M1 - T.D. Logique Décidabilité et machines de Turing

Les machines de Turing de *Turing's world* ont des transitions d'un type plus restreint que celles des machines vues en cours ; la machine peut : soit changer le symbole écrit sur la bande mais alors la tête ne bouge pas, soit bouger sa tête, mais alors le symbole écrit sur la bande ne change pas ; on peut donc écrire les transitions de ces machines sous forme de quadruplets, par exemple pour la figure 0.1,

EXERCICE 1 Montrer que les machines de Turing restreintes de *Turing's world* peuvent simuler les machines décrites en cours (il suffit de rajouter des états). ♢

EXERCICE 2 Que fait la machine de Turing de la figure 0.1 ? Réaliser cette Machine avec le logiciel *Turing's World* (à faire en TME). ♢

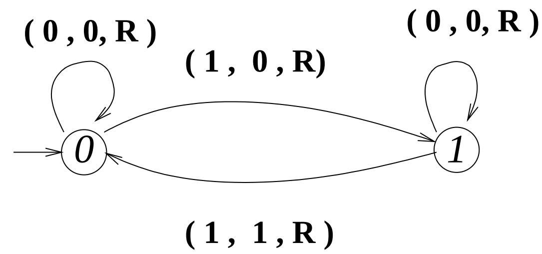

Figure 0.1 Une Machine de Turing à deux états, d'alphabet $\{0, 1, B\}$

EXERCICE 3 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 4 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 5 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 unaire sur la bande. ♢

EXERCICE 6 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

T.S.V.P.

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 7   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.

Comparez (en TME) votre machine avec la machine *ParenCheck* de *Turing's World*. ♢

Exercice 8   Montrer que les fonctions suivantes sont primitives récursives :
1) la fonction $f(n) = n \bmod 2$ (le reste de la division de n par 2).
2) la fonction $f(n) = \lfloor n/2 \rfloor$ (le quotient de la division de n par 2). ♢

Exercice 9   Montrer que les fonctions suivantes sont primitives récursives :
1) les fonctions constantes.
2) la multiplication, définie par

$$x.0 = 0$$
$$x.(y+1) = x.y + x$$

2) l'exponentiation, définie par

$$x^0 = 1$$
$$x^{(y+1)} = x^y.x$$

♢

Exercice 10   Un prédicat (ou un sous-ensemble) $P \subseteq \mathbb{N}^k$ est dit (primitif) récursif ssi sa fonction caractéristique est (primitive) récursive. Soit A un sous-ensemble (primitif) récursif de $\mathbb{N}^k$, et soient f et g deux fonctions (primitives) récursives. Montrer que la fonction h définie par

$$h(x_1, \dots, s_k) = \begin{cases} f(x_1, \dots, s_k) & \text{si } (x_1, \dots, s_k) \in A \\ g(x_1, \dots, s_k) & \text{sinon} \end{cases}$$

est (primitive) récursive. ♢

Exercice 11   Trouver l'erreur dans le raisonnement suivant : pour décider si deux programmes calculent le même résultat pour une valeur donnée X, on lance les deux programmes et on on compare les résultats retournés. Peut-on décider si deux programmes calculent le même résultat pour une valeur donnée X ? ♢

Exercice 12   1) Montrer que les sous-ensembles récursifs de $\mathbb{N}^k$ sont clos par union, intersection, complémentation.

2) Montrer que les sous-ensembles récursivement énumérables de $\mathbb{N}^k$ sont clos par union, intersection. Que pensez-vous de la complémentation ? ♢