Calculabilité/Complexité – 13/11/2020 – durée 1h

Lire les questions ATTENTIVEMENT. Il est possible d'admettre des réponses pour ne pas rester bloqué dans un problème. Il n'est pas nécessaire de tout finir pour avoir 20 (mais il faut toucher à tout). Une réponse à une partie ne peut dépasser une page A4 manuscrite. Chaque partie doit être déposée indépendamment dans la case Tomuss correspondante.

1 Fonctions récursives

On veillera à bien se placer dans le cas des définitions de la page 4 notamment au niveau des paramètres.

On a vu en cours (slide 78 cours 4) qu'il était possible de construire et déconstruire des couples (et donc des n-uplets) à l'aide de fonctions récursives primitives. Ces couples sont codés par des entiers (normal : on travaille avec des fonctions numériques).

On se propose de travailler sur des listes vues comme couples : (élément, liste). On suppose donc connues les opérations récursives primitives suivantes :

- Le constructeur de liste vide, noté $\langle\rangle$, qu'on considérera comme une fonction constante.
- La fonction de test de vacuité `vide?` qui retourne 0 si et seulement si son paramètre est la liste vide.
- Le constructeur de paires $\langle , \rangle$ (qu'on pourra appeler `cons`) qui à partir de deux entiers x et y construit (l'entier z noté) $\langle x, y\rangle$ codant la paire (x, y).
- Les deux projections associées : t et r telles que $t(\langle x, y\rangle) = x$, la *tête* et $r(\langle x, y\rangle) = y$, le *reste*.

Pour simplifier on supposera qu'on n'est jamais dans un cas où on essaye d'appliquer t ou r à la liste vide.

On peut dès lors représenter facilement une liste l non vide comme une paire $\langle a, l'\rangle$ où a est le premier élément (élément de rang 0) et l' est une liste.

Exemple. La liste $[3; 5; 2]$ sera le résultat de $\texttt{cons}(3, \texttt{cons}(5, \texttt{cons}(2, \langle\rangle)))$, noté de façon équivalente $\langle 3, \langle 5, \langle 2, \langle\rangle\rangle\rangle\rangle$. La tête de cette liste peut être obtenue par $t(\langle 3, \langle 5, \langle 2, \langle\rangle\rangle\rangle\rangle)$ qui vaut donc 3. Le reste de cette liste peut être obtenu par $r(\langle 3, \langle 5, \langle 2, \langle\rangle\rangle\rangle\rangle)$ qui vaut donc $\langle 5, \langle 2, \langle\rangle\rangle\rangle$.

Question 1. Montrer que la fonction `nthrm` qui prend un entier k et une liste l et extrait le k-ième reste de la liste est récursive **primitive**. On considérera que k est toujours inférieur à la longueur de la liste.
Exemple : $\texttt{nthrm}(2,\ \langle 3, \langle 5, \langle 4, \langle\rangle\rangle\rangle\rangle) = \langle 4, \langle\rangle\rangle$.
Exemple : $\texttt{nthrm}(0,\ \langle 3, \langle 5, \langle 4, \langle\rangle\rangle\rangle\rangle) = \langle 3, \langle 5, \langle 4, \langle\rangle\rangle\rangle\rangle$.

Question 2. Éventuellement à l'aide de la question précédente, montrez que la fonction `nth` qui prend un entier k et une liste l et extrait l'élément de rang k de l est récursive **primitive**.
Exemple : $\texttt{nth}(1,\ \langle 3, \langle 5, \langle 4, \langle\rangle\rangle\rangle\rangle) = 5$.

Question 3. Éventuellement à l'aide de questions précédentes, montrez que la fonction `revconcat` qui prend un entier k, une liste l_1 de longueur supérieure à k, une liste l_2 et retourne la liste constituée des k premiers éléments de l_1 dans l'ordre inverse mis en tête de l_2 est récursive **primitive**.
Exemple : $\texttt{revconcat}(2,\ \langle 1, \langle 2, \langle 3, \langle 4, \langle\rangle\rangle\rangle\rangle\rangle,\ \langle 5, \langle 6, \langle 7, \langle 8, \langle\rangle\rangle\rangle\rangle\rangle) = \langle 2, \langle 1, \langle 5, \langle 6, \langle 7, \langle 8, \langle\rangle\rangle\rangle\rangle\rangle\rangle\rangle$.
Exemple : $\texttt{revconcat}(0,\ \langle 1, \langle 2, \langle 3, \langle 4, \langle\rangle\rangle\rangle\rangle\rangle,\ \langle 5, \langle 6, \langle 7, \langle 8, \langle\rangle\rangle\rangle\rangle\rangle) = \langle 5, \langle 6, \langle 7, \langle 8, \langle\rangle\rangle\rangle\rangle\rangle$.

Question 4. Éventuellement à l'aide de questions précédentes, montrer que la fonction `length` qui prend une liste l et retourne un entier représentant la longueur de l est **récursive**.

2 Machines de Turing

Question 5. Proposez une machine de Turing simple reconnaissant le langage des mots sur $\{a, b\}$ possédant aa en leur milieu (ces mots sont donc en particulier de longueur paire).

3 Machines à deux piles

Question 6. Soit $V = \{a, b\}$ un alphabet. Soit Q un ensemble d'états. Une machine à deux piles est un dispositif à états comportant deux piles p_1 et p_2 et muni des opérations suivantes :

- $push_1$ qui sur la donnée d'un caractère de V empile ce caractère sur la pile p_1,
- $push_2$ qui sur la donnée d'un caractère de V empile ce caractère sur la pile p_2,
- $vide_1$ et $vide_2$ les tests de vacuité des piles.
- pop_1 qui dépile et retourne le caractère au sommet de la pile p_1, la pile p_1 est donc modifiée (elle a perdu un étage),
- pop_2 qui dépile et retourne le caractère au sommet de la pile p_2, la pile p_2 est donc modifiée (elle a perdu un étage),
- Une fonction de transition qui à
  - la valeur des caractères aux sommets des piles et
  - l'état courant

  associe
  - un nouvel état et
  - une séquence de $push_n$ et pop_n.

Montrez comment représenter une machine de Turing avec une machine à deux piles. En particulier :

1. Comment représenter une configuration c d'une machine de Turing (contenu de la bande et état) par une configuration c^p de la machine à deux piles (contenu des piles et état),
2. Comment représenter la fonction de transition c'est-à-dire comment définir, pour les différents cas de transition d'une configuration c_1 à une configuration c_2 de la machine de Turing, les opérations de transition pour la machine à deux piles la faisant passer de c_1^p à c_2^p.

Question 7. L'arrêt des machines à deux piles est-il décidable ? Justifiez votre réponse.

Constantes

$$C_{k,c} \in \mathcal{F}_k, \quad C_{k,c}(x_1, \ldots, x_k) = c$$

Successeur

$$S \in \mathcal{F}_1, S(x) = x + 1$$

Projections

$$\pi_{k,i} \in \mathcal{F}_k, \quad \pi_{k,i}(x_1, \ldots, x_i, \ldots, x_k) = x_i$$

Schéma de *Composition* :
— f à n arguments
— $g_1, \ldots, g_n$ à m arguments
$\rightsquigarrow$ h à m arguments

$$h(x_1, \ldots, x_m) = f(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$$

$h = \mathbf{Comp}_{n,m}(f, g_1, \ldots, g_n)$

Schéma de *récursion primitive* :
— b à n arguments
— h à $n+2$ arguments
$\rightsquigarrow$ f à $n+1$ arguments

$$\begin{aligned} f(0, x_1, \ldots, x_n) &= b(x_1, \ldots, x_n) \\ f(k+1, x_1, \ldots, x_n) &= h(k, x_1, \ldots, x_n, \underbrace{f(k, x_1, \ldots, x_n)}) \end{aligned}$$

$f = \mathbf{Rec}(b, h)$

Schéma de *minimisation* :
— g à $n+1$ arguments
$\rightsquigarrow$ f à n arguments

$$f(x_1, \ldots, x_n) = \min\{k \mid g(k, x_1, \ldots, x_n) = 0\}$$

$f = \mathbf{Min}(g)$