

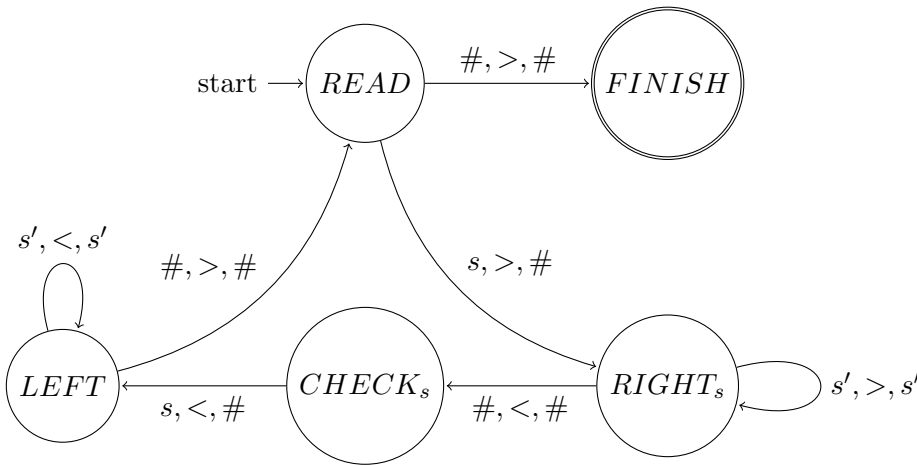
1 Exemples de Machines de Turing

Exercice 1. Soit Σ un alphabet de la bande (telle qu'elle est en entrée) avec $\#$ marquant la fin du mot d'entrée. Donner des Machines de Turing pour les langages suivants :

- $L_1 = \{y \text{ rev}(y) \mid y \in \Sigma^*\}$
- $L_2 = \{yy \mid y \in \Sigma^*\}$
- L^* pour L un langage récursivement énumérable reconnu par une machine de Turing M qui ne lit pas avant le premier caractère et efface le contenu de sa bande avant de terminer.

Solution 1.

1. Pour L_1 , les états sont : $FINISH$, $READ$, $LEFT$, $RIGHT_s$ et $CHECK_s$ pour $s \in \Sigma$. $FINISH$ is the end state, $READ$ the initial. The machine reads a letter s , go to the right compare with the rightmost letter and goes back to the beginning.



2. Pour L_2 , notre solution va insérer des marqueurs D et M . Au début on met les deux à la fin puis tant que D n'est pas au début on décale M de un et D de deux.

Dans la suite on a $R_i \xrightarrow{s, >, s} R_i$ et $L_i \xrightarrow{s, <, s} L_i$. On considère le cas du mot vide à part $Init \xrightarrow{\#, >, \#} OK$ et à deux lettres $Init \xrightarrow{x, >, x} \xrightarrow{x, >, x} \xrightarrow{\#, >, \#} OK$.

On initialise en allant à la fin et $wxy\#$ devient $wDxMy\#$:

$$Init \xrightarrow{s, >, s} R_1 \xrightarrow{x, >, D} I_2(x) \xrightarrow{y, >, x} I_3(y) \xrightarrow{\#, >, M} I_4(y) \xrightarrow{\#, <, y} L_{pousse}$$

ensuite on pousse M une fois et D deux ($w_1xyDw_2zMw_3$ devient $w_1Dxyw_2Mzw_3$) :

$$L_{pousse} \xrightarrow{M, <, z} P_1(z) \xrightarrow{z, <, M} L_2 \xrightarrow{D, <, y} P_2(x, y) \xrightarrow{y, <, x} P_3(x) \xrightarrow{x, <, D} T_{debut}$$

T_{debut} va tester si on est début si c'est le cas on testera que notre mot est de la forme $DyMy\#$ si ce n'est pas le cas il faut retourner à gauche et repousser donc $T_{debut} \xrightarrow{\#, >, \#} Check$ et $T_{debut} \xrightarrow{s, >, s} R_{pousse} \xrightarrow{M, <, z} P_1(z)$.

Enfin pour vérifier on va simplement remplacer chaque lettre s vue à gauche par un marqueur D et à droite par M .

$$Check \xrightarrow{s, >, D} R_{check}(s) \xrightarrow{M, >, M} R_{check2}(s) \xrightarrow{s, <, M} L_{debut} \xrightarrow{D, >, D}, Check$$

$$Enfin \text{ on a une condition échappatoire } Check \xrightarrow{D, >, D} B \xrightarrow{M, >, M} B \xrightarrow{\#, >, \#} OK$$

3. Pour L^* on a tester toutes les coupures $w = x_1x_2 \dots x_n$ et transformer w en $x_1\$x_2 \dots \x_n puis lancer M sur x_n , si ça termine en OK alors lancer M sur x_{n-1} etc.

$$\text{Pour découper la bande } R_{init} \xrightarrow{s, >, s} R_{init} \xrightarrow{s, >, \$} M_s \xrightarrow{u, >, s} M_u \xrightarrow{\#, <, u} Ret \xrightarrow{s, <, s} Ret \xrightarrow{\$, >, \$} R_{init}$$

$$\text{Pour revenir au dernier \$ quand on atteint la fin : } R_{init} \xrightarrow{\#, <, \#} Last\$ \xrightarrow{s \neq \$, <, s} Last\$$$

Pour reconnaître via M : $Last\$ \xrightarrow{\$, >, \$} M_{init}$ et continuer si besoin $M_{OK} \xrightarrow{\#, <, \#} \xrightarrow{\$, <, \#} Last\$$ ou finir si on a tout reconnu $M_{OK} \xrightarrow{\#, <, \#} \xrightarrow{D, >, \#} OK$

Exercice 2. On suppose que les entiers sont écrits en binaire. Les entrées sorties sont faites sur la bande qui contient k nombres. Pour k nombres, la bande est $((0^*1^*)^*\#)^k$ (k fois des 0 et 1 suivis de #). Décrire des machines de Turing pour réaliser les opérations suivantes (à chaque fois la bande de fin doit contenir un nombre, c'est à dire des 0 et des 1 suivis de #) :

1. passer de petit-boutien à grand-boutien (et vice-versa) ;
2. calculer la somme de deux entiers ;
3. étant donnés deux machines M_1 et M_2 qui calculent deux fonctions f_1 et f_2 , donner un machine qui calcule $f_2 \circ f_1$.

Solution 2.

1. Cas du mot vide $init \xrightarrow{\#, >, \#} OK$

Cas général on va à la fin, $init \xrightarrow{s, >, s} init \xrightarrow{\#, <, \$} Push$

Notre tête est à la fin sur la dernière lettre, on va pousser les lettres vers le début, $Push \xrightarrow{u, <, \$} P_u \xrightarrow{s, <, u} P_s \xrightarrow{\#, >, \#} A_s$

A_s va poser s après le premier \$ rencontré (notez qu'on en avait laissé deux notre tête est sur la dernière lettre non retournée) : $A_s \xrightarrow{u, >, u} A_s \xrightarrow{\$, <, s} Push$

Maintenant quand $Push$ lit un \$ c'est qu'on a fini de retourner, il suffit de supprimer ce \$ en poussant les lettres depuis la fin : $Push \xrightarrow{\$, >, \#} End \xrightarrow{s, >, s} End \xrightarrow{\#, <, \#} Push_2 \xrightarrow{s, <, s} P_s^2 \xrightarrow{u, <, s} P_u^2 \xrightarrow{\#, >, u} OK$.

2. On suppose que les nombres sont en petit-boutien de la forme $x_1 + x_2$: on décale pour mettre de la forme $\$x_1 + x_2$ et à chaque fois on va transformer pour ce soit de la forme $w\$w_10^{|w|} + w_2$ avec $|w_i| + |w| = |x_i|$.

$init \xrightarrow{s, >, \$} push_s \xrightarrow{u, >, s} push_u \xrightarrow{\#, <, u} beg \xrightarrow{s, <, s} beg \xrightarrow{\$, >, \$} ha_0$

on est retourné au début on peut faire notre half-adder ha_i pour $i \in \{0, 1\}$:

$ha_i \xrightarrow{s, >, s} ha_{s+i}^2 \xrightarrow{u, >, u} ha_{s+i}^2 \xrightarrow{+, >, 0} ha_{s+i}^3 \xrightarrow{v, <, +} ha_{s+v+i}^4 \xrightarrow{u, <, u} ha_{s+v+i}^4 \xrightarrow{\$, >, (s+v+i) \bmod 2} ha_{(s+v+i) \div 2}$.

La condition d'échappement de ha_i est $ha_i \xrightarrow{\#, >, i} OK$

2 Stabilité des langages énumérables, décidables, etc.

Soient A, B des langages décidables et C et D des langages énumérables.

Exercice 3. Pour chacune des paires (X, Y) parmi (A, B) , (B, C) , (C, D) et (D, A) est-ce que $L_{X, Y}$ est nécessairement décidable ou énumérable pour les langages suivants :

1. $L_{X, Y} = X \cap Y$
2. $L_{X, Y} = X \cup Y$
3. $L_{X, Y} = X \setminus Y$

Solution 3. $X \cap Y$ est nécessairement décidable quand X et Y le sont et énumérables quand X et Y le sont. Pareil pour $X \cup Y$.

$X \setminus Y$ est décidable si X et Y le sont mais n'est nécessairement énumérable que dans le cas Y décidable et X énumérable.

Exercice 4. Montrer que si X et \bar{X} sont énumérables alors X est décidable.

Solution 4. Pour décider de $m \in X$, on énumère simultanément X et \bar{X} et dès que l'on trouve $m \in X$ ou $m \in \bar{X}$ on peut répondre. Comme les deux tests tournent simultanément, la partie X termine forcément quand $m \in X$ et la partie \bar{X} termine sinon.

3 Machines de Turing exotiques

- Une machine à k piles est une machine de Turing avec une bande d'entrée et k bandes de travail, où les bandes de travail sont remplacées par des piles ;
 - une machine à file est une machine de Turing avec une bande d'entrée et une bande de travail, où les bandes de travail sont remplacées par des files : on peut ajouter des éléments part la gauche et les lire par la droite ;
 - une machine à k 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).
1. Montrer qu'une machine de Turing est équivalente à une machine à deux piles.
 2. Montrer qu'une de Turing est équivalente à une machine à une file.
 3. Montrer qu'une machine à une pile peut-être simulée par une machine à deux compteurs.
 4. Montrer qu'une machine de Turing est équivalente à une machine à deux compteurs.

Solution 4.

1. L'idée c'est de stocker l'état de la bande sur deux piles (G, D) . Quand on faire bouger la tête de la MT vers la droite on lit dans D et on écrit dans G et inversement pour aller à gauche on lit dans G et écrit dans D .
2. L'idée c'est qu'on stocke l'état $m_1 \dots m_k m_{k+1} \dots m_n$ sous la forme de la file $m_k \dots m_1 \# m_n \dots m_{k+1}$ quand on est sur l'élément m_k et qu'on veut lire la bande vers la droite. Quand on veut lire vers la gauche on ajoute un deuxième symbole T pour dire qu'il faut faire le tour de la file et on lit les éléments jusqu'à trouver ce T , le dernier élément alors lu correspond à l'élément strictement à gauche.
3. L'idée c'est qu'une pile $p_0 \dots p_n$ dont l'alphabet est $\{B, Z\}$ peut être vu comme un nombre $\sum_{p_k=B} 2^k$. Notre compteur C_1 va donc stocker l'état de la pile de cette façon. Pour lire le dernier élément de la pile il faut diviser C_1 par deux. Pour cela on répète tant que nécessaire prendre deux éléments dans C_1 pour en mettre un dans C_2 . À la fin il nous reste un ou zéro élément dans C_1 , ça nous dit si l'état du haut de la pile est B ou Z .
On vide alors l'éventuel élément restant dans C_1 puis on déplace le contenu de C_2 dans C_1 en répétant tant que possible prendre un dans C_2 et le mettre dans C_1 .
Pour empiler un élément on multiplie C_1 par deux (copier C_1 dans C_2 puis répéter "prendre un dans C_2 et mettre deux dans C_1 ") puis on ajoute un si l'élément à ajouter est B .
4. On va produire une machine à 4 compteurs avec une machine à deux compteurs. Pour cela C_1 va contenir $p_1^{d_1} p_2^{d_2} p_3^{d_3} p_4^{d_4}$ avec $p_1 = 2, p_2 = 3, p_3 = 5, p_4 = 7$. Pour incrémenter un compteur i on multiplie par p_i , pour décrémenter on divise par p_i et pour tester si un compteur est vide on teste le résultat de $C_1 \bmod p_i$. Avec ces 4 compteurs on en déduit deux pour faire une pile et deux autres pour faire une deuxième pile. Une machine avec deux piles pouvant simuler une MT, le résultat est obtenu.