M1 - T.M.E. Machines de Turing

Le logiciel JFLAP se trouve sous Unix ; si vous le souhaitez, vous pouvez le récupérer à cette adresse : `http://www.cs.duke.edu/csed/jflap/`, où il y a des versions pour Unix, Windows et MacOS X.

Il y a aussi un manuel de référence `http://www.jflap.org/tutorial/`.

Lancer le logiciel **JFLAP** avec la commande

`java -jar /usr/local/jflap/JFLAP.jar`

et choisir **Turing Machine**. Il apparaît alors une fenêtre d'édition, dans laquelle on peut dessiner une Machine de Turing. La barre d'outils située sous l'onglet **Editor** contient six boutons ; de gauche à droite :

- **Attribute Editor** que l'on nommera ici bouton **A**
- **State Creator** (bouton **S**)
- **Transition Creator** (bouton **T**)
- **Deleter** (bouton **D**).
- **Building Block Creator** (bouton **B**)
- **Block Transition Creator** (bouton **BT**).

Un exemple

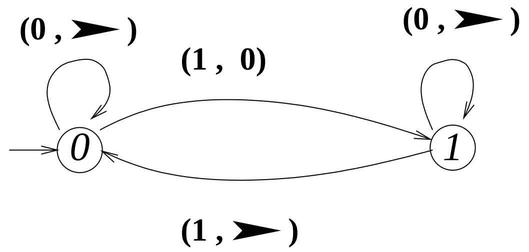

Figure 0.1 Une Machine de Turing à deux états, d'alphabet $\{0, 1, B\}$

Les transitions de la machine de la figure 0.1 sont données par les quintuplets (où $\rightarrow$ indique un déplacement de la tête vers la droite, et $STAY$ pas de mouvement de la tête) :

$$(q_0, 0, 0, \rightarrow, q_0) \quad , \quad (q_1, 0, 0, \rightarrow, q_1) \quad , \quad (q_1, 1, 1, \rightarrow, q_0) \quad , \quad (q_0, 1, 0, STAY, q_1)$$

L'état initial est l'état q_0, noté *0* sur le diagramme. Création de l'automate $\mathcal{A}$

Pour dessiner la machine $\mathcal{A}$:

– créer les états : cliquer sur le bouton **S** ; cliquer en trois endroits différents de la fenêtre d'édition, les états apparaissent avec les noms q_0, q_1, q_2

– choisir la nature des états (initial, final) : cliquer sur le bouton **A** ; faire un clic droit sur l'état q_0 et choisir *Initial.*

– créer les transitions : cliquer sur le bouton **T**

  - transition $(q_0, 0, 0, \rightarrow, q_0)$: cliquer sur l'état q_0 et taper 0 , 0 et choisir R dans les cadres qui apparaissent

T.S.V.P.

- transition $(q_1, 0, 0, \rightarrow, q_1)$: promener la souris, bouton gauche enfoncé, de l'état q_1 à l'état q_1, lâcher le bouton puis taper 0 , 0 et choisir R dans les cadres qui apparaissent
- transition $(q_1, 1, 1, \rightarrow, q_0)$: promener la souris, bouton gauche enfoncé, de l'état q_1 à l'état q_0, lâcher le bouton puis taper 1 , 1 et choisir R dans les cadres qui apparaissent
- autres transitions : sur le même modèle
- attention : on ne peut taper qu'une seule étiquette dans le même cadre READ ou WRITE.

– sauver le fichier en le nommant “exo1-tme”.

Vous découvrirez tout seul les autres utilisations du bouton **A** et l'utilisation du bouton **D**.

0.1 Exercices

Exercice 1 Réaliser la machine de la figure 0.1 en utilisant le logiciel *JFLAP*. Que fait cette machine ? ♢

Exercice 2 Construire une machine de Turing qui, étant donné un nombre n représenté en notation unaire sous la forme $\#\underbrace{11\ldots1}_{n \text{ fois}}*$ fabrique la représentation binaire inversée de n : précisément, votre machine remplacera les uns par des zéros et écrira après le séparateur * l'image miroir de la représentation binaire de n. Par exemple, #1111* deviendra #0000*001 . ♢

Exercice 3 Construire une machine de Turing qui, étant donné un mot $a_1a_2\ldots a_n$ construit son image miroir : précisément, votre machine ira écrire un séparateur $ à la fin du mot, puis elle remplacera les lettres du mot par des blancs et ira écrire les lettres du mot après le séparateur $. Par exemple, *001 deviendra *BBB$100, où B désigne un blanc. ♢

Exercice 4 Combiner les deux machines construites précédemment pour fabriquer une machine de Turing qui calcule la représentation binaire d'un nombre donné en unaire. Vous pourrez utiliser les boutons Building Block Creator et Block Transition Creator. ♢

Exercice 5 Peut-on simuler un automate fini par une machine de Turing ? Construire une machine de Turing qui accepte le langage $a^*b = \{a^nb | n \geq 0\}$. ♢

Exercice 6 On suppose que les entiers sont en représentation unaire sur la bande, i.e. n est représenté par $\ldots BBB\underbrace{11\ldots1}_{n \text{ fois}}BBB\ldots$.

1. Construire une machine de Turing qui réalise la fonction successeur, c'est-à-dire que étant donné la configuration $\underbrace{11\ldots1}_{n \text{ fois}}$ sur la bande, elle met $\underbrace{11\ldots1}_{n+1 \text{ fois}}$ sur la bande puis s'arrête. On pourra boucler sur l'état q_0 jusqu'à ce qu'on arrive au premier bloc de uns, où on passe dans l'état q_1. On remplace ensuite le Blanc suivant ces uns par un 1 et on revient en arrière pour s'arrêter sur le premier 1 dans un état q_2.
2. Avec la même représentation des entiers construire un additioneur : étant donné la configuration $\underbrace{11\ldots1}_{n \text{ fois}}B\underbrace{11\ldots1}_{m \text{ fois}}$ sur la bande, la machine met $\underbrace{11\ldots1}_{n+m \text{ fois}}$ sur la bande puis s'arrête.

3. Comment pourrait-on réaliser un additioneur si les nombres sont en représentation binaire ? ♢

Exercice 7 Construire une machine de Turing qui réalise la fonction $n \longrightarrow 2n$

1. En supposant que les entiers sont en représentation unaire sur la bande.
2. En supposant que les entiers sont en représentation binaire sur la bande. ♢

Exercice 8 Construire une machine de Turing qui accepte les mots de la forme $a^n b^n$. On pourra remplacer le premier a par X, puis remplacer le premier b par Y, puis retourner à gauche pour remplacer le second a par X, puis remplacer le second b par Y, et ainsi de suite jusqu'à ce qu'il ne reste plus de a dans le début du mot. On vérifie ensuite qu'il n'y a que des Y dans la fin du mot pour accepter. ♢

Exercice 9 Soit $A = \{(,)\}$ l'alphabet constitué de deux parenthèses (ouvrante et fermante). L'ensemble $D \subseteq A^*$ des parenthésages bien formés, appelé langage de Dyck, est défini par

(B) $\varepsilon \in D$,

(I) si x et y sont dans D, alors (x) et xy sont aussi dans D.

Construire une machine de Turing qui accepte les mots du langage de Dyck. On suppose les mots donnés sous la forme $*w*$, et la machine devra commencer avec sa tête de lecture positionnée sur la première lettre de w. Elle avance jusqu'à trouver la première parenthèse fermante, qu'elle efface, puis retourne en arrière pour aller effacer la parenthèse ouvrante correspondante, et recommence tout ce processus. ♢