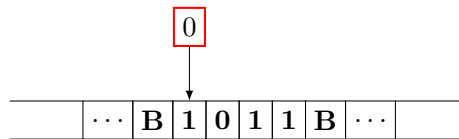


## Objectif et exemple simple.

L'objectif de ce PFE est de bien comprendre la notion de machine de Turing, un modèle mathématique du concept d'algorithme. Un bon manuscrit de ce PFE doit comporter au moins :

1. Un historique sur la machine de Turing à un ou plusieurs rubans.
2. Lien entre la machine de Turing et un algorithme.
3. Quelques notions de complexité des algorithmes.
4. Comprendre la représentation d'une machine de Turing à l'aide d'un diagramme ressemblant à celui que vous avez vu en théorie des graphes.
5. Présentation d'exemples simples de machines de Turing réalisant une tâche donnée.

Une fois la notion de machine de Turing est bien assimilée, vous pouvez regarder l'exemple très simple suivant d'une machine de Turing qui incrémente de 1 un nombre binaire donné. Ce binaire est écrit sur le ruban :



Le binaire **1011** est donné juste à titre d'exemple. Comme vous pouvez le voir, la tête est placée au bit fort du binaire et la machine se trouve à l'état initial 0. Le fonctionnement de cette machine se fait à l'aide de la fonction de transition ou encore à l'aide d'un nombre fini d'instructions chacune ayant la forme suivante :

$$(p, a, b, A, q) \quad (1)$$

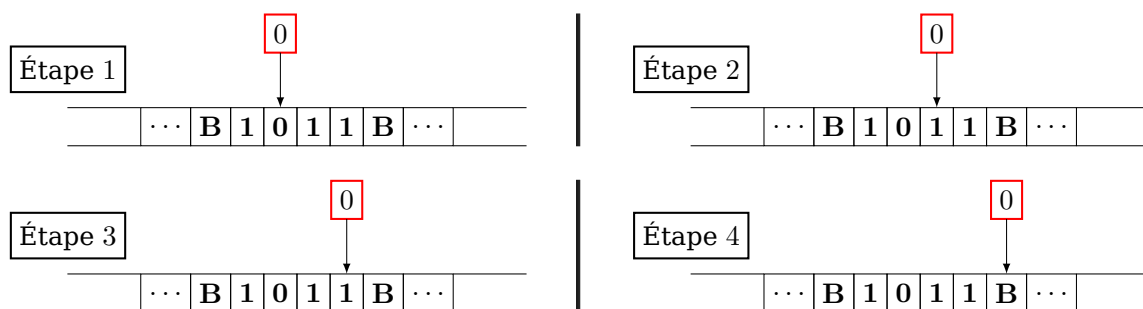
où dans ce cas, on doit comprendre que la machine est à l'état  $p$  et sa tête de lecture pointe sur le caractère  $a$ . L'exécution de l'instruction se fait en effaçant le caractère  $a$  et en le remplaçant par  $b$  puis ensuite en déplaçant la tête de lecture à droite si l'action est  $A = \mathbf{R}$  et à gauche si  $A = \mathbf{L}$  et enfin la machine passe à l'état  $q$ .

Dans notre cas, d'incrémement par 1, on déplace à droite la tête de lecture jusqu'à ce qu'on rencontre un caractère blanc **B** sans changer d'état ni de caractère. Cela peut être réalisé par les instructions :

1 :  $(0, 0, 0, \mathbf{R}, 0)$ .

2 :  $(0, 1, 1, \mathbf{R}, 0)$

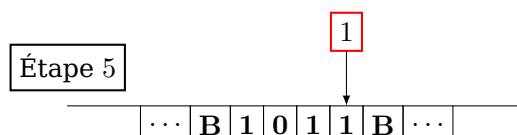
qui conduisent étape par étape aux situations :



On passe de l'étape 4 à l'étape 5 en changeant l'état 0 par 1 et en déplaçant à gauche la tête de lecture pour retrouver le bit faible. Cela se résume en l'instruction :

3 :  $(0, \mathbf{B}, \mathbf{B}, \mathbf{L}, 1)$

qui conduit à la situation :



On traite maintenant notre binaire de droite vers la gauche sachant que l'état actuel de la machine est 1 de la manière suivante : Si on rencontre 1, on le remplace par 0 et on déplace à gauche la tête de lecture sans changer d'état, c'est ce qui est résumé par l'instruction :

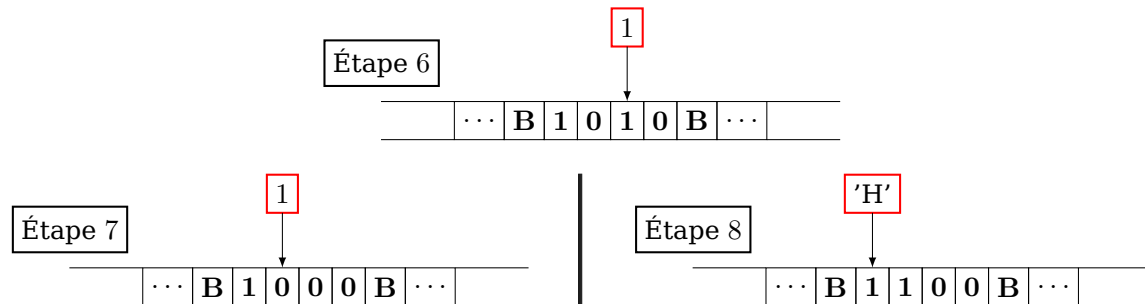
4 : (1, 1, 0, L, 1)

Si on rencontre le caractère 0 ou le caractère B, on le remplace par 1, on déplace à gauche la tête de lecture et la machine passe à l'état final 'H', c'est ce qui est résumé par les instructions :

5 : (1, 0, 1, L, 2)

6 : (1, B, 1, L, 2)

Dans notre cas du binaire 1011, voici les étapes qui restent :



Lorsqu'on arrive à l'état final 'H', les traitements s'arrêtent. Le résultat est le binaire écrit sur le ruban.

## Application

Pour la partie application, vous pouvez simuler une (des) machine(s) de Turing réalisant une (différentes) tâche(s). Dans la Figure 1 ci-dessous, on voit les étapes expliquées plus haut pour l'incréméntation du binaire 1011.

Figure 1 – Machine pour incrémenter un binaire.

L'application doit inclure une base de données de machines de Turing qui réalisent chacune une tâche donnée. L'utilisateur a le choix entre ces tâches. Imaginons qu'une tâche consiste à faire la somme de deux binaires. Une fois cette tâche est choisie par l'utilisateur, ce dernier est invité à entrer ses binaires. L'application

- lui affiche un ruban sur lequel est écrit ses binaires séparés par un caractère spécial,
- lui affiche une tête de lecture et vers quoi elle pointe et l'état initial,
- l'invite à démarrer sa tâche,
- lui montre ce que fait la machine, comme la Figure 1, étape par étape avant d'arriver à la dernière étape où on trouve le ruban avec le résultat de la somme et l'état final.

## Un autre exemple : La somme de deux binaires.

On donne une machine de Turing qui calcule la somme  $b_1 + b_2$  de deux binaires  $b_1$  et  $b_2$ . Au début, on écrit ces deux binaires sur un ruban et on les sépare par un caractère blanc **B**. On met l'état à 0 et on fait pointer la tête sur le caractère gauche de  $b_1$ . Si par exemple,  $b_1 = 1101$  et  $b_2 = 1011$ , alors la situation est

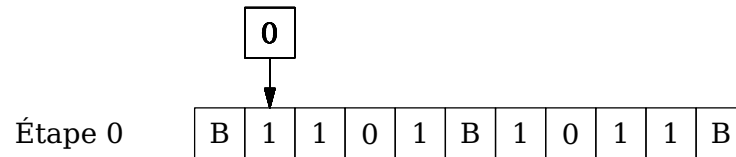


Figure 2 – Situation initiale pour le calcul de 1101+1011

À chaque fois, on décrémente  $b_2$  de 1 et on incrémente  $b_1$  de 1 jusqu'à ce que  $b_2$  devienne nul. Une instruction peut être vue comme une chaîne de caractères  $ecda_f$  et permet de passer d'une étape à la suivante. Cela signifie que  $e$  est l'état courant,  $c$  est le caractère sur lequel pointe la tête et qui sera remplacé par  $d$ ,  $A$  le sens du déplacement et  $f$  l'état suivant. La machine de Turing n'est alors que le regroupement de ces instructions. Dans le cas d'une machine de Turing réalisant la somme de deux binaires, voici les 17 instructions :

000R0, 011R0, 0BBR1, 100R1, 111R1, 1BBL2, 201L2, 210L3, 2BBR5,  
300L3, 311L3, 3BBL4, 401R0, 410L4, 4B1R0, 51BR5, 5BBLH

Dans ce cas

- 0, 1 et B sont les caractères,
- 0, 1, 2, 3, 4, 5 et H sont les états avec H l'état final
- et L, pour gauche, et R, pour droite, sont les sens des déplacements.

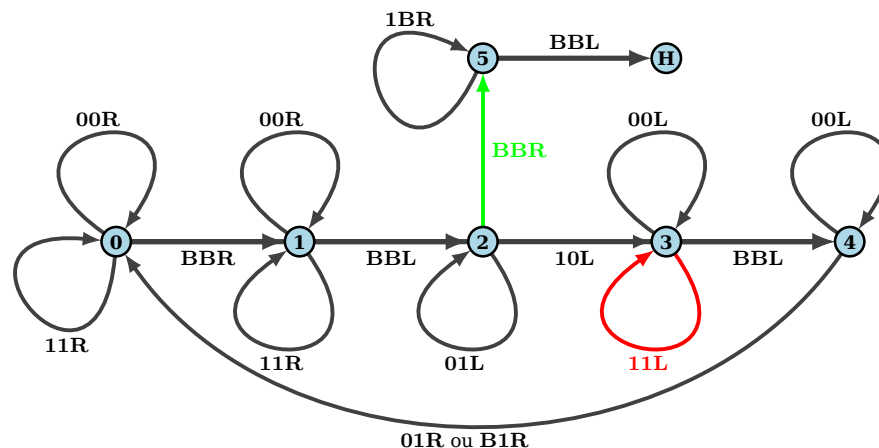


Figure 3 – Diagramme de la machine de Turing réalisant la somme.

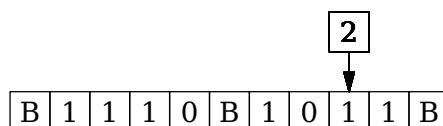
Par exemple, l'instruction **311L3** est représentée en **rouge** dans la Figure 3 et cela signifie que lorsque la machine est à l'état 3 et lit le caractère 1, elle le change en 1, déplace la tête à gauche et passe à l'état 3. De même, l'instruction **2BBR5** est représentée en **vert** dans la Figure 3 et cela signifie que lorsque la machine est à l'état 2 et lit le caractère B, elle le change en B, déplace la tête à droite et passe à l'état 5. Le diagramme la Figure 3 peut être représenté en Python à l'aide de la liste

```
somme_bin = ['000R0', '011R0', '0BBR1', '100R1', '111R1', '1BBL2',
              '201L2', '210L3', '2BBR5', '300L3', '311L3', '3BBL4',
              '401R0', '410L4', '4B1R0', '51BR5', '5BBLH']
```

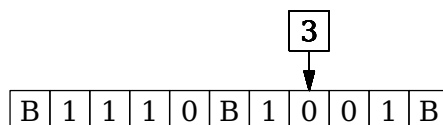
(2)

Dans le schéma de la page 5, vous pouvez voir une illustration de la somme des deux binaires 1101 et 1011. Cette illustration a été réalisée à l'aide d'un programme écrit avec le langage de dessin vectoriel Asymptote dont la syntaxe est très proche de celle de C++. Ce choix est tout simplement pour son intégration avec LATEX utilisé pour la production de ce document. Je vous présente le code écrit dans ce dernier langage dans la page 6 à titre indicatif et comme vous pouvez remarquer la réalisation de ce programme est essentiellement basée sur les points suivants :

1. Savoir la situation initiale, celle de la Figure 2. Un moyen de coder cette situation en Python est `situation0=('B1101B1011B','0',1)`.
2. Savoir comment passer d'une situation donnée à la situation suivante. Si par exemple la situation est



qui peut être codée par `('B1110B1011B','2',8)`, alors le programme doit chercher la seule instruction ou chaîne de caractères dans la liste `somme_bin` définie dans (2) et qui commence par 21, c'est-à-dire `'210L3'=somme_bin[7]`. Dans le codage ci-dessus, le nombre 8 est pour dire que la tête lit le caractère 1 ce qui représente, en Python, le caractère `s[8]`, où `s` est la chaîne de caractères `'B1110B1011B'`. L'instruction choisie par le programme nous permet de passer de l'ancienne situation `('B1110B1011B','2',8)` ci-haut à la nouvelle situation :



qu'on peut coder par `('B1110B1001B','3',7)`.

3. Savoir la situation finale pour l'arrêt du programme. On rappelle que la machine de Turing s'arrête lorsqu'elle arrive à l'état final 'H'. Ainsi, la situation finale doit avoir la forme suivante `('une_chaine_caractères','H',k)`.

Comme vous pouvez le vérifier, la situation finale lorsque le départ est `('B1101B1011B','0',1)` est `situation_f=('11000BBBBBB','H',9)`. Le résultat de la somme des binaires 1101 et 1011 s'obtient à partir de cette situation finale et plus particulièrement à partir de la chaîne de caractères `situation_f[0]`, qu'on peut noter `c` pour simplifier, c'est-à-dire que `c='11000BBBBBB'`. La somme des binaires 1101 et 1011 n'est alors que la chaîne de caractères 11000 qu'on peut obtenir de `c` à l'aide de `c.replace('B','')` selon la syntaxe de Python.

## Une illustration.

L'illustration ci-dessous montre comment obtenir la somme des binaires 1101 et 1011 étape par étape. C'est à ouvrir avec Adobe ou Okular et utiliser les boutons au dessous pour l'animation. Par exemple





- le bouton  pour aller à l'étape suivante,
- le bouton  pour aller à l'étape finale,
- le bouton  pour revenir à l'étape précédente
- le bouton  pour revenir à l'étape initiale.

Figure 4 – Illustration d'une machine de Turing calculant la somme de deux binaires.

## Code en langage Asymptote.

```
import geometry;
import animate;
settings.twice=true;
size(12cm,0);
void situation(string s, string t, int k)
{
for(int i=0;i<length(s);++i)
{
label(substr(s,i,1),(i,0));
label(t,(k,2));
draw((i-0.5,-0.5)--(i-0.5,0.5)--(i+0.5,0.5)--(i+0.5,-0.5)--cycle);
draw((k-0.5,1.5)--(k-0.5,2.5)--(k+0.5,2.5)--(k+0.5,1.5)--cycle);
draw((k,1.5)--(k,0.5),Arrow());
}
}
animation Anim;
string[] somme_bin={'000R0', '011R0', '0BBR1', '100R1', '111R1', '1BBL2', '201L2', '210L3', '2BBR5',
'300L3', '311L3', '3BBL4', '401R0', '410L4', '4B1R0', '51BR5', '5BBLH'};
int p=somme_bin.length, n=0,tete=1;
string etat='0';
string bin1='1101', bin2='1011'; // les binaires à sommer
string donnee="B"+bin1+"B"+bin2+"B"; // les binaires écrits sur le ruban
situation(donnee,etat,tete);
while(etat!='H')
{
situation(donnee,etat,tete);
pair inst=(-3,-7);
pair inst0=(inst.x,inst.y-0.5), inst1=(inst.x,inst.y+0.5), inst2=(inst.x+9,inst.y+0.5), inst3=(inst.x+9,inst.y-0.5);
label(format("L'étape %i",n),(-3,0));
for(int i=0;i<p;++i)
{
label(somme_bin[i],(8,-i-2),E);
}
for(int i=0;i<p;++i)
{
string mot1=substr(somme_bin[i],0,2);
string mot2=etat+substr(donnee,tete,1);
if(mot1==mot2)
{
string d=insert(donnee,tete,substr(somme_bin[i],2,1)); string w=erase(d,1+tete,1); donnee=w;
string u=somme_bin[i],x;
draw((8,-i-2-0.5)--(8,-i-2+0.5)--(10.7,-i-2+0.5)--(10.7,-i-2-0.5)--cycle,linewidth(1));
draw((7,-1.5)--(7,-18.5)--(11.5,-18.5)--(11.5,-1.5)--cycle,linewidth(1));
if(substr(u,3,1)=='R'){tete=tete+1;x='droite';}
if(substr(u,3,1)!='R'){tete=tete-1;x='gauche';}
etat=substr(somme_bin[i],4,1);
string v0="L'état actuel est "+substr(u,0,1)+",", v1="la machine lit le caractère "+substr(u,1,1)+", ",
v2="le change en le caractère "+substr(u,2,1)+", ", v3="la t\^ete se déplace à "+x,
v4="et l'état suivant sera "+substr(u,4,1);
label(v0,(-4.4,-5),E);label(v1,(-4.4,-7),E);label(v2,(-4.4,-9),E);label(v3,(-4.4,-11),E);
label(v4,(-4.4,-13),E);
label("L'instruction qui sera utilisée ",inst+(-1.4,-8),E);
label("est encadrée ci-contre",inst+(-1.4,-9),E);
break;
}
}
n=n+1;
Anim.add(); erase();
}
situation(donnee,etat,tete);
label("L'étape finale",(-3,0));
string dd=replace(donnee,'B','');
string res="La somme de "+bin1+" et "+bin2+" est "+dd+" qu'on voit sur le ruban";
label(res,(0,-2));
Anim.add(); erase(); restore();
label(Anim.pdf("controls",delay=600)); // ici 600 ms
```