Calculabilité & Complexité [M1]

Hugo Férée, Valia Mitsou

Feuille de TD n°2
Problèmes indécidables

Exercice 1 *À partir du problème de l'arrêt, construire un problème qui n'est ni semi-décidable, ni co-semi-décidable.*

Exercice 2

1. *À quoi correspond l'équivalent d'une machine universelle dans votre langage de programmation préféré ?*
2. *Quelle est la spécification d'une machine universelle en terme d'énumération des fonctions récursives ?*

Exercice 3 *Montrer que déterminer si un programme reconnaît le langage vide est un problème indécidable.*

Exercice 4 (Castor affairé (*busy beaver*)) *On définit les deux fonctions suivantes :*

— $BB(n)$ *est le plus grand nombre de transitions qu'une machine à* n *peut effectuer sur l'entrée vide*[1].

— $BB_{score}(n)$ *est la taille du plus grand mot que peut calculer une machine à* n *états sur l'entrée vide.*

1. *Montrer que ces deux fonctions sont bien définies.*
2. *Montrer que* BB *n'est pas calculable en se ramenant au problème de l'arrêt.*
3. *Montrer que* BB_{score} *n'est pas calculable.*
   Indice : commencer par écrire la phrase en français de moins de 100 caractères qui décrive le plus grand nombre possible.

Remarque *Tibor Radó définit busy beaver et détermine* $BB(1) = 1$ *et* $BB(2) = 6$ *en 1962.* $BB(3) = 21$ *est déterminé en 1960, puis* $BB(4) = 107$ *est prouvé par Brady en 1983. La preuve (formelle, en Coq !) que* $BB(5) = 47176870$ *a été très récemment (mai 2024) terminée par un groupe d'amateurs (voir* `https://bbchallenge.org` *).*

Exercice 5 *Monter que l'équivalence de machines de Turing n'est ni r.e. ni co-r.e. On pourra par exemple montrer par l'absurde que si c'était le cas, on pourrait décider le problème de l'arrêt.*

Exercice 6 (Code mort (**)) *Monter (au choix) que :*

— *savoir si une machine a des états inutiles/inaccessibles est un problème indécidable.*

— *savoir si un programme* `C` *contient du code mort est un problème indécidable.*

1. si elle s'arrête sur l'entrée vide