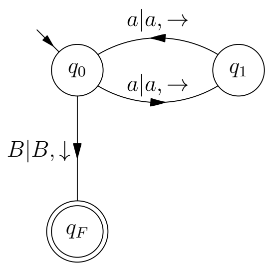
TD1 : Machines de Turing

Rappel de notations :

- Une MT $\mathcal{M}$ est un tuple $(Q, \Gamma, \Sigma, \delta, q_0, B, q_F)$.
- La fonction de transition partielle est $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \downarrow, \to\}$.
- On note uqv la configuration $(q, {}^{\omega}Bu.vB^{\omega}, z)$ pour toute position $z \in \mathbb{Z}$.
- La configuration initiale associée à l'entrée $w \in \Sigma^*$ est $c_0(w) = q_0 w = (q_0, {}^{\omega}B.wB^{\omega}, 0)$.

Definition. Un mot $w \in \Sigma^*$ est *accepté* par $\mathcal{M}$ lorsque le calcul commençant avec la configuration intitiale associée au mot w s'arrête et que l'état lors de l'arrêt est q_F. Sinon, le mot est *rejeté* par $\mathcal{M}$. Le *langage reconnu* par $\mathcal{M}$ est l'ensemble des mots acceptés par $\mathcal{M}$.

Exercice 1. On peut définir la fonction de transition d'une MT à l'aide d'un automate fini, par exemple :



état	lettre	état	lettre	dépl.
q_0	a	q_1	a	$\to$
q_0	B	q_F	B	$\downarrow$
q_1	a	q_0	a	$\to$

FIGURE 1 – L'automate à gauche décrit la fonction de transition de la table de droite.

(1) Quel est le langage reconnu par la MT de la figure 1 ?

(2) Donner une MT avec $\Sigma = \{a, b, c\}$ qui reconnaît $\Sigma^* aba \Sigma^*$.

(3) Donner une MT qui reconnaît le langage $\{0^k 1^k, k \in \mathbb{N}\}$ avec $\Sigma = \{0, 1\}$.

(4) Donner une MT qui reconnaît le langage $\{a^{2^k}, k \in \mathbb{N}\}$ avec $\Sigma = \{a\}$.

On peut voir une MT comme un programme qui calcule une fonction. Dans ce cas, l'image d'un mot $w \in \Sigma^*$ est obtenue, lorsque le calcul partant de la configuration initiale associée à w s'arrête, en lisant le mot qui commence à la position de la tête et va jusqu'au premier symbole B. Si le calcul ne s'arrête pas, la fonction n'est pas définie sur w.

Exercice 2. Dans cet exercice, on décrit des MT qui calculent des fonctions.

(1) Quelle est la fonction calculée par la MT de la figure 2 ?

(2) Donner une MT calculant $n \mapsto n + 1$ avec $\Sigma = \{0, 1\}$ en interprétant un mot sur Σ comme un entier en base 2.

(3) Donner une MT calculant le plus grand chiffre présent dans l'écriture d'un nombre en base 3. (Par exemple $101 \mapsto 1$ et $20 \mapsto 2$.)

(4) Donner une MT qui sur l'entrée $w = w_0 w_1 \ldots w_n$ calcule son miroir : $w_n \ldots w_1 w_0$.

Exercice 3. Pour aller plus loin en programmation Turing.

(1) Donner une MT qui reconnaît le langage des mots bien parenthésés. Ici $\Sigma = \{(,)\}$.

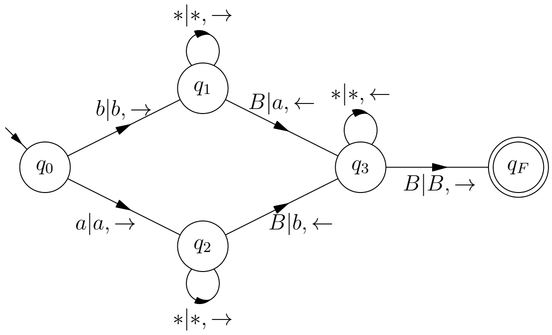


FIGURE 2 – Dans cet automate, $*$ est utilisé pour représenter une lettre de l'alphabet non utilisée par une autre transition partant du même état.

(2) Donner une MT qui reconnaît l'ensemble des mots $u\#v$ tels que u est un facteur de v (il existe w_1 et w_2 tels que $v = w_1uw_2$) sur l'alphabet $\Sigma = \{a, b\}$.

(3) Donner une MT qui prend en entrée 2 nombres m et n et qui calcule la partie entière de la moyenne de n et m.

(4) Donner une MT calculant le nombre d'occurrences du motif aa dans un mot sur l'alphabet $\Sigma = \{a, b, c\}$.

Exercice 4. En utilisant les idées vues dans les exercices précédents,

(1) Donner des MT qui calculent l'addition et la multiplication de deux nombres binaires.

(2) Donner une MT qui prend en entrée une expression arithmétique (non parenthésée) sur l'alphabet $\{0, 1, +, \times\}$ dans laquelle les nombres sont donnés en binaire et calcule le résultat de l'opération.

(3) La substitution de Thue-Morse est la fonction $\sigma : \left\{ \begin{array}{ccc} a & \mapsto & ab \\ b & \mapsto & ba \end{array} \right.$. On peut la généraliser à tout mot sur l'alphabet $\Sigma = \{a, b\}$ avec $\forall u, v \in \Sigma^*, \sigma(uv) = \sigma(u)\sigma(v)$. Donner une MT qui sur l'entrée (u, k) calcule $\sigma^k(u)$. (Par exemple $\sigma^3(a) = \sigma^2(ab) = \sigma(abba) = abbabaab$.)