Feuille d'exercices n°2

Machines à registres, machines de Turing

Machines à registres

Exercice 1.

1. Décrire des machines à registres qui calculent les fonctions sg et $\overline{\text{sg}}$.
2. Décrire une machine dont le calcul termine en 0 sur 0, et qui ne termine pas pour tout autre entier.

Exercice 2.

1. Simuler directement l'instruction for par un programme goto.
2. Simuler directement l'instruction

   $$\text{repeat } S \text{ until } R_i = 0$$

   par un programme goto.

Exercice 3. Démontrer que toute fonction calculable par programme for est récursive primitive.

Procéder par induction sur le niveau d'imbrication des boucles for. Montrer que le contenu de chaque registre de la machine est une fonction récursive primitive des entrées en utilisant le schéma de récurrences mutuelles.

Machines de Turing Une autre notion de machine, assez différente des machines à registres, est due à Turing en 1936. Les machines de Turing ont une gestion de la mémoire de plus bas niveau : on va lire les entiers chiffre par chiffre alors que pour les machines à registres les instructions d'incrémentation et de décrémentation ont été considérées comme élémentaires. De ce fait la représentation des entiers a une importance pour les machines de Turing.

De plus, alors que pour les machines à registres l'accès à un registre est considéré comme en temps constant, pour une machine de Turing, il faut éventuellement parcourir un ruban pour accéder à la partie de la mémoire correspondante, et parcourir la zone mémoire correspondante pour lire l'entier sur une machine de Turing, c'est à dire que l'accès à la mémoire est séquentiel.

IL existe plusieurs variantes de machines de Turing. Dans la suite, on appellera machine de Turing un dispositif constitué de plusieurs rubans, chacun muni d'une tête de lecture/écriture et d'un ensemble d'états. Un ruban est une suite de cases infinie à droite (les cases d'un ruban sont numérotées par des entiers naturels) [1]. la tête se déplace sur le ruban : quand elle est à une position donnée, elle peut modifier ce contenu, puis se déplacer à droite d'une case, à gauche d'une case ou rester immobile. Le fonctionnement de la machine est décrit par une suite de *transitions* : chaque transition décrit, en fonction de l'état initial et du contenu des cases à la position des têtes, un changement d'état et d'éventuelles écritures et déplacements de chacune des têtes.

Plus formellement, une machine de Turing M à k rubans ($k \in \mathbb{N}^*$) est définie comme $M = (\Gamma, \Sigma, Q, q_0, q_f, \delta)$, où Γ l'alphabet de travail contient un symbole spécial de début de ruban $\sharp$, Σ l'alphabet d'entrée est un sous-ensemble de $\Gamma \setminus \{\sharp\}$, Q est l'ensemble des états, q_0 et q_a sont deux états particuliers (éléments de Q) appelés état initial et état acceptant, et δ appelée fonction de transition est une fonction partielle définie sur :

$$\delta : Q \times \Gamma^k \longrightarrow Q \times \Gamma^k \times \{\lhd, \rhd, -\}^k,$$

le troisième élément de la sortie indique le sens de déplacement de la tête.

On suppose de plus que la fonction δ n'est pas définie pour l'état q_a, vérifie que la machine ne modifie jamais la case initiale ($\sharp$), et qu'une tête ne se déplace pas à gauche en début de ruban (lecture de $\sharp$). La fonction de transition est décrite par un ensemble fini de règles de transitions. On peut ajouter un état refusant q_r : pour représenter un prédicat, pas besoin de bande de sortie, un état acceptant et un état refusant suffisent ; alors la fonction de transition n'est pas non plus définie pour l'état refusant.

On suppose dans la suite que la machine a au moins 2 rubans, du moins quand elle calcule une fonction : le premier ruban est le ruban d'entrée et il n'est pas modifié par la machine (aucune écriture), le dernier ruban est le ruban de sortie et il n'est pas lu par la machine (une transition ne dépend pas de la

1. Les machines de Turing originales ont des rubans bi-infinis

lecture sur cette bande). Les autres rubans sont les rubans de travail. Cette contrainte n'est pas indispensable mais elle sera utile pour parler de complexité en espace (l'espace de travail est indépendant des espaces de lecture et écriture).

L'alphabet de travail contient un symbole particulier, le blanc noté $\sqcup$ qui n'appartient pas à l'alphabet d'entrée et permet de séparer les arguments sur le ruban d'entrée pour une fonction à plusieurs arguments.

Pour calculer les fonctions sur les entiers, on va les représenter en binaire,se contenter de l'alphabet d'entrée $\Sigma = \{0,1\}$, et de l'aphabet de travail $\Gamma = \Sigma \cup \{\sqcup\}$.

À l'initialisation tous les rubans de la machine sont supposés contenir le symbole $\sharp$ en début de ruban. Les rubans de travail et le ruban de sortie ne contiennent sinon que des blancs. Les têtes sont toutes en début de ruban.

Une fonction partielle de $\mathbb{N}^k \to \mathbb{N}$ est calculable par une machine de Turing, signifie qu'à chaque fois que la machine est initialisée comme décrit ci-dessus, avec de plus les entiers $x_1, \ldots, x_k$ écrits en binaire séparés par des blancs sur le ruban d'entrée à partir de la position initiale (après $\sharp$) :

$$\sharp x_1 \sqcup x_2 \sqcup \ldots \sqcup x_n \sqcup \sqcup \ldots$$

alors si $f(x_1, \ldots, x_k) \downarrow$ le calcul de la machine termine dans l'état final avec $f(x_1, \ldots, x_k)$ écrit sur la bande de sortie (à partir de $\sharp$), sinon le calcul ne termine pas ou pas dans l'état final.

Exercice 4. Définir explicitement une machine de Turing qui n'utilise pas de ruban de travail :

1. qui teste la parité d'un entier écrit en binaire ;
2. qui teste si un entier est nul ;
3. qui décrémente de 1 un entier ;
4. qui incrémente de 1 un entier (indiquer comment modifier la précédente).

Exercice 5. Indiquer comment construire une machine de Turing qui recopie la seconde entrée sur une bande de travail puis calcule l'addition de deux entiers en binaire par l'algorithme du primaire.

Le but des exercices qui suivent est essentiellement de montrer qu'il est possible de simuler le fonctionnement d'une machine à registres par une machine de Turing et réciproquement.

Exercice 6. Définir une machine de Turing :

1. qui recopie le ruban d'entrée sur le premier ruban de travail, ne modifie rien d'autre et repositionne les têtes de lecture en position initiale ;
2. qui prend k arguments $x_1, \ldots, x_k$ en entrée et renvoie $x_i - 1$ pour $1 \leq i \leq k$;
3. qui prend k arguments $x_1, \ldots, x_k$ en entrée et renvoie $x_i + 1$ pour $1 \leq i \leq k$;
4. qui prend k arguments $x_1, \ldots, x_k$ en entrée et teste si $x_i = 0$ pour $1 \leq i \leq k$.

Exercice 7. Indiquer comment définir une machine de Turing $Init_{k,n}$, $k \leq n$ qui prend en entrée k entiers $x_1, \ldots, x_k$ et recopie sur le premier ruban de travail la suite de ces k entiers en binaire séparés par des blancs, suivie éventuellement de $n-k$ l'entier fois 0 séparés par des blancs.

Exercice 8. Indiquer comment simuler une machine à n registres calculant une fonction partielle f : $\mathbb{N}^k \to \mathbb{N}$ ($k \leq n$) par une machine de Turing, c'est-à-dire que la machine de Turing calcule la même fonction partielle f.

On appelle *configuration* d'une machine de Turing une description exhaustive d'une machine à un instant donné du calcul. La configuration d'une machine de Turing à k rubans (telle que définie dans cette feuille) est décrite par :

— l'état, élément de Q ;
— le contenu des k rubans, soit k suites finies d'éléments de Γ ;
— la position des k têtes soit k entiers.

On peut donc définir une configuration d'une machine à k rubans comme un élément de $Q \times (\Gamma^{<\omega})^k \times \mathbb{N}^k$.

Exercice 9. Proposer un codage des configurations, et indiquer comment utiliser ce codage pour montrer que toutes les fonctions calculables par machine de Turing sont récursives partielles, et donc calculables par machines à registres. On peut se restreindre aux machines à 3 bandes (il est possible de montrer par ailleurs que toute fonction calculable sur une machine de Turing à $k+2$ rubans (k bandes de travail) est calculable sur une machine à une seule bande de travail, en codant sur celle-ci le travail sur les k bandes.