Calculabilité

TP4

Y. Deville
C-H. Bertrand Van Ouytsel - V. Coppé - A. Gerniers
N. Golenvaux - M. Parmentier

Mars 2021

Questions du test

En vous aidant du théorème de Rice, déterminer si l'ensemble défini ci-dessous est récursif.

1. L'ensemble des programmes qui calculent la fonction $f : \mathbb{N} \to \mathbb{N}$ telle que $f(n) = 2n$.

   Réponse : Non-récursif

   - Soit A, l'ensemble des programmes i qui calculent $\varphi_i = f(n) = 2n$. De tels programmes existent, donc $A \neq \emptyset$.
   - Alors tous les programmes j dans $\overline{A}$ calculent $\varphi_j \neq \varphi_i$. Il existe bien des programmes qui ne calculent pas $f(n)$, donc $A \neq \mathbb{N}$.
   - Par la contraposée du théorème de Rice, A n'est pas récursif.

Questions du test

En vous aidant du théorème de Rice, déterminer si l'ensemble défini ci-dessous est récursif.

2. L'ensemble des programmes qui renvoient 0 pour au moins une entrée.

 Réponse : Non-récursif

 - Soit $A = \{i \mid \exists x : \varphi_i(x) = 0\}$, $A \neq \emptyset$ (e.g. $P_i(x) \equiv$ `return 0`) et $A \neq \mathbb{N}$ (e.g. $P_j(x) \equiv$ `return 42`)
 - $\forall i \in A, \exists x : \varphi_i(x) = 0$ et $\forall j \in \overline{A}, \forall x : \varphi_j(x) \neq 0$

 $\Longrightarrow \forall i \in A, \forall j \in \overline{A}, \exists x : \varphi_i(x) \neq \varphi_j(x)$

 $\Longrightarrow \forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$
 - Par la contraposée du théorème de Rice, A n'est pas récursif.

Questions du test

En vous aidant du théorème de Rice, déterminer si l'ensemble défini ci-dessous est récursif.

3. L'ensemble des programmes qui terminent après moins de 1000 instructions pour l'entrée 0.

   Réponse : Récursif

   On peut décider A en exécutant les 999 premières instructions de $P(0)$. Vérifions que Rice s'applique :

   Comme $A \neq \emptyset$ et $A \neq \mathbb{N}$, par le théorème de Rice, $\exists i \in A$ et $\exists j \in \overline{A}$ tel que $\varphi_i = \varphi_j$.
   Par exemple :

   $P_i \equiv$ `return 42`

   $P_j \equiv$
   ```
   for k in range(1000):
     sleep(1)
   return 42
   ```

Questions du test

En vous aidant du théorème de Rice, déterminer si l'ensemble défini ci-dessous est récursif.

4. L'ensemble des programmes qui terminent pour au moins deux entrées différentes.

   Réponse : Non-récursif

   - Soit $A = \{i \mid \exists x, y : x \neq y \wedge \varphi_i(x) \neq \bot \wedge \varphi_i(y) \neq \bot\}$, $A \neq \emptyset$ (e.g. $P_i(x) \equiv$ `return 0`) et $A \neq \mathbb{N}$ (e.g. $P_j(x) \equiv$ `while True: pass`)
   - $\forall i \in A, \exists x, y : x \neq y \wedge \varphi_i(x) \neq \bot \wedge \varphi_i(y) \neq \bot$
     $\forall j \in \overline{A}, \forall x, y : x \neq y \to (\varphi_j(x) = \bot \vee \varphi_j(y) = \bot)$

     $\implies \forall i \in A, \forall j \in \overline{A}, \exists x, y :$
     $x \neq y \wedge (\varphi_i(x) \neq \varphi_j(x) \vee \varphi_i(y) \neq \varphi_j(y))$

     $\implies \forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$
   - Par la contraposée du théorème de Rice, A n'est pas récursif.

Questions du test

5. L'ensemble des programmes qui calculent une fonction donnée $f : \mathbb{N} \to \mathbb{N}$. Votre réponse dépend-elle de la fonction f ?

Réponse :

- **Non récursif** si f est calculable.

  Soit $A = \{i \mid \forall n : \varphi_i(n) = f(n)\}$. $A \neq \emptyset$ car la fonction étant calculable, un programme i qui la calcule existe. $A \neq \mathbb{N}$ car il existe des programmes qui ne calculent pas f.

  $\Longrightarrow \forall i \in A, \forall n : \varphi_i(n) = f(n)$ et $\forall j \in \overline{A}, \exists n : \varphi_j(n) \neq f(n)$

  $\Longrightarrow \forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$

  Par la contraposée du théorème de Rice, A n'est pas récursif.

- **Récursif** si f non calculable pacr que dans ce cas $A = \emptyset$ (car il n'est pas possible d'écrire un programme qui calcule une fonction non calculable).

Questions du test

Les phrases suivantes sont-elles vraies ?

6. Il existe un programme Python qui décide si la fonction calculée par un programme donné a un domaine fini.

   Réponse : Faux

   - Soit $A = \{i \mid \varphi_i$ possède un domaine fini $\}$. $A \neq \emptyset$ car il existe de tels fonctions (e.g. : une fonction retournant 42 pour l'entrée 1 et $\perp$ pour toutes les autres entrées). $A \neq \mathbb{N}$ car il existe aussi des fonctions calculables n'ayant pas un domaine fini (par exemple : $f(n) = n$).
   - $\forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$ car ces fonctions ont des domaines différents : la première a un domaine fini tandis que la seconde a un domaine infini.
   - Par la contraposée du théorème de Rice, A n'est pas récursif. La fonction caractéristique de A n'est donc pas calculable et un tel programme ne peut donc exister.

Questions du test

7. Il n'existera jamais un programme Python ramasse-miettes (garbage collector) qui soit à la fois sûr et optimal.

   I.e. un programme capable de déterminer sans jamais faire d'erreur durant l'exécution de n'importe quel programme s'il est possible de libérer de l'espace mémoire qui a été utilisée précédemment (autrement dit, si l'espace mémoire en question ne sera plus utilisé d'une quelconque manière que ce soit durant toute la suite de l'exécution du programme).

   Réponse : Vrai

$$\left[\begin{array}{l}\texttt{x = 0}\\ P_n(k)\\ \texttt{return x}\end{array}\right.$$

Décider **de manière certaine** si on peut libérer `x` après la 1ère instruction revient à toujours pouvoir calculer $halt(n, k)$ pour savoir si `x` risque d'être ré-utilisé par après dans le programme. Or $halt$ est une fonction non-calculable. Il n'est donc pas possible de savoir si `x` sera ré-utilisé.

Questions du test

Les phrases suivantes sont-elles vraies ?

8. L'ensemble $A = \{i \mid P_i(x)$ se termine pour un certain $x\}$ est récursif.

   Réponse : Faux, il n'est pas récursif.

   - Soit $A = \{i \mid \exists x : \varphi_i(x) \neq \bot\}$, $A \neq \emptyset$ (e.g. $P_i(x) \equiv$ `return 0`) et $A \neq \mathbb{N}$ (e.g. $P_j(x) \equiv$ `while True: pass`)
   - $\forall i \in A, \exists x : \varphi_i(x) \neq \bot \; \forall j \in \overline{A}, \forall x : \varphi_j(x) = \bot$

     $$\implies \forall i \in A, \forall j \in \overline{A}, \exists x : \varphi_i(x) \neq \bot \wedge \varphi_j(x) = \bot$$
     $$\implies \forall i \in A, \forall j \in \overline{A}, \exists x : \varphi_i(x) \neq \varphi_j(x)$$
     $$\implies \forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$$
   - Par la contraposée du théorème de Rice, A n'est donc pas récursif.

Questions du test

9. L'ensemble $A = \{i \in \mathbb{N} \mid P_i(x) \text{ se termine pour un certain } x\}$ est récursivement énumérable.

   Réponse : Vrai
   On simule une exécution parallèle de $P_i(0)$, $P_i(1)$, $P_i(2)$, ... (infinité de P_i) en exécutant 1 opération élémentaire à la fois en suivant la table en diagonale, jusqu'au moment où un P_i se termine.

	Instructions du programme						
$P_i(0)$	●	●	●	●	●	●	$\cdots$
$P_i(1)$	●	●	●	●	`return 1`		
$P_i(2)$	●	●	●	●	●	●	$\cdots$
$\vdots$							

Questions du test

La phrase suivante est-elle vraie ?

10. Nous avons vu que la fonction qui calcule les résultats du prochain lotto est bien calculable (il suffit d'écrire autant de programmes qu'il y a de combinaisons possibles, chacun produisant une des combinaisons possibles quelle que soit l'entrée).
    Mais est-il possible, étant donné un programme Python, de déterminer s'il produira justement la bonne combinaison ?
    Votre réponse est-elle différente en fonction de si vous connaissez à l'avance la bonne combinaison ?

Questions du test

Réponse : Faux, cela n'est pas possible.

- Soit $A = \{i \mid \varphi_i \text{ produit la bonne combinaison}\}$. $A \neq \emptyset$ car la fonction est calculable. $A \neq \mathbb{N}$ car il existe aussi des fonctions calculables renvoyant la mauvaise combinaison.
- $\forall i \in A, \forall j \in \overline{A} : \varphi_i \neq \varphi_j$ car, par définition, les programmes i renvoient la bonne combinaison et les programmes j renvoient une mauvaise combinaison.
- Par la contraposée du théorème de Rice, A n'est pas récursif. La fonction caractéristique de A n'est donc pas calculable et il n'est donc pas possible de déterminer si un programme produira la bonne combinaison.
- Connaître la combinaison à l'avance ne change rien à la démonstration.

Question 1 du TP

Question : Let $A = \{i \mid \exists x, y : \varphi_i(x) = \varphi_i(y) \neq \perp, x \neq y\}$ be the set of programs that halts with returning the same value for at least two different inputs. Using Rice's theorem, show that A is not recursive.

Question 1 du TP

Question : Let $A = \{i \mid \exists x, y : \varphi_i(x) = \varphi_i(y) \neq \bot, x \neq y\}$ be the set of programs that halts with returning the same value for at least two different inputs. Using Rice's theorem, show that A is not recursive.

Réponse :

1. On vérifie que $A \neq \emptyset$ et $A \neq \mathbb{N}$:

$$\varphi_1(x) \triangleq 42 \quad \Longrightarrow \quad 1 \in A \quad \Longrightarrow \quad A \neq \emptyset$$

$$\varphi_2(x) \triangleq x \quad \Longrightarrow \quad 2 \in \overline{A} \quad \Longrightarrow \quad A \neq \mathbb{N}$$

Question 1 du TP

2. On montre qu'aucun programme dans A ne calcule la même fonction qu'un programme dans $\overline{A}$:
   $\forall i \in A$:
   $$\exists x, y : x \neq y \land \varphi_i(x) = \varphi_i(y) \land \varphi_i(x) \neq \bot \land \varphi_i(y) \neq \bot$$
   $\forall j \in \overline{A}$:
   $$\forall x, y : x \neq y \Rightarrow \varphi_j(x) \neq \varphi_j(y) \lor \varphi_j(x) = \bot \lor \varphi_j(y) = \bot$$
   $\forall i \in A, \forall j \in \overline{A}$:
   $$\begin{aligned} \exists x, y : x \neq y \land \varphi_i(x) = \varphi_i(y) \land \\ \big(\varphi_j(x) \neq \varphi_j(y) \lor \varphi_i(x) \neq \varphi_j(x) \lor \varphi_i(y) \neq \varphi_j(y)\big) \end{aligned}$$
   $\Longrightarrow \varphi_i \neq \varphi_j$

   Par le théorème de Rice, A n'est donc pas récursif.

Question 2 du TP

Question : Let $A \subseteq \mathbb{N}$, $A \neq \mathbb{N}$ and $A \neq \emptyset$. If there exists $i \in A$ and $j \in \mathbb{N} \setminus A$ such that $\varphi_i = \varphi_j$, can we say that A is recursive ? Prove your answer.

Question 2 du TP

Question : Let $A \subseteq \mathbb{N}$, $A \neq \mathbb{N}$ and $A \neq \emptyset$. If there exists $i \in A$ and $j \in \mathbb{N} \setminus A$ such that $\varphi_i = \varphi_j$, can we say that A is recursive ? Prove your answer.

Réponse : Faux

Contre-exemple :

$A = \{i \,|\, \forall x : P_i(x)$ s'arrête et P_i a un nombre impair d'instructions$\}$

Soient les programmes :

$P_1(x) \equiv$ `return x` $\qquad P_2(x) \equiv$ `sleep(1); return x`

On a bien $A \neq \emptyset$, $A \neq \mathbb{N}$ et $\exists i \in A, j \in \overline{A} : \varphi_i = \varphi_j$.

Question 2 du TP

À présent, montrons que A n'est pas récursif. Supposons par l'absurde qu'il le soit. Alors on a un programme $P_A(n)$ qui décide A. Mais on peut alors décider HALT :

$$P_{\mathsf{HALT}}(n,k) \equiv \left[\begin{array}{l} \texttt{if } P_n \texttt{ has an odd number of instructions:} \\ \quad \texttt{write } P_q(y) \equiv \left[\ \texttt{return } P_n(k)\right. \\ \texttt{else:} \\ \quad \texttt{write } P_q(y) \equiv \left[\begin{array}{l}\texttt{sleep(1)} \\ \texttt{return } P_n(k)\end{array}\right. \\ \texttt{return } P_A(q) \end{array}\right.$$

C'est absurde. Donc il ne peut être vrai que A est récursif.

Question 3 du TP

Question : Let $A \subseteq \mathbb{N}$ not recursive ($A \neq \mathbb{N}$ and $A \neq \emptyset$). Can we say that $\forall i \in A$ and $\forall j \in \mathbb{N} \setminus A$, $\varphi_i \neq \varphi_j$? Prove your answer.

Question 3 du TP

Question : Let $A \subseteq \mathbb{N}$ not recursive ($A \neq \mathbb{N}$ and $A \neq \emptyset$). Can we say that $\forall i \in A$ and $\forall j \in \mathbb{N} \setminus A$, $\varphi_i \neq \varphi_j$? Prove your answer.

Réponse : Faux

Contre-exemple :

$A = \{i \,|\, \forall x : P_i(x)$ s'arrête et P_i a un nombre impair d'instructions$\}$

n'est pas récursif (ce qui implique $A \neq \emptyset$ et $A \neq \mathbb{N}$).

Pourtant $\exists i \in A, j \in \overline{A} : \varphi_i = \varphi_j$.

Question 4 du TP

Question : Let $A = \{i \mid P_i(x) \text{ halts for every input } x\}$.

1. Is A recursive ? Why ?
2. Is A recursively enumerable ? Why ?
   (Hint : use the reduction method.)

Question 4 du TP

Question : Let $A = \{i \mid P_i(x)$ halts for every input $x\}$.

1. Is A recursive ? Why ?

Réponse : A n'est pas récursif

$$\forall i \in A : \forall x, \varphi_i(x) \neq \perp$$

$$\forall j \in \overline{A} : \exists x, \varphi_j(x) = \perp$$

$$\Longrightarrow \varphi_i \neq \varphi_j$$

Par le théorème de Rice, A n'est pas récursif (car $A \neq \emptyset$ et $A \neq \mathbb{N}$).

Question 4 du TP

Question : Let $A = \{i \mid P_i(x) \text{ halts for every input } x\}$.

2. Is A recursively enumerable ? Why ?

Réponse : A n'est pas récursivement énumérable car on peut réduire $\overline{HALT}$ (qui n'est pas rec. énum.) à A.

Réduction :

$$P_{\overline{H}}(n,k) \equiv \left[\begin{array}{l} \texttt{write } P_q(x) \equiv \left[\begin{array}{l} \texttt{do } x \texttt{ instructions of } P_n(k) \\ \texttt{if } P_n(k)\texttt{has terminated:} \\ \quad \texttt{while true: pass} \\ \texttt{else:} \\ \quad \texttt{return } 1 \end{array}\right. \\ \\ \texttt{return } P_A(q) \end{array}\right.$$