DM8

Dans cet exercice, on manipule des fonctions écrites dans un langage de programmation, par exemple OCaml. On distingue la notation f, correspondant à la fonction elle-même, ou l'algorithme, et la notation $\langle f\rangle$, désignant le code source de la fonction f. Ainsi, si on considère le code :

```
let rec f lst = match lst with
  | [] -> 0
  | _ :: q -> 1 + f q
```

alors on distingue l'objet f, de signature `' a list -> int`, et l'objet $\langle f\rangle$, de signature `string`, correspondant à la chaîne de caractères :

```
"let rec f lst = match lst with | [] -> 0 | _ :: q -> 1 + f q "
```

Pour simplifier l'étude, on suppose que l'ensemble des fonctions manipulées sont de signature `string -> bool`, sauf une fonction `universel : string -> string -> bool` telle que l'appel à `universel <f> x` simule l'exécution de `f x`. On note Σ^* l'ensemble des chaînes de caractères.

Si f est une fonction, on note $L(f)$, appelé *langage de* f, l'ensemble des arguments pour lesquels la fonction renvoie `true`, c'est-à-dire :

$$L(f) = \{x \in \Sigma^* \mid f(x) \text{ termine et renvoie } \texttt{true}\}$$

Question 1. Montrer que le problème Appartient suivant est indécidable et semi-décidable.

- *Instance* : un code source $\langle f\rangle \in \Sigma^*$ et un argument $x \in \Sigma^*$.
- *Question* : est-ce que $x \in L(f)$?

Question 2. Montrer que le problème Diagonal suivant n'est pas semi-décidable.

- *Instance* : un code source $\langle f\rangle \in \Sigma^*$.
- *Question* : est-ce que $\langle f\rangle \notin L(f)$?

Indication : on pourra supposer qu'il est semi-décidable, résolu partiellement par un algorithme A, et s'intéresser à $\langle A\rangle$ et $L(A)$.

Question 3. Montrer que Diagonal $\leqslant$ *coAppartient*. Que peut-on en déduire sur les problèmes complémentaires de Diagonal et Appartient ?

Question 4. On appelle *propriété des langages de fonctions* tout sous-ensemble de $\mathcal{P}(\Sigma^*)$. Si $P \subseteq \mathcal{P}(\Sigma^*)$, on assimile P et le problème de décision suivant, qu'on notera également P.

- *Instance* : un code source $\langle f\rangle \in \Sigma^*$.
- *Question* : est-ce que $L(f) \in P$?

Est-ce que *Diagonal* est une propriété des langages de fonctions ? Justifier.

Question 5. On rappelle qu'un langage est un sous-ensemble de Σ^*. Si $L \subseteq \Sigma^*$, on assimile L et le problème de décision suivant, noté également L.

- *Instance* : une chaîne $x \in \Sigma^*$.
- *Question* : est-ce que $x \in L$?

Dans la suite, on cherche à montrer le *théorème de Rice.*

Soit P une propriété non triviale des langages semi-décidables. Alors P n'est pas décidable.

Propriété non triviale des langages semi-décidables signifie qu'il existe deux langages semi-décidables L_1 et L_2 tels que $L_1 \in P$ et $L_2 \notin P$ (ce n'est ni une propriété vérifiée par aucun langage semi-décidable, ni par tous). On pose P une telle propriété non triviale des langages semi-décidables. Comme P est non triviale, soit $L \in P$ semi-décidable. Comme L est semi-décidable, soit f_L un algorithme qui résout partiellement L.
Justifier qu'on peut supposer, sans perte de généralité, que $\varnothing \notin P$. On fera cette hypothèse pour la suite.

Question 6. Soient $\langle f\rangle \in \Sigma^*$ et $x \in \Sigma^*$. On définit la fonction suivante :

```
let g y =
  universel <f> x && universel <f_L> y
```

En considérant la fonction `g`, montrer que Appartient $\leqslant P$.

Question 7. En déduire le théorème de Rice.

Question 8. Montrer que le problème suivant est indécidable.

- *Instance* : un code source $\langle f \rangle \in \Sigma^*$.
- *Question* : est-ce que $L(f) \neq \emptyset$?

Question 9. On considère le problème suivant.

- *Instance* : un code source $\langle f \rangle \in \Sigma^*$.
- *Question* : est-ce que le code source de f contient au moins 5 boucles `while` ?

□ **9.1.** Ce problème est-il décidable ?
□ **9.2.** Cela résultat contredit-il le théorème de Rice ?