TD 6 – Machines de Turing

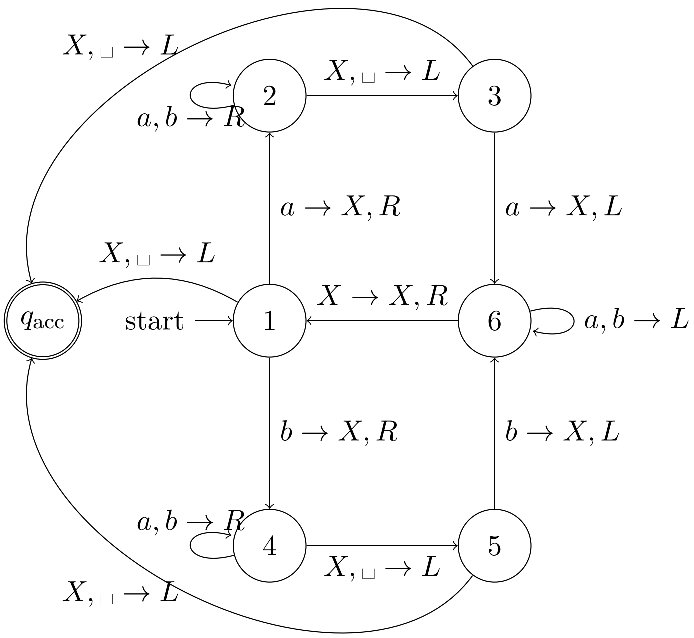
1 Décision de langages et calcul de fonctions par des machines de Turing

Exercice 1. Pour chacun des langages suivants, donner (explicitement, avec tous les détails) une machine de Turing (MT) le décidant.

1. $\{w \in \{a,b\}^* \mid w = w^{\mathcal{R}}\}$ (langage des palindromes sur $\{a,b\}$).
2. $\{a^n b^n c^n \mid n \in \mathbb{N}\}$.
3. $\{a^{2^n} \mid n \in \mathbb{N}\}$.
4. $\{ww \mid w \in \{a,b\}^*\}$.

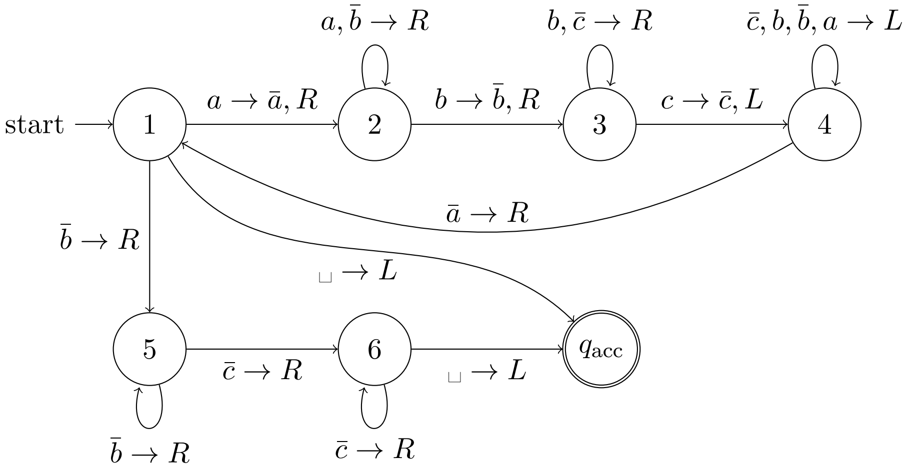
Solution 1.

1. On construit la MT suivante :



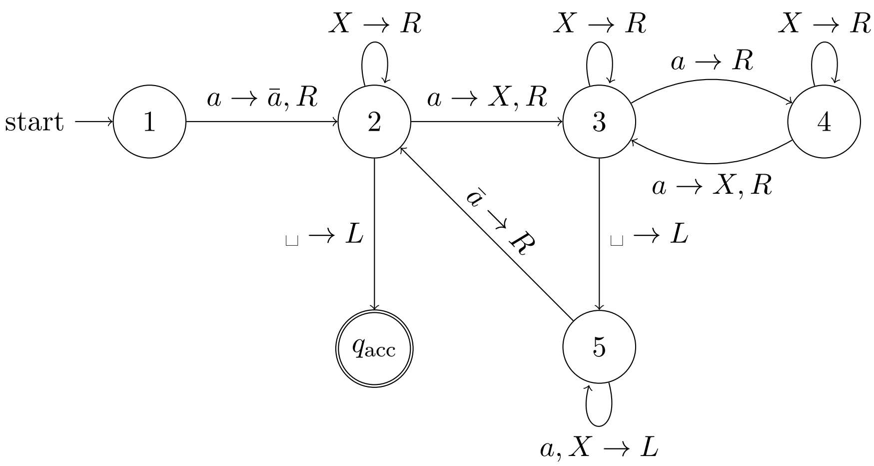
.

2. On construit la MT suivante :



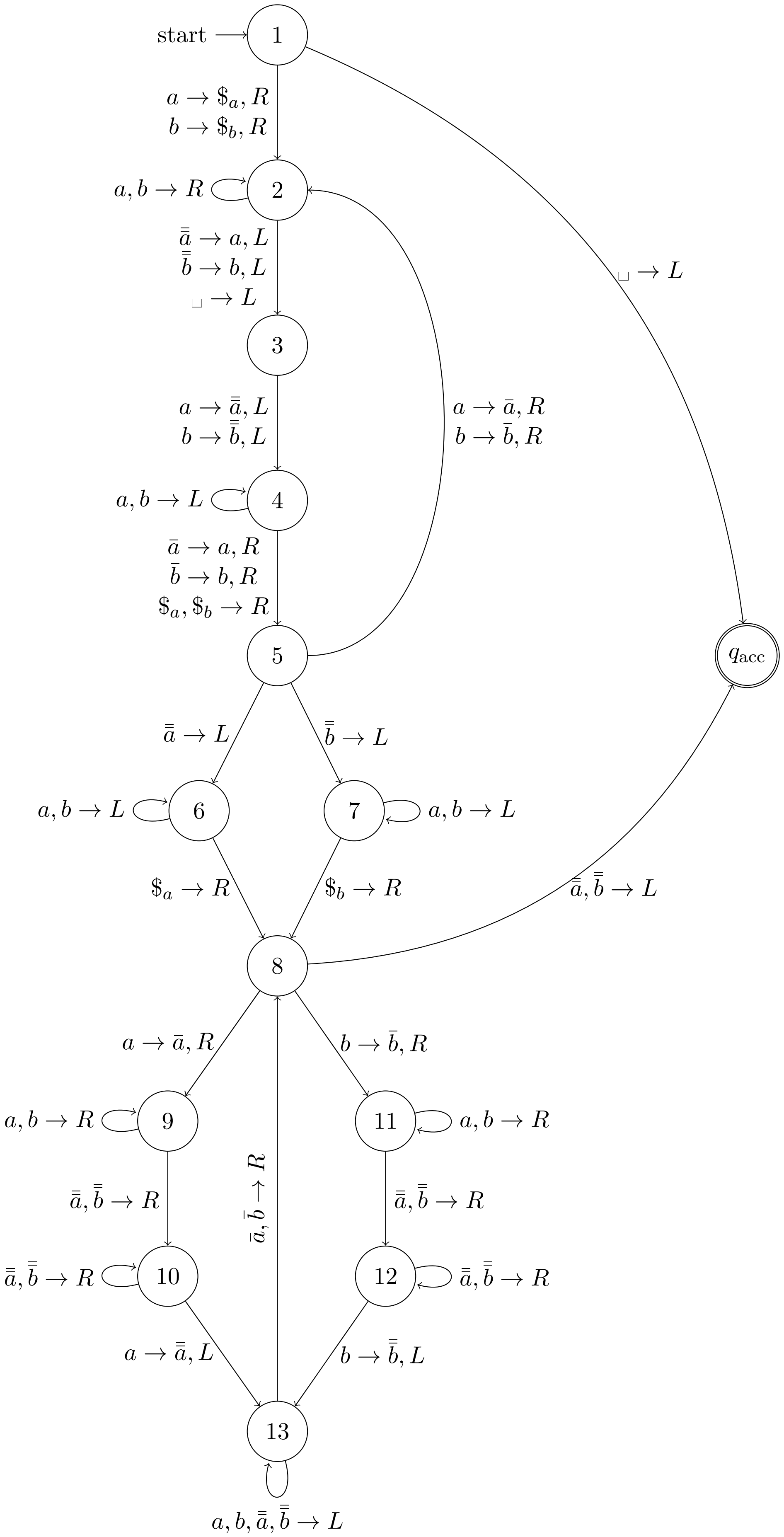
.

3. On construit la MT suivante :



.

4. On construit la MT suivante :



.

Une MT $\mathcal{M} = (Q, \Sigma, \Gamma, q_0, q_{\text{acc}}, q_{\text{rej}})$ qui est un décideur peut également être vue comme *calculant* la fonction $f\colon \Sigma^* \to \mathrm{T}^*$ telle que $\mathrm{T} \subseteq \Gamma$ et pour tout $w \in \Sigma^*$, $f(w) = uv$ où $q_0 w \vdash^*_{\mathcal{M}} u q_{\mathrm{acc}} v$ ou $q_0 w \vdash^*_{\mathcal{M}} u q_{\mathrm{rej}} v$.

Pour $n \in \mathbb{N}$ un entier naturel, on dénote par $\langle n \rangle$ sa représentation binaire gros-boutiste.

Exercice 2. Pour chacune des fonctions suivantes, donner (explicitement, avec tous les détails) une MT la calculant.

1. La fonction $r\colon \{0,1\}^* \to \{0,1\}^*$ telle que, pour tout $w \in \{0,1\}^*$, $r(w) = w^{\mathcal{R}}$. (C'est une fonction renversant une chaîne de bits.)
2. La fonction $s\colon \{0,1,\#\}^* \to \{0,1\}^*$ telle que, pour tout $w \in \{0,1,\#\}^*$,

$$s(w) = \begin{cases} \langle n_1 + n_2 \rangle & \text{si } w = \langle n_1 \rangle \# \langle n_2 \rangle \text{ pour } n_1, n_2 \in \mathbb{N} \\ \varepsilon & \text{sinon .} \end{cases}$$

   (C'est une fonction additionnant deux entiers.)
3. Étant donné deux fonctions $f_1\colon \Sigma^* \to \Gamma^*$ et $f_2\colon \Gamma^* \to \mathrm{T}^*$ (où $\Sigma, \Gamma, \mathrm{T}$ sont des alphabets) calculées respectivement par les MT $\mathcal{M}_1$ et $\mathcal{M}_2$, la fonction $f_2 \circ f_1\colon \Sigma^* \to \mathrm{T}^*$ composition de f_1 et f_2.

Solution 2. Cela se fait de manière similaire à l'exercice précédent.

2 Reconnaissance de langages par des machines de Turing

On appellera dans la suite *MT normalisée*, une MT $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_1, q_2, q_3)$ telle que $Q = \{q_1, \ldots, q_n\}$ avec $n \in \mathbb{N}, n \geq 3$, $\Sigma = \{a_1, \ldots, a_l\}$ avec $l \in \mathbb{N}, l \geq 1$ et $\Gamma = \{a_1, \ldots, a_l, a_{l+1}, \ldots, a_k\}$ avec $k \in \mathbb{N}, k \geq l+1$, où a_k correspond au symbole blanc. Il n'est pas difficile de voir qu'à toute MT on peut (de manière non unique) associer une MT normalisée ayant exactement le même comportement.

Exercice 3. Proposer un encodage $\langle \cdot \rangle$ pour les MT normalisées, c'est-à-dire, une application qui à toute MT normalisée $\mathcal{M}$ associe un mot $\langle \mathcal{M} \rangle$ sur $\{0,1\}$ de manière injective et vérifiant qu'il existe une MT universelle pour cet encodage, c'est-à-dire, reconnaissant le langage des mots $\langle \mathcal{M} \rangle \# w$ où $\mathcal{M}$ est une MT et $w \in \{0,1\}^*$ encode un mot sur l'alphabet d'entrée de $\mathcal{M}$ que cette dernière accepte.

Solution 3. On peut définir $\langle \cdot \rangle$ comme étant l'application telle que pour toute MT normalisée $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_1, q_2, q_3)$ telle que $Q = \{q_1, \ldots, q_n\}$ avec $n \in \mathbb{N}, n \geq 3$, $\Sigma = \{a_1, \ldots, a_l\}$ avec $l \in \mathbb{N}, l \geq 1$ et $\Gamma = \{a_1, \ldots, a_l, a_{l+1}, \ldots, a_k\}$ avec $k \in \mathbb{N}, k \geq l+1$, on ait

$$\langle \mathcal{M} \rangle = 1^n 0001^l 0001^k 000 \Big(\prod_{i=4}^{n} \prod_{j=1}^{k} \langle (q_i, a_j), \delta(q_i, a_j) \rangle 00 \Big) \Big(\prod_{j=1}^{k-1} \langle (q_1, a_j), \delta(q_1, a_j) \rangle 00 \Big) \langle (q_1, a_k), \delta(q_1, a_k) \rangle \ ,$$

où pour tous $i, x \in [\![1, n]\!]$, $j, y \in [\![1, k]\!]$ et $D \in \{R, L\}$, $\langle (q_i, a_j), (q_x, a_y, D) \rangle = 1^i 01^j 01^x 01^y 0 \langle D \rangle$ avec $\langle D \rangle = \begin{cases} 1 & \text{si } D = R \\ 11 & \text{sinon} \end{cases}$.

Exercice 4. Pour chacun des langages suivants, décrire, sans donner tous les détails, une MT le reconnaissant.

1. $\{\langle \mathcal{M} \rangle \mid \mathcal{M}$ MT normalisée s'arrêtant sur le mot vide$\}$.
2. $\{\langle \mathcal{M} \rangle \mid \mathcal{M}$ MT normalisée telle que $\mathcal{L}(\mathcal{M}) \neq \emptyset\}$.

Solution 4. On suppose avoir une MT universelle $\mathcal{U}$ pour l'encodage $\langle \cdot \rangle$ qui, lorsqu'elle a en entrée $\langle \mathcal{M} \rangle \# w$ pour $\mathcal{M}$ une MT normalisée et $w \in \{0,1\}^*$ encodant un mot sur l'alphabet d'entrée de $\mathcal{M}$, s'arrête si et seulement si $\mathcal{M}$ s'arrête sur le mot encodé par w, ainsi qu'une MT $\mathcal{V}$ décidant le langage des mots sur $\{0,1\}$ correspondant à des encodages avec $\langle \cdot \rangle$ de MT normalisées.

1. On construit une MT $\mathcal{M}$ qui, étant donné un mot $w \in \{0,1\}^*$ en entrée, se comporte selon l'algorithme suivant :

si $\mathcal{V}(w)$ rejette **alors**
  rejeter
fin si
$\mathcal{U}(w, \varepsilon)$
accepter .

2. On ordonne les mots de $\{0,1\}^*$ selon l'ordre radiciel (le mot $x \in \{0,1\}^*$ est plus petit que le mot $y \in \{0,1\}^*$ lorsque $|x| < |y|$ ou $|x| = |y|$ et x est plus petit que y selon l'ordre lexicographique, ce qui revient aussi à dire que l'entier dont la représentation binaire est $1x$ est plus petit que l'entier dont la représentation binaire est $1y$) en une suite $(x_i)_{i \in \mathbb{N}}$.

   On construit une MT $\mathcal{M}$ qui, étant donné un mot $w \in \{0,1\}^*$ en entrée, se comporte selon l'algorithme suivant :

   si $\mathcal{V}(w)$ rejette **alors**
     rejeter
   fin si
   $i \leftarrow 0$
   tant que vrai **faire**
     pour $j \leftarrow 0 \dots i$ **faire**
       si $\mathcal{U}(w\#x_i)$ accepte en i étapes **alors**
         accepter
       fin si
     fin pour
   fin tant que.

3 Équivalence de modèles

Soit les variantes suivantes de MT.

— Une machine à $k \in \mathbb{N}_{>0}$ piles est une MT avec une bande d'entrée en lecture seule et k bandes de travail, où les bandes de travail sont remplacées par des piles (on peut ajouter et lire uniquement tout à droite).
— Une machine à file est une MT avec une bande d'entrée en lecture seule et une bande de travail, où la bande de travail est remplacée par une file (on peut ajouter uniquement tout à droite et lire uniquement tout à gauche).
— Une machine à $k \in \mathbb{N}_{>0}$ compteurs est une machine à k piles où l'alphabet de pile est $\{B, Z\}$ et Z est un symbole de fond de pile. Un entier i peut être stocké dans une pile en comptant le nombre de symboles B. On peut incrémenter, décrémenter le compteur et tester si le compteur est vide (symbole Z en tête de pile).

Exercice 5. Prouver les énoncés suivants, sans donner tous les détails.

1. Une MT est équivalente à une machine à deux piles.
2. Un MT est équivalente à une machine à une file.
3. Une machine à une pile peut être simulée par une machine à deux compteurs.
4. Une MT est équivalente à une machine à deux compteurs.

Solution 5.

1. Étant donné une MT $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, on voit que toute configuration uqv de $\mathcal{M}$ où $q \in Q$, $u \in \Gamma^*$ et $v \in \Gamma^+$ peut être stockée sur une machine à deux piles en stockant u sur la première pile et $v^{\mathcal{R}}$ sur la seconde pile. Lorsque la tête se déplace vers la gauche, il suffit de dépiler de la première pile et d'empiler sur la seconde, et lorsque la tête se déplace vers la droite, il suffit de dépiler sur la seconde pile et d'empiler sur la première.
2. Étant donné une MT $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$, on voit que toute configuration uqv de $\mathcal{M}$ où $q \in Q$, $u \in \Gamma^*$ et $v \in \Gamma^+$ peut être stockée sur une machine à file en stockant $v\#u^{\mathcal{R}}\#$ sur celle-ci (avec $\# \notin \Gamma$). Lorsque la tête se déplace vers la gauche, il suffit de déplacer la première

lettre de $u^{\mathcal{R}}$ au début de la file, après avoir modifié la première lettre de v. Lorsque la tête se déplace vers la droite, il suffit de déplacer la première lettre de v juste après le premier # de la file, en la modifiant. Ceci se fait en parcourant la file de gauche à droite (ce qui se fait simplement en retirant chaque lettre du début pour la rajouter à la fin, l'une après l'autre) et en comptant le nombre de # rencontrés.

3. Étant donné une pile sur un alphabet Γ et une bijection $\sigma\colon \Gamma \to [\![0, |\Gamma| - 1]\!]$, on stocke un contenu $u \in \Gamma^+$ de la pile en stockant l'entier $\sum_{i=0}^{|u|-1} |\Gamma|^i \sigma(u_{|u|-i})$ dans le premier compteur, le second compteur étant vide.

   Pour empiler une lettre $a \in \Sigma$ sur la pile, on multiplie l'entier du premier compteur par $|\Gamma|$, en décrémentant 1 par 1 le premier compteur tout en incrémentant par $|\Gamma|$ le second compteur, puis en recopiant le résultat dans le premier compteur, en y ajoutant $\sigma(a)$.

   Pour dépiler la lettre du haut de la pile, on effectue la division euclidienne par $|\Gamma|$ du premier compteur, en décrémentant $|\Gamma|$ par $|\Gamma|$ le premier compteur tout en incrémentant par 1 le second compteur : le premier compteur contient alors le reste correspondant à l'entier $\sigma(a)$ avec $a \in \Gamma$ le sommet de la pile, et le second compteur contient le dividende, l'entier représentant le nouveau contenu de la pile, que l'on recopie dans le premier compteur pour finir.

4. Il est possible de simuler une machine à 4 compteurs en utilisant une machine à 2 compteurs, en représentant le contenu $(n_1, n_2, n_3, n_4) \in \mathbb{N}^4$ des 4 compteurs par l'entier ${p_1}^{n_1}{p_2}^{n_2}{p_3}^{n_3}{p_4}^{n_4}$ avec $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7$ dans le premier des deux compteurs. Pour incrémenter le compteur $i \in [\![1, 4]\!]$, on multiplie par p_i le contenu du premier des deux compteurs, pour le décrémenter, on divise par p_i le contenu du premier des deux compteurs, et pour tester s'il est vide, on teste si le contenu du premier des deux compteurs est différent de $0 \mod p_i$, à chaque fois en utilisant le second des deux compteurs.

   Une MT peut être simulée par une machine à 2 piles, qui peut être simulée par une machine à 4 compteurs, et donc par une machine à 2 compteurs.