Calculabilité

TP7

Y. Deville
C-H. Bertrand Van Ouytsel - V. Coppé - A. Gerniers
N. Golenvaux - M. Parmentier

Avril 2021

Questions du test

Voici la description d'une machine de Turing avec comme alphabets $\Sigma = \{a, b\}$ et $\Gamma = \{a, b, x, B, 0, 1\}$.

1. Résultat pour l'entrée $bbba$?

   Réponse : 0

2. Résultat pour l'entrée $baba$?

   Réponse : 1

3. Quelle seront les deux premières cases dont le contenu sera modifié en x si l'entrée commence par un a ?

   Réponse : La première case ($=$ premier a) et le premier b

state	symbol	state	movement	symbol
start	a	seekB	$\rightarrow$	x
start	b	seekA	$\rightarrow$	x
start	x	start	$\rightarrow$	B
start	B	stop	$\downarrow$	1
seekA	a	restart	$\leftarrow$	x
seekA	b	seekA	$\rightarrow$	b
seekA	x	seekA	$\rightarrow$	x
seekA	B	false	$\leftarrow$	B
seekB	b	restart	$\leftarrow$	x
seekB	a	seekB	$\rightarrow$	a
seekB	x	seekB	$\rightarrow$	x
seekB	B	false	$\leftarrow$	B
restart	a	restart	$\leftarrow$	a
restart	b	restart	$\leftarrow$	b
restart	x	restart	$\leftarrow$	x
restart	B	start	$\rightarrow$	B
false	a	false	$\leftarrow$	B
false	b	false	$\leftarrow$	B
false	x	false	$\leftarrow$	B
false	B	stop	$\downarrow$	0

Questions du test

Voici la description d'une machine de Turing avec comme alphabets $\Sigma = \{a, b\}$ et $\Gamma = \{a, b, x, B, 0, 1\}$.

4. Que signifie le symbole x (point de vue exécution) ?

   Réponse : Emplacement d'un symbole d'entrée qui a déjà été vérifié par la machine de Turing

5. Que calcule cette machine de Turing ?

   Réponse : Si l'entrée a le même nombre de a que de b

state	symbol	state	movement	symbol
start	a	seekB	$\rightarrow$	x
start	b	seekA	$\rightarrow$	x
start	x	start	$\rightarrow$	B
start	B	stop	$\downarrow$	1
seekA	a	restart	$\leftarrow$	x
seekA	b	seekA	$\rightarrow$	b
seekA	x	seekA	$\rightarrow$	x
seekA	B	false	$\leftarrow$	B
seekB	b	restart	$\leftarrow$	x
seekB	a	seekB	$\rightarrow$	a
seekB	x	seekB	$\rightarrow$	x
seekB	B	false	$\leftarrow$	B
restart	a	restart	$\leftarrow$	a
restart	b	restart	$\leftarrow$	b
restart	x	restart	$\leftarrow$	x
restart	B	start	$\rightarrow$	B
false	a	false	$\leftarrow$	B
false	b	false	$\leftarrow$	B
false	x	false	$\leftarrow$	B
false	B	stop	$\downarrow$	0

Questions du test

6. Si une machine de Turing permet de décider un langage L, alors L est récursif.

   Réponse : Vrai, si une machine de Turing peut décider L alors L est T-récursif. Par la thèse de Church-Turing, un ensemble T-récursif est récursif.

7. L'exécution d'une machine de Turing se termine toujours.

   Réponse : Faux, on peut par exemple imaginer une machine qui alternerait entre des mouvements droite/gauche indéfiniment.

Questions du test

8. Les machines de Turing non déterministes sont plus puissantes (permettent de calculer davantage de fonctions) que les machines de Turing déterministes.

   Réponse : Faux, il existe une machine de Turing qui interprète les machines de Turing non déterministes. Les fonctions calculables par les machines non déterministes sont donc calculables par les déterministes. Elles ont ainsi la même puissance.

9. Les machines de Turing équipées d'un oracle pour l'ensemble

$$A = \{x \in \mathbb{N} \mid x\%2 = 0\}$$

   sont plus puissantes que les machines de Turing de base.

   Réponse : Faux.

   Décider A est possible sans oracle (e.g. $P(x) \equiv$ `return 1-(x%2)`). Une machine de Turing équipée d'un tel oracle n'est donc pas plus puissante qu'une machine de Turing de base.

Questions du test

10. Les machines de Turing équipées d'un oracle pour l'ensemble

$$A = \{x \in \mathbb{N} \mid \varphi_x(x) = 42\}$$

sont plus puissantes que les machines de Turing de base.

Réponse : Vrai. On peut montrer que A est non-récursif.

$A \neq \emptyset$ (e.g. $P(x) \equiv$ `return 42`) et $A \neq \mathbb{N}$ (e.g. $P(x) \equiv$ `return 0`) :

$$\implies \forall x \in A, \ : \varphi_x(x) = 42 \ \text{ et } \ \forall x \in \overline{A} \ : \varphi_x(x) \neq 0$$

$$\implies \forall i \in A, \forall j \in \overline{A}, \exists x : \varphi_i(x) \neq \varphi_j(x)$$

Par le théorème de Rice, on sait donc que A est non-récursif. Une machine de Turing équipée d'un tel oracle serait donc, par exemple, capable d'évaluer la fonction caractéristique de A, ensemble non récursif, à l'inverse d'une machine de Turing de base qui ne le pourrait pas.

Question 1 du TP

Question : For each of the following functions, build a Turing machine, using $\Sigma = \{0, 1\}$:

1. $f(n) = n \times 2$
2. $f(n) = \text{not}(n)$ (f inverts the bits of n)

Check them with a Turing machine simulator and simulate them with different inputs.

Question 1 du TP

Question : For each of the following functions, build a Turing machine, using $\Sigma = \{0, 1\}$:

1. $f(n) = n \times 2$

Réponse :

$$\Gamma = \{B, 0, 1\}$$

start	0	start	$\rightarrow$	0
start	1	start	$\rightarrow$	1
start	B	stop	$\downarrow$	0

Question 1 du TP

Question : For each of the following functions, build a Turing machine, using $\Sigma = \{0, 1\}$:

2. $f(n) = \text{not}(n)$ (f inverts the bits of n)

Réponse :

$$\Gamma = \{B, 0, 1\}$$

start	0	start	$\rightarrow$	1
start	1	start	$\rightarrow$	0
start	B	stop	$\downarrow$	B

Question 2 du TP

Question : Suppose we have two Turing machines A and B with $\Sigma = \{a, b\}$. Their starting and halting states are (s_A, h_A) and (s_B, h_B), respectively. Explain how to combine A and B into a new Turing machine implementing the following pseudo-code :

1. if(a) then $\{A\}$ else $\{B\}$
2. while not(a) do $\{A\}$

In this pseudocode, if(a) means "if a is the symbol under the head", and not(a) means "the symbol under the head is not a", none of which consumes a symbol.

Question 2 du TP

Question : Suppose we have two Turing machines A and B with $\Sigma = \{a, b\}$. Their starting and halting states are (s_A, h_A) and (s_B, h_B), respectively. Explain how to combine A and B into a new Turing machine implementing the following pseudo-code :

1. if(a) then $\{A\}$ else $\{B\}$

Réponse :

start	a	s_A	$\downarrow$	a
start	b	s_B	$\downarrow$	b
start	B	s_B	$\downarrow$	B
h_A	a	stop	$\downarrow$	a
h_A	b	stop	$\downarrow$	b
h_A	B	stop	$\downarrow$	B
h_B	a	stop	$\downarrow$	a
h_B	b	stop	$\downarrow$	b
h_B	B	stop	$\downarrow$	B

Question 2 du TP

Question : Suppose we have two Turing machines A and B with $\Sigma = \{a, b\}$. Their starting and halting states are (s_A, h_A) and (s_B, h_B), respectively. Explain how to combine A and B into a new Turing machine implementing the following pseudo-code :

2. while not(a) do $\{A\}$

Réponse :

start	a	stop	$\downarrow$	a
start	b	s_A	$\downarrow$	b
start	B	s_A	$\downarrow$	B
h_A	a	stop	$\downarrow$	a
h_A	b	s_A	$\downarrow$	b
h_A	B	s_A	$\downarrow$	B

Question 3 du TP

Question : Are those Turing machines equivalent to the standard model of Turing machine ? Explain.

1. "Big jumps" : this Turing machine can move its head of n cells to the left and to the right. The transition function is thus $S \times \Gamma \to S \times (\{L, R\} \times \mathbb{N}) \times \Gamma$.
2. "Online" : this Turing machine can only move to the right or stay at the same cell, and cannot move to the left. The transition function is thus
   $S \times \Gamma \to S \times \{R, \downarrow\} \times \Gamma$.

Question 3 du TP

Question : Are those Turing machines equivalent to the standard model of Turing machine ? Explain.

1. "Big jumps" : this Turing machine can move its head of n cells to the left and to the right. The transition function is thus $S \times \Gamma \to S \times (\{L, R\} \times \mathbb{N}) \times \Gamma$.

Réponse : Vrai
Il y a moins d'instructions à exécuter (meilleur efficacité), mais on conserve la même puissance : on peut remplacer

s	x	s'	$\to_n$	x'

par

s	x	right_{n-1}	$\to$	x'
right_{n-1}	?	right_{n-2}	$\to$	?
$\vdots$				
right_1	?	s'	$\to$	?

Question 3 du TP

Question : Are those Turing machines equivalent to the standard model of Turing machine ? Explain.

2. "Online" : this Turing machine can only move to the right or stay at the same cell, and cannot move to the left. The transition function is thus
   $S \times \Gamma \to S \times \{R, \downarrow\} \times \Gamma$.

Réponse : Faux

Ce modèle est restrictif. Exemples impossibles à résoudre :

- Inverser le 1er bit si le dernier est 0
- Ajouter 1 à un nombre binaire
- L'exemple du test

Question 4 du TP

Question : Is it possible to simulate any deterministic finite automaton with a Turing machine ? Explain.

Question 4 du TP

Question : Is it possible to simulate any deterministic finite automaton with a Turing machine ? Explain.

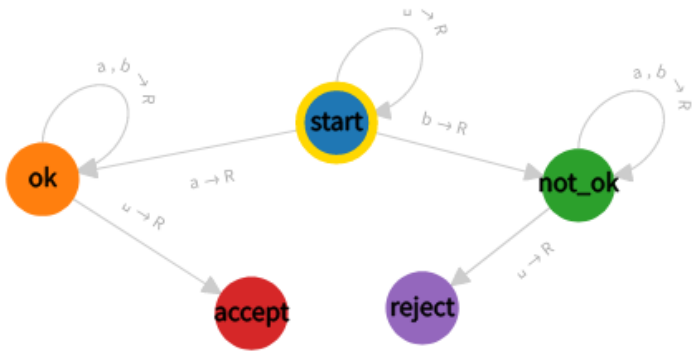
Réponse : Vrai. Il est toujours possible de simuler un automate avec une machine de Turing (modèle plus puissant).

Dans ces automates, les caractères de l'input sont lus progressivement : un à chaque transition d'état. Une fois tous les caractères lus, l'automate accepte ou rejette l'input en fonction de son état final.

Une machine de Turing peut facilement simuler un tel comportement. Chaque état de l'automate équivaut à un état de la machine de Turing. Ensuite, pour chaque transition de l'automate, nous lisons la valeur courante et changeant d'état en accord avec celle-ci. Lors de cette transition, nous écrivons une valeur quelconque sur le ruban et passons à la valeur de droite. Un symbole blanc (B) est également nécessaire pour indiquer la fin de la gestion de l'input et la transition vers un état acceptant/de rejet.

Question 4 du TP

Un exemple de transformation est présenté ci-dessous pour un automate vérifiant que l'input commence par la lettre a.



$$\Gamma = \{B, a, b\}$$

start	a	ok	$\rightarrow$	a
start	b	not_ok	$\rightarrow$	b
start	B	reject	$\rightarrow$	B
ok	a	ok	$\rightarrow$	a
ok	b	ok	$\rightarrow$	b
ok	B	accept	$\rightarrow$	B
not_ok	a	not_ok	$\rightarrow$	a
not_ok	b	not_ok	$\rightarrow$	b
not_ok	B	reject	$\rightarrow$	B
accept	B	accept	$\downarrow$	B
reject	B	reject	$\downarrow$	B

Question 5 du TP

Question : What is the computational complexity of this Turing machine ?

Let TM2T be a Turing machine able to use two strips of tapes instead of one. Show that there exists a TM2T that computes the same function as this traditional TM, while achieving a quadratic computational speedup.

This Turing machine has the alphabets $\Sigma = \{a, b\}$ and $\Gamma = \{a, b, x, B, 0, 1\}$.

state	symbol	state	movement	symbol
start	a	seekB	$\rightarrow$	x
start	b	seekA	$\rightarrow$	x
start	x	start	$\rightarrow$	B
start	B	stop	$\downarrow$	1
seekA	a	restart	$\leftarrow$	x
seekA	b	seekA	$\rightarrow$	b
seekA	x	seekA	$\rightarrow$	x
seekA	B	false	$\leftarrow$	B
seekB	b	restart	$\leftarrow$	x
seekB	a	seekB	$\rightarrow$	a
seekB	x	seekB	$\rightarrow$	x
seekB	B	false	$\leftarrow$	B
restart	a	restart	$\leftarrow$	a
restart	b	restart	$\leftarrow$	b
restart	x	restart	$\leftarrow$	x
restart	B	start	$\rightarrow$	B
false	a	false	$\leftarrow$	B
false	b	false	$\leftarrow$	B
false	x	false	$\leftarrow$	B
false	B	stop	$\downarrow$	0

Question 5 du TP

Indices :

- Pour la complexité de la TM normale, considérez le pire cas : $a \dots ab \dots b$ (un mot de longueur n avec $\frac{n}{2}$ fois a suivi de $\frac{n}{2}$ fois b).
- Pour la TM2T, une bande ne regarde que les a, l'autre bande ne regarde que les b.

Solution :

- Pour le 1er a, il faut aller jusqu'au 1er b et revenir, ce qui fait $\mathcal{O}(n)$ opérations. Pareil pour le 2ème a, le 3ème, etc. On a donc $\mathcal{O}(n^2)$ opérations.
- Définissons que la bande 1 s'occupe des a, et la bande 2 s'occupe des b. Le but est de placer la tête de lecture 1 sur un a et la tête 2 sur un b. Dans ce cas, on a une paire et on peut effacer ces symboles (les remplacer par B) et continuer. En pratique, si a est le premier symbole par exemple, la tête 1 reste dessus, pendant que la tête 2 avance jusqu'au premier b, effaçant tous les a sur son chemin. Quand elles sont toutes les deux sur leur symbole, on efface les symboles et les têtes avancent chacune jusqu'au moment où elle retombent sur leurs symboles respectifs, après quoi on recommence. Quand une des têtes a lu toute l'entrée, et arrive donc sur un B, deux cas peuvent se présenter : l'autre tête n'arrive pas à trouver un de ses symboles et finit par tomber sur B, dans ce cas il faut mettre l'output à 1. Si par contre elle arrive à trouver un de ses symboles, c'est qu'il y a un nombre différent de a que de b, et il faut mettre l'output à 0.
- Comme chaque tête a parcouru une fois toute l'entrée, on a visité $2n$ case. La complexité est donc $\mathcal{O}(n)$, et on a bien un speedup quadratique.