Machine de Turing

Un peu d'histoire

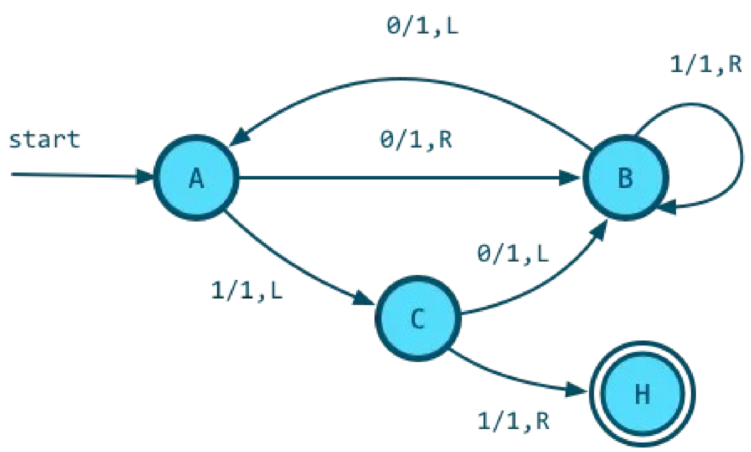
Vous avez peut-être déjà entendu parler d'Alan Turing, à la tête de l'équipe qui a réussi à déchiffrer des messages de la machine Enigma pendant la deuxième guerre mondiale, donnant ainsi un net avantage aux alliés ce qui a nettement réduit la durée du conflit.
Tout juste avant l'arrivée des premiers ordinateurs, ce même Alan Turing a eu l'idée en 1936 d'une machine abstraite (donc pas une vraie que l'on peut toucher, mais juste une description théorique), assez rudimentaire, composée d'un état interne, d'un ruban sur lequel on peut écrire/lire des lettres et de règles (le programme) disant comment faire évoluer cette machine.
A priori rien de bien révolutionnaire, sauf qu'en fait si ! Les fondements que cette machine de Turing a contribué à mettre en place ont pu aboutir quelques années plus tard au concept d'ordinateurs comme vous les connaissez.

Composition de la machine de Turing

La machine de Turing est composée des éléments suivants :

- d'un ruban infini divisé en cases, dans lesquelles la machine peut écrire des symboles d'un alphabet.
- d'une tête de lecture et d'écriture
- d'une table de transition, dont chaque ligne :
  - est associée à un état
  - spécifie les actions à effectuer quand la machine est dans cet état en fonction du symbole lu par la tête de lecture ayant comme actions possibles :
    - Écrire un symbole (choisi dans un alphabet, 0 ou 1 souvent)
    - Se déplacer d'une case sur la gauche ou sur la droite
    - Changer d'état
- S'arrêtant quand on atteint un état désigné comme « terminal »

Une table de transition peut être représentée par un **automate fini déterministe** :



Ici, la table de transition du **castor affairé à trois états** (l'état initial A, le B, le C et l'état terminal H n'étant pas comptabilisé) L'alphabet est compris des caractères 0 et 1.
Chaque flèche est une transition sur laquelle est indiqué :
- le symbole lu sur le ruban,
- le symbole écrit sur le ruban à la place du symbole lu
- la direction vers laquelle on se déplace sur le ruban.
Par exemple, « 0/1, R » indique que le symbole 0 est lu, que 1 est écrit et qu'on se déplace à droite.

Définition formelle

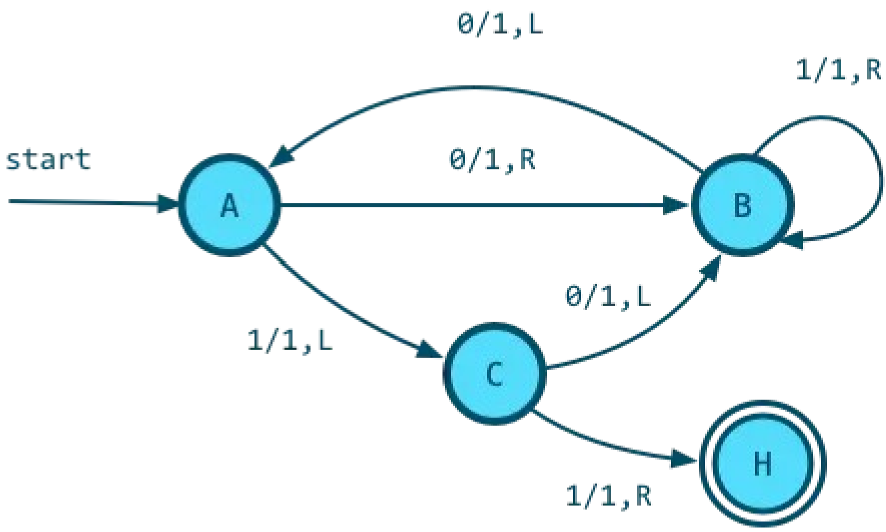
Plusieurs définitions formelles proches les unes des autres peuvent être données d'une machine de Turing. L'une d'elles , relativement courante, est choisie ici. Une machine de Turing est un quintuplet $(Q, \Gamma, q_0, \delta, F)$ où :

- Q est un ensemble fini d'*états* ;
- Γ est l'*alphabet de travail* des symboles de la bande, contenant B un symbole particulier (dit *blanc*), $B \in \Gamma$;
- q_0 est l'*état initial*, $q_0 \in Q$;
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ est la *fonction de transition* ;
- F est l'ensemble des *états acceptants* (ou finals , terminaux), $F \subseteq Q$.

Il s'agit d'un modèle de machine de Turing complète et déterministe ; i.e $\forall q \in Q, \forall \gamma \in \Gamma, \delta(q, \gamma)$ est définie et unique .

<u>Exercice 1 : Exécuter la machine de Turing suivante avec un ruban contenant uniquement des 0.</u>

```
                    A
0   0   0   0   0   0   0   0   0   0   0   0
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
.   .   .   .   .   .   .   .   .   .   .   .
```



<u>Exercice 2 : Représenter l’automate correspondant aux transitions suivantes puis tester la machine sur un ruban contenant uniquement des 0.</u>

Pour une machine à deux états (**A** et **B**), le castor affairé correspond à la table de transition suivante

État	Symbole	
	0	**1**
A	• Écrire le symbole **1** • Déplacer le ruban à **droite** • Passer à l'état **B**	• Écrire le symbole **1** • Déplacer le ruban à **gauche** • Passer à l'état **B**
B	• Écrire le symbole **1** • Déplacer le ruban à **gauche** • Passer à l'état **A**	• Écrire le symbole **1** • Déplacer le ruban à **droite** • Passer à l'état **STOP**

```
                 A
0  0  0  0  0  0  0  0  0  0
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
.  .  .  .  .  .  .  .  .  .
```

<u>Exercice 3 : Dessiner l’automate correspondant aux transitions suivantes puis tester la machine sur un ruban avec des 0.</u>

Pour une machine à quatre états, le castor affairé correspond à la table de transition suivante :

État	Symbole	
	0	**1**
A	• Écrire le symbole **1** • Déplacer le ruban à **droite** • Passer à l'état **B**	• Écrire le symbole **1** • Déplacer le ruban à **gauche** • Passer à l'état **B**
B	• Écrire le symbole **1** • Déplacer le ruban à **gauche** • Passer à l'état **A**	• Écrire le symbole **0** • Déplacer le ruban à **gauche** • Passer à l'état **C**
C	• Écrire le symbole **1** • Déplacer le ruban à **droite** • Passer à l'état **STOP**	• Écrire le symbole **1** • Déplacer le ruban à **gauche** • Passer à l'état **D**
D	• Écrire le symbole **1** • Déplacer le ruban à **droite** • Passer à l'état **D**	• Écrire le symbole **0** • Déplacer le ruban à **droite** • Passer à l'état **A**

Tester cette automate sur
http://turingmachine.vassar.edu/

Combien de 1 apparaissent sur le ruban ?

Castor Affairé

Le castor affairé à n état est la machine de Turing à (n+1) états qui en prenant en entrée un ruban contenant uniquement des 0 doit écrire le maximum de 1 possible sur le ruban avant de s’arrêter.
Vous avez vu dans les question précédentes les castors affairés de 2, 3 et 4 états.

A partir de 6 états, le castor affairé est **incalculable**. Le nombre d’étapes pour que l’algorithme finisse est au moins supérieur à 10 ↑↑ 15 (nous ne sommes même pas sûr du nombre d’étapes) , autrement dit le nombre d'étapes qu'elle effectue avant de s'arrêter est au moins :

$$10\uparrow\uparrow 15=10^{10^{\cdot^{\cdot^{10}}}}$$ avec 15 itérations de puissance de 10.

On a déjà $10\uparrow\uparrow 2=10^{10}=10\,000\,000\,000$ et $10\uparrow\uparrow 3=10^{10^{10}}=10^{10000000000}$, soit bien plus que l’age de l’univers en milliseconde $\approx 4{,}4\times 10^{20}$, soit bien plus que le nombre d’atomes dans l’univers $\approx 10^{80}$…

Conclusion

Avec cette activité le lien avec l’informatique est clair : on exécute et on écrit des programmes, OK.

Mais la question qui reste est de savoir quel est l’intérêt de continuer à utiliser ce modèle rudimentaire inventé au milieu des années 30. Pourquoi s’en souvenir plus de 80 ans plus tard alors que nous avons des ordinateurs très puissants que nous programmons dans des langages dits "évolués", bien plus proche du langage naturel ?
Parce que ce modèle, si rudimentaire soit-il, a la même capacité de calcul que n’importe quel ordinateur.

Oui nos ordinateurs modernes sont bien plus rapides, efficaces, agréables à programmer, mais ce qui est incroyable c’est que pour tout problème qu’un ordinateur sait résoudre, si complexe soit-il, on peut concevoir une machine de Turing (sans doute énorme et peu efficace) qui fait la même chose. Si vous ne voyez toujours pas l’intérêt cela va venir.

Imaginez maintenant que vous trouvez un problème pour lequel vous réussissez à prouver qu’aucune machine de Turing n’est capable de le résoudre. Avec cette preuve et le fait que la machine de Turing sait faire la même chose que les ordinateurs, on obtient automatiquement le résultat suivant : aucun ordinateur, même si dans 100 ans ils sont bien plus puissants, ne pourra résoudre ce problème.

Et faire cette preuve sur les machines de Turing est rendu plus simple car le modèle de ces machines est très précis et les "instructions" ou règles sont d’un type très restreint.

Bonus : application à la recherche textuelle

Représenter une machine de Turing sur un alphabet {0;1} à 4 états qui arrive a un état terminal quand elle à lu un mot 101 sur le ruban.