en base 2. Cette machine pourra également être décomposée en sous-machines.
d) Expliquez comment modifier la machine construite en c) pour qu'elle fasse des conversions de nombres unaires vers un nombre exprimé en une base B_2 quelconque.
e) Déduisez de tout ce qui précède une machine qui fait la conversion de nombres écrits dans une base B_1 vers une base B_2.

3.2. Montrez qu'une machine de Turing peut être simulée par un automate sans ruban mais avec deux piles et deux symboles différents.

…/…

4 – décidabilité – indécidabilité

4.1. Soit un automate à pile M tel que M comporte un et un seul état.

a) Montrez que M accepte tout mot $w \in \Sigma^*$ si et seulement si il accepte tout mot $w' \in \Sigma^*$ de longueur 1.

b) Montrez si le problème suivant est décidable ou non :

Etant donné un automate à pile M à un et un seul état, déterminer si $L(M) = \Sigma^*$.

4.2. Pour chacun des problèmes suivants, montrez s'il est décidable ou non :

a) Etant donnés une machine de Turing M et un symbole *a*, déterminer si M écrit toujours *a* avec le mot vide en entrée.

b) Etant donnée une machine de Turing M, déterminer s'il existe des entrées w telles que M entre dans tous ses états pendant le calcul de w.

5 – complexité

On appelle 3-COLORING le problème suivant :

Soit G un graphe non orienté.
Déterminer si les nœuds de G peuvent être coloriés avec 3 couleurs de manière à ce que deux nœuds adjacents n'aient pas la même couleur.

a) Montrez que 3-COLORING est dépendant du graphe G. Est-il toujours possible de le résoudre ?

b) Montrez que 3-COLORING appartient à la classe NP.

c) Décrivez une réduction polynomiale de SAT vers 3-COLORING.

d) Justifiez pourquoi 3-COLORING n'est pas NP-complet.

6 – un petit jeu pour finir

Une assemblée de *n* joueurs souhaite jouer au tarot. Le tarot peut se jouer à 3, 4 ou 5, mais nous ne retiendrons ici que le jeu à 5. Nous posons donc $n > 5$. Ces joueurs ne disposent malheureusement que d'un seul jeu. Une solution consiste à distribuer 5 jetons, et chacun des joueurs possédant un jeton jouera la partie. A la fin de la partie, chaque joueur détenant un jeton le donne à un autre. On arrivera ainsi à ce que chaque partie soit jouée par 5 joueurs, et que ces 5 joueurs ne sont pas toujours les mêmes.
Chaque joueur souhaite évidemment jouer le plus souvent possible. On se pose donc le problème suivant :

Déterminer si tous les joueurs jouent un même nombre de parties.

a) Construisez, ou décrivez précisément le fonctionnement d'une machine (automate avec ou sans pile, ou machine de Turing) qui simule la répartition des joueurs dans une succession de parties.

b) Quelle est la complexité du problème énoncé ci-dessus ? Prouvez votre réponse.

c) Proposez un algorithme particulier permettant de rendre la répartition des parties équitable. Quelle est la complexité de votre proposition ? Prouvez votre réponse.

Nb. On ne s'occupe ici que de la répartition de 5 joueurs parmi *n* joueurs, pas du jeu de tarot lui-même !