Université Bordeaux 1
Master-1 d'Informatique, 2010–2011

Modèles de Calcul : feuille 2

Machines de Turing

I. Machines simples

Dans cette série d'exercices, il faut représenter la machine de Turing avec tous les détails : l'alphabet de la bande, l'ensemble d'états, la position initiale de la tête, la fonction de transition, etc.

Exercice 2.1

Écrire une machine de Turing qui calcule la fonction $n \to n+1$, les entiers étant codés en binaire.

Exercice 2.2

Écrire une machine de Turing qui calcule la fonction

$$n \to \begin{cases} 0 & \text{si } \ n = 0, \\ n-1 & \text{sinon,} \end{cases}$$

les entiers étant codés en binaire.

Exercice 2.3

1. Donner une machine de Turing à deux bandes qui calcule la représentation en binaire d'un entier donné en représentation unaire.
2. Donner une machine de Turing à une bande qui fait la même chose.

Exercice 2.4

Construire une machine de Turing dont la tête visite toutes les cases de la bande.

Exercice 2.5

Construire une machine de Turing à trois bandes qui calcule la somme de deux entiers représentés en binaire.

Exercice 2.6

Construire une machine de Turing à une bande qui calcule la fonction $x \to 3x$, x étant un entier représenté en binaire.

Exercice 2.7

1. Construire une machine de Turing qui vérifie l'égalité de deux mots donnés.
2. Construire une machine de Turing qui recopie le mot d'entrée à un autre endroit de la bande.

Exercice 2.8

1. Soit $A = \{a, b, c, d\}$. Construire une machine de Turing acceptant le langage L des mots $u \in A^*$ ne contenant pas le facteur ad.
2. De façon générale, étant donné un langage rationnel L, comment construire une machine de Turing reconnaissant le langage L ?

II. Machines plus avancées

Dans cette série, il faut décrire chaque machine de manière informelle (mais exacte).

Exercice 2.9

1. Construire une machine de Turing reconnaissant les mots de Dyck.
2. Construire une machine de Turing reconnaissant les mots sur $\{a, b\}$ ayant le même nombre d'occurrences de a et de b.
3. Construire une machine de Turing reconnaissant $\{u \in \{a, b, c\}^* \ / \ |u|_a = |u|_b = |u|_c\}$.

Exercice 2.10

Construire une machine de Turing à trois bandes qui calcule le produit de deux nombres représentés en binaire.

Exercice 2.11

Décrire une machine de Turing qui reconnaît le langage $L \subset \{a, b, \#\}^*$ possédant les mots $u\#v$ où u précède v en ordre lexicographique, et seulement ces mots.

III. Simulation

Exercice 2.12

Toute machine de Turing peut être simulée par une autre machine dont la tête se déplace (à droite ou à gauche) lors de chacune de ses opérations et ne reste jamais sur la même case.

Exercice 2.13

Simuler une machine de Turing à une bande infinie dans les deux sens par une machine à une bande infinie dans un seul sens.

Exercice 2.14

1. Simuler une machine de Turing à deux bandes par une machine à une seule bande.
2. Si on définit la complexité de calcul comme le nombre d'opérations faites par une machine de Turing, quel est alors le rapport entre les complexités de la machine initiale (à deux bandes) et de la machine à une bande qui la simule ?

Exercice 2.15

Montrer que l'on peut simuler toute machine de Turing par une autre machine de Turing qui n'utilise pour son alphabet de travail que les lettres 0, 1 et $\square$.

Conclusion. Toute machine de Turing peut être simulée par une machine dite *canonique* : une seule bande infinie dans un sens et avec l'alphabet de travail $\{0, 1, \square\}$. Ainsi, pour programmer, on peut utiliser n'importe quel alphabet et autant de bandes que l'on veut, et pour démontrer des théorèmes on peut utiliser des machines canoniques.

IV. Décidabilité

Exercice 2.16

Le problème de déterminer si une machine de Turing s'arrête sur tout mot d'entrée après au plus 5 pas de calcul est-il décidable ? Justifier.

Exercice 2.17

Le problème de l'inversion de marche. On considère le problème suivant : étant donnée une machine de Turing M qui se déplace (à gauche ou à droite) lors de chacune de ses transitions, et un mot d'entrée x, déterminer si M, en travaillant sur l'entrée x, se déplace toujours à droite ou au moins une fois vers la gauche. On suppose qu'au début du travail la tête de lecture est à gauche du mot. Ce problème est-il décidable ou indécidable ?

Exercice 2.18

Montrer que le problème suivant est décidable : étant donné une machine de Turing M, un mot d'entrée x et un entier n, déterminer si M, en travaillant sur x, sort du segment de bande contenant le mot x, n cases à droite de x et n cases à gauche de x. On suppose qu'au début du travail la tête de lecture est au début du mot.

Exercice 2.19

Montrer que le problème de l'arrêt de machine de Turing *sur la bande vide* est indécidable.
Indication. Utiliser l'indécidabilité du problème de l'arrêt standard : pour une paire (M, x) où M est une machine de Turing et x est un mot d'entrée, dire si M, étant appliquée à x, s'arrête ou non.

Exercice 2.20

Montrer que le problème suivant est indécidable :

> Instance : une paire (M, x) où M est une machine de Turing et x est un mot d'entrée ;
> Question : la machine M, en travaillant sur x, passe-t-elle par tous ses états ?

Remarque : dans ce problème, on ne demande pas si M s'arrête ou non.

Exercice 2.21

Soit $\mathcal{M}$ la classe des machines de Turing ayant l'alphabet de travail $\{0, 1, \Box\}$. Dans les questions suivantes on suppose qu'au début de travail la bande d'entrée est vide.

1. Soit $\mathcal{M}_n \subset \mathcal{M}$, $n \in \mathbb{N}$, l'ensemble des machines ayant exactement n états. (On suppose que pour $n = 0$ la machine en question ne fait rien et s'arrête immédiatement.) Montrer que l'ensemble $\mathcal{M}_n$ est fini.
2. Soit $\mathcal{A}_n \subset \mathcal{M}_n$ l'ensemble des machines qui s'arrêtent sur la bande vide. Montrer que de telles machines existent et l'ensemble $\mathcal{A}_n$ est donc non vide.
3. Soit $T(n)$ le plus grand nombre d'opérations faites par une machine appartenant à $\mathcal{A}_n$. Expliquer pourquoi la fonction $T(n)$ est définie pour tout n.
4. Montrer que la fonction $T(n)$ ne peut être majorée par aucune fonction calculable. C'est-à-dire, il n'existe pas de fonction calculable et totale $f(n)$ telle que $T(n) \leq f(n)$ pour tout n.
5. Expliquer pourquoi la dernire affirmation est très surprenante.