Nom : Sujet : Jury : Page : /

32.- Décidabilité et indécidabilité. Exemples.

Niveau : MP1
Prérequis : Langages formels + Logique
Motivation : Donner une intuition de la notion de calculabilité en montrant que, contrairement à ce qui a été vu jusque là, certains problèmes ne peuvent pas être résolus par des programmes informatique quel que soit le langage.

I) Introduction à la calculabilité

1.) Préliminaires

Proposition 1 : On souhaite définir ce qui est calculable, c'est-à-dire ce qui'un ordinateur peut calculer à l'aide d'une séquence d'opérations décrites par un algorithme.
On considère ici que cette séquence peut être infinie et que l'ordinateur n'a aucune limite mémorielle pour stocker des informations.

Définition 2 : Dans toute cette leçon, on appelle algorithme un programme rédigé en C ou OCaml.

Remarque 3 : Il est possible de définir une notion de calculabilité indépendamment du langage : on utilise pour cela les fameuses machines de Turing.

Remarque 4 : Pour éviter les problèmes liés à la taille maximale des entiers en C et OCaml, on se limitera essentiellement à des chaînes de caractères, qui permettent de représenter n'importe quel type manipulable par un ordinateur.

E.) Calculabilité et décidabilité

Définition 5 : Une fonction (mathématique) totale $f : E \to S$ est calculable s'il existe un existe un algorithme A tel que, pour toute entrée $e \in E$, l'algorithme A appliqué à e produit le résultat $f(e)$ en un temps fini.

Exemples 6 : Les fonctions usuelles telles que les polynômes, les parties entières des racines carrées, logarithmes, ... sont calculables.

Exemple 7 : Calcul de la fonction carré en OCaml

```
let carre (e: string): string =
    let n = BigInt.of_string e in
    BigInt.to_string (BigInt.mult n n)
```

Activité 8 : Programmation en OCaml d'un module BigInt permettant de manipuler des entiers de taille arbitraire.

Propriété 9 : Il existe un nombre infini indénombrable de fonctions non calculables.

Définition 10 : Un problème de décision sur un ensemble E est une fonction totale f de E vers $\mathbb{B} = \{V, F\}$. Un algorithme A résoud f si, pour toute entrée $e \in E$, l'algorithme A appliqué à e termine en un temps fini et produit le résultat $f(e)$. Chaque élément $e \in E$ du domaine d'entrées est appelé une instance du problème. On dit que $e \in E$ est positive si $f(e) = V$, ou négative sinon.

Remarque 11 : On suppose ici que le domaine E des entrées possible est infini et dénombrable, pour avoir une représentation finie des données.

Exemples 12 : Exemples de problèmes de décision :
- "n est-il premier ?" pour $n \in \mathbb{N}$
- "g est-il connexe ?" pour g un graphe.

Définition 13 : Un problème de décision $f : E \to \mathbb{B}$ est **décidable** si f est calculable, c'est-à-dire qu'il existe un algorithme A tel que, pour toute entrée $e \in E$, l'algorithme A appliqué à e renvoie f(e) en un temps fini.
Un problème **indécidable** est un problème de décision qui n'est pas décidable, c'est-à-dire qui ne peut pas être résolu par un algorithme.

Exemples 14 :
- "n est-il premier ?" pour $n \in \mathbb{N}$ est décidable.
- "g est-il connexe ?" pour g un graphe, est décidable.
- "G peut-elle générer le mot vide ?" pour G une grammaire algébrique, est décidable.

Remarque 15 : Les problèmes pour lesquels E est fini sont peu intéressant d'un point de vue théorique car on peut, du moins en théorie, énumérer toutes les entrées possibles, déterminer à l'avance le résultat attendu pour chacune, puis proposer "l'algorithme" qui se contente de consulter la table des résultats précalculés pour répondre immédiatement.

Exemple 16: "f s'arrête-t-elle sur l'entrée <=> ?" pour f une fonction OCaml string → string de 100 caractères ou moins est décidable.

Remarque 17: Les problèmes de décision où l'entrée n'apparait pas dans la sortie sont décidables et peu intéressants.

Exemples 18:
- "P = NP ?" pour $n \in \mathbb{N}$, est décidable.
- "Les extra-terrestres existent-ils ?" pour g un graphe, est décidable.

Définition 19: Un problème de décision $f : E \to \mathbb{B}$ est Semi-décidable s'il existe un algorithme A tel que pour toute entrée $e \in E$, l'algorithme A appliquée à e :
- termine en un fini et renvoie V si $f(e) = V$
- renvoie F ou ne termine pas sinon.

II] Problèmes décidables et indécidables

1.) Problème de l'arrêt

Définition 20: Le problème de l'arrêt est le problème de décision halts prenant en entrée :
- une chaîne de caractère f contenant le code source d'un programme OCaml
- une chaîne de caractères e contenant les entrées pour le programme donné par f

et renvoyant V si f termine sur e en un temps fini et F sinon.

Théorème 21: Le problème de l'arrêt est indécidable.

2.) Algorithme universel

Propriété 22: Il existe une fonction eval : string → string → string OCaml prenant en entrée le code source s d'une fonction OCaml f de type string → string, et un argument e de type string pour f, et telle que :
- eval s e termine et renvoie la valeur produite par f e si l'exécution de f e termine
- eval s e ne se termine pas si f e ne termine pas.

Remarque 23: La fonction eval peut être vue comme un interpréteur OCaml (que l'on sait exister).

3.) Réductions calculatoires

Définition 24: Soient $f_1 : E_1 \to \mathbb{B}$ et $f_2 : E_2 \to \mathbb{B}$ deux problèmes de décision. On dit que f_1 se réduit (calculatoirement) à f_2, et on note $f_1 \leq f_2$, s'il existe une fonction calculable $g : E_1 \to E_2$ telle que, pour tout $e \in E_1$, on a : $f_1(e) = f_2(g(e))$.

Propriété 25: Soient $f_1 : E_1 \to \mathbb{B}$ et $f_2 : E_2 \to \mathbb{B}$ deux problèmes de décision. Si f_1 est indécidable et $f_1 \leq f_2$ alors f_2 est indécidable.

Exemple 26: Montrer que le problème consistant à déterminer si une fonction OCaml de type string → bool renvoie toujours true sur toute entrée est indécidable.

On peut montrer ce résultat par réduction au problème de l'arrêt. Pour cela, on suppose par l'absurde qu'il existe une fonction OCaml trivial : string → bool telle que trivial s = V si et seulement si s est le code d'une fonction f : string → bool telle que f renvoie true pour toute entrée. On construit alors la réduction suivante:

```
let halts (s: string) (e: string) : bool =
    trivial "fun _ -> let _ = eval s e in true"
```

Exemple 27 : Problème de Correspondance de Post (PCP)

Le problème défini par:
- entrées : $(u_1, v_1), \dots, (u_n, v_n)$ sur un alphabet Σ
- Sortie : existe-t-il $i_1, \dots, i_k \in [\![1; n]\!]$ tels que $u_{i_1} \cdots u_{i_k} = v_{i_1} \cdots v_{i_k}$

est indécidable.

4.) Problèmes sur les langages formels

Exemples 28 : Les problèmes de décision consistant à déterminer si, pour deux automates finis A_1 et A_2, on a : $L(A_1) \subseteq L(A_2)$, $L(A_1) = L(A_2)$, $L(A_1) \cap L(A_2) = \emptyset$ et $L(A_1) = \Sigma^*$ sont décidables.

DEV : Décidabilité des problèmes de grammaires algébriques

4.) Indécidabilité de l'équivalence sémantique

DEV : Théorème de Rice