Théorie de la complexité – Travaux Dirigés n. 3

1 Décidabilité et indécidabilité

On rappelle quelques définitions importantes en commençant par celle des fonctions calculables.

Définition 1 (Fonctions calculables)

Étant donné un alphabet Σ, une fonction (totale) $f : \Sigma^* \to \Sigma^*$ est dite calculable (ou récursive) quand elle peut s'implémenter par une machine de Turing.

Attention à la phase implicite de codage

On utilisera souvent le terme "calculable" pour désigner des fonctions dont les ensembles de départ et d'arrivée ne sont pas un ensemble de mots Σ^*. Dans ce cas on suppose implicitement qu'un *codage* (par des mots) des objets que nous considérons est fixé. Par exemple, on peut parler de fonctions $g : \mathbb{N} \to \mathbb{N}$ calculables (ou non) puisque-on peut coder les entiers en binaire sur l'alphabet $\Sigma = \{0,1\}$. Dans ce cas, l'énoncé "g est calculable" veut dire qu'il existe une fonction calculable (au sens de la Définition 1) $f : \Sigma^* \to \Sigma^*$ telle que pour tout mot $w \in \Sigma^*$, si $n \in \mathbb{N}$ est l'entier codé par w, alors $f(w)$ code $g(n)$. Bien sûr, on peut coder des objets bien plus compliqués que des entiers (par exemple, des machines de Turing, des graphes, etc).

Attention, dans les énoncés, le codage sera souvent implicite : on parle directement de fonction calculable même si les ensembles de départ et d'arrivée ne sont pas un même ensemble de mots Σ^*.

On passe maintenant aux langages décidables et indécidables. On rappelle que cette définition considère les machines de Turing de décision.

Définition 2 (Langages décidables)

Soit Σ un alphabet et $L \subseteq \Sigma^*$ un langage. On dit que L est *décidable* (ou récursif), quand il existe une machine de Turing (de décision) M telle que pour tout mot $w \in \Sigma^*$:

- $w \in L$ si et seulement si M accepte w.
- $w \notin L$ si et seulement si M rejette w.

Symétriquement, un langage qui n'est pas décidable est dit *indécidable*

Observons quand un langage est décidable, une machine de Turing qui témoigne de cette propriété doit terminer sur toute entrée. On va relaxer cette condition pour obtenir une autre notion : la semi-décidabilité.

Définition 3 (Langages semi-décidables)

Soit Σ un alphabet et $L \subseteq \Sigma^*$ un langage. On dit que L est *semi-décidable* (ou récursivement énumérable), quand il existe une machine de Turing (de décision) M telle que pour tout mot $w \in \Sigma^*$, $w \in L$ si et seulement si M accepte w.

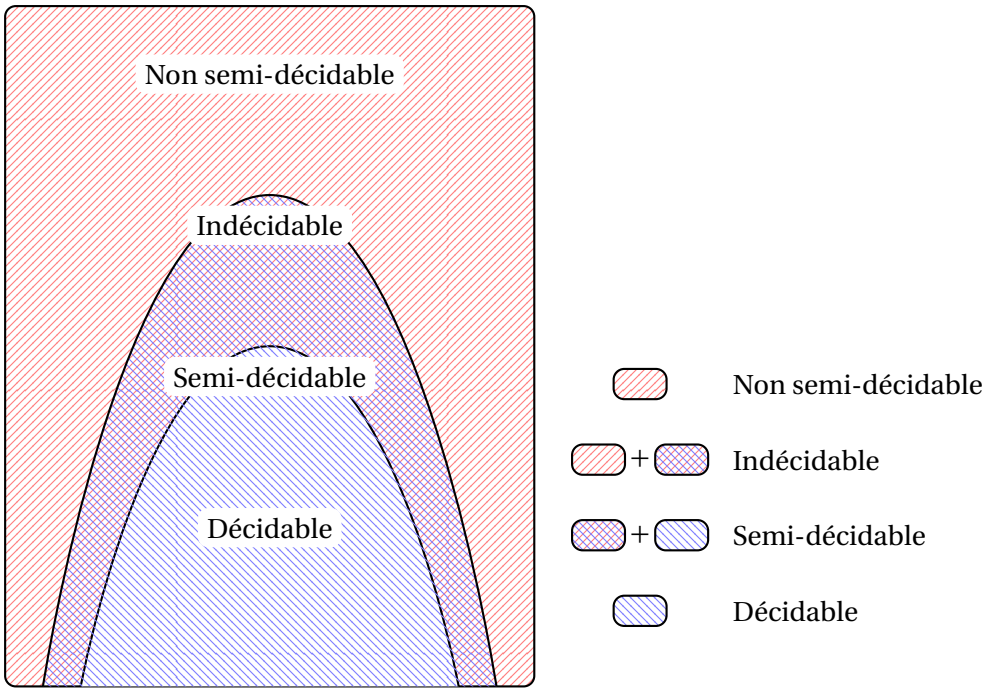


FIGURE 1 – Comparaisons des classes de langages

On remarquera que la définition des langages décidables est plus contraignante que celle des langages semi-décidables. Si L est semi-décidable, une machine qui témoigne de cette propriété a deux comportements possibles sur un mot s'entrée $w \notin L$: soit elle rejette, soit elle ne termine pas. Ainsi, tout langage décidable est en particulier semi-décidable. On résume la situation dans la Figure 1.

Des langages aux problèmes de décision : encore et toujours le codage

De même que pour le terme "calculable", on utilisera souvent les termes "décidable", "indécidable" et "semi-décidable" pour désigner des ensembles qui de ne sont pas des langages (i.e. contenant autre chose que des mots). Encore une fois, cela suppose qu'un codage est fixé. Par exemple, on dira que l'ensemble $E \subseteq \mathbb{N}$ des entiers pairs est décidable car le langage des mots de $\{0,1\}^*$ codant un entier pair est décidable (au sens de la Définition 2). De la même façon, l'ensemble des graphes complets (toutes les paires de sommets sont reliées par une arrête) est décidable car on peut coder un graphe par un mot de $\{0,1\}^*$ et le langage des mots codant un graphe complet est décidable (au sens de la Définition 2). Comme expliqué ci-dessus, le codage est souvent implicite dans nos énoncés.

Enfin, dans ce contexte, il est usuel (et équivalent) de parler de "problèmes de décision" au lieu d'ensembles. Par exemple, les deux ensembles ci-dessus correspondent aux problèmes suivants :

Parité

ENTRÉE : Un entier $n \in \mathbb{N}$.
QUESTION : Est-ce que n est pair ?

Complétude

ENTRÉE : Un graphe G.
QUESTION : Est-ce que G est complet ?

Nous sommes maintenant prêts à donner un premier exemple de problème indécidable : le problème de l'arrêt, que nous présentons dans le théorème suivant.

Théorème 4 (Indécidabilité du problème de l'arrêt)

On considère le problème de l'arrêt des machines de Turing (noté **HALT**) :
- **ENTRÉE** : Une machine de Turing M et un mot w écrit sur l'alphabet d'entrée de M.
- **QUESTION** : Est-ce que l'exécution de M sur w termine ?

Le problème **HALT** est semi-décidable et indécidable.

Exercices

Exercice 1 Let but est de construire une fonction $f : \mathbb{N} \to \mathbb{N}$ qui est croissante (au sens large) et non-calculable.

1. Justifier qu'il existe une fonction $g : \mathbb{N} \to \{0,1\}$ qui est non-calculable.
2. Utiliser g pour construire la fonction désirée f. ∎

Exercice 2 Montrer que toutes les fonctions $f : \mathbb{N} \to \mathbb{N}$ décroissantes (au sens large) sont calculables. ∎

Exercice 3 Parmi les cinq fonctions suivantes quatre sont calculables et savoir si la cinquième est calculable ou non est un problème ouvert. Trouver les quatre fonctions calculables (justifier) :

1. $$f : \mathbb{N} \to \{0,1\}$$
$$n \mapsto \begin{cases} 1 \text{ si le développement décimal de } \pi \text{ contient un nombre fini de 1} \\ 0 \text{ sinon} \end{cases}$$

2. $$f : \mathbb{N} \to \{0,1\}$$
$$n \mapsto \begin{cases} 1 \text{ si } \pi \text{ contient exactement } n \text{ fois le chiffre 1 dans son développement décimal} \\ 0 \text{ sinon} \end{cases}$$

3. $$f : \mathbb{N} \to \{0,1\}$$
$$n \mapsto \begin{cases} 1 \text{ si } \pi \text{ contient au moins } n \text{ fois le chiffre 1 dans son développement décimal} \\ 0 \text{ sinon} \end{cases}$$

4. $$f : \mathbb{N} \to \{0,1\}$$
$$n \mapsto \begin{cases} 1 \text{ si } \pi \text{ contient un bloc } \textit{maximal} \text{ de exactement } n \text{ chiffres 1 consécutifs} \\ \quad \text{dans son développement décimal} \\ 0 \text{ sinon} \end{cases}$$

5. $$f : \mathbb{N} \to \{0,1\}$$
$$n \mapsto \begin{cases} 1 \text{ si } \pi \text{ contient un bloc } \textit{maximal} \text{ de au moins } n \text{ chiffres 1 consécutifs} \\ \quad \text{dans son développement décimal} \\ 0 \text{ sinon} \end{cases}$$

∎

Exercice 4 On considère un alphabet Σ et un langage $L \subseteq \Sigma^*$. On suppose que L et $\Sigma^* \setminus L$ sont tous deux semi-décidables (on dit que L est à la fois semi-décidable et co-semi-décidable). Montrer que L est décidable. ∎

Exercice 5 On considère le problème **un-HALT** suivant :
— **ENTRÉE** : Une machine de Turing M et un mot w écrit sur l'alphabet d'entrée de M.
— **QUESTION** : Est-ce que l'exécution de M sur w ne termine pas ?
Le problème **un-HALT** est-il décidable ? Est-il semi-décidable ? ∎

Exercice 6 Soit $X \subseteq \mathbb{N}$ un ensemble *semi-décidable*. Montrer qu'il existe un ensemble *décidable* $Z \subseteq \mathbb{N} \times \mathbb{N}$ tel que

$$X = \{x \mid \exists y \text{ tel que } (x,y) \in Z\}.$$

∎

Exercice 7 On considère l'alphabet $\Sigma = \{0,1\}$. Étant donné un langage $L \subseteq \Sigma^*$, on dit qu'*une machine M énumère L* si l'exécution de M sur le mot vide (ε) écrit successivement tous les mots de L (et uniquement ceux-ci) sur sa bande. En particulier, si L est infini, l'exécution de M ne termine pas (on sait juste que tout mot de L finira un jour par être écrit sur la bande de M).

Étant donné un langage $L \subseteq \Sigma^*$ montrer que L est semi-décidable si et seulement si il existe une machine de Turing qui énumère L (d'où le terme récursivement énumérable). ∎

Exercice 8 On considère l'alphabet $\Sigma = \{0,1\}$ et deux langages $K, L \subseteq \Sigma^*$ qui sont semi-décidables (potentiellement non disjoints, i.e. $K \cap L \neq \emptyset$). Construire deux autres langages semi-décidables $K', L' \subseteq \Sigma^*$ tels que :

— $K' \subseteq K$,
— $L' \subseteq L$,
— $K' \cap L' = \emptyset$,
— $K' \cup L' = K \cup L$. ∎

2 Réductions - Comment montrer l'indécidabilité d'un problème

Maintenant que nous connaissons un premier problème indécidable (**HALT**), nous allons utiliser une nouvelle méthode pour prouver que d'autres problèmes indécidables existent : les *réductions*.

Définition 5 (Réductions)
Considérons un alphabet Σ et deux langages $K, L \subseteq \Sigma^*$. Une réduction de K vers L est une fonction **calculable** $f : \Sigma^* \to \Sigma^*$ telle que :

$$\text{Pour tout } w \in \Sigma^*, \quad w \in K \text{ si et seulement si } f(w) \in L$$

On peut maintenant énoncer le théorème que nous utiliseront pour prouver que des langages (et les problèmes qu'ils encodent) sont indécidables.

Théorème 6

Considérons un alphabet Σ et deux langages $K, L \subseteq \Sigma^*$ tels qu'il existe une réduction de K vers L. On a les propriétés suivantes :

1. Si L est décidable alors K est décidable.
2. Si K est indécidable alors L est indécidable.
3. Si L est semi-décidable alors K est semi-décidable.
4. Si K n'est pas semi-décidable alors L n'est pas semi-décidable.

Le Théorème 6 nous donne une nouvelle méthode pour prouver qu'un langage est indécidable. On dispose d'un "stock" de problèmes indécidables (qui pour l'instant n'en contient qu'un seul : **HALT**). Si on veut prouver qu'un nouveau problème P est indécidable, on peut choisir (astucieusement) un problème Q qui est déjà dans le stock et exhiber une réduction de Q vers P. On obtient ensuite l'indécidabilité de P par la seconde propriété du Théorème 6.

Exercices

Exercice 9 Montrer que la relation de réduction entre langages est transitive, c'est-à-dire si L_1 se réduit à L_2 et L_2 se réduit à L_3, alors L_1 se réduit à L_3. ∎

Exercice 10 Parmi les problèmes suivants, lesquels sont décidables, lesquels sont semi-décidables et lesquels sont indécidables? On justifiera chaque affirmation (en particulier l'indécidabilité doit être prouvée par une réduction).

- **HALT**$_\varepsilon$: étant donné une machine de Turing M, est-ce que M s'arrête sur le mot vide "ε"?
- **UNIV** : étant donné une machine de Turing de décision M et un mot d'entrée w, est-ce que M accepte w?
- **HALT$_{\text{timed}}$** : étant donné une machine de Turing M, un mot d'entrée w et un entier t, est-ce que M accepte w en moins de t étapes?
- **HALT$_{\text{or}}$** : étant donné deux machines de Turing M_1, M_2 et un mot d'entrée w, est-ce que M_1 ou M_2 s'arrête sur w?
- **HALT**$_\exists$: étant donné une machine de Turing M, est-ce qu'il existe un mot w tel que M s'arrête sur w?
- **STATE** : étant donné une machine de Turing M et un entier n, est-ce que M a n états?
- **HALT**$_\forall$: étant donné une machine de Turing M, est-ce que M s'arrête sur tous les mots d'entrée w?
- **EQUIV** : étant donné deux machines de Turing M_1, M_2 est-ce que M_1 et M_2 donnent le même résultat pour tout mot d'entrée w?
- **PI** : étant donné une machine de Turing M, est-ce que M calcule π? C'est-à-dire que pour tout $i \in \mathbb{N}$, M s'arrête sur l'entrée i et écrit le i-eme chiffre du développement décimal de π. ∎

3 Théorème de Rice

On présente maintenant le théorème de Rice qui généralise une partie de l'exercice 10.

Théorème 7

On considère un alphabet Σ. Soit F l'ensemble de toutes les fonctions partielles de Σ^* dans Σ^* implémentées par une machine de Turing. On considère un sous-ensemble $E \subseteq F$ tels que $E \neq \emptyset$ et $E \neq F$. Le problème suivant est indécidable :

- **ENTRÉE** : Une machine de Turing M.
- **QUESTION** : Est-ce que la fonction partielle implémentée par M appartient à E?

Une façon moins formelle d'énoncer le théorème de Rice est la suivante : "toute propriété sémantique et non-triviale des machines de Turing est indécidable".

- Le terme "sémantique" veut dire la propriété ne doit dépendre que de la sémantique de la machine (i.e. de la fonction partielle qu'elle implémente).
- Le terme "non-triviale" veut dire qu'il existe au moins une machine qui la satisfait et une autre qui ne la satisfait pas (celà correspond à l'hypothèse que $E \neq \emptyset$ et $E \neq F$ dans le théorème).

Exercices

Exercice 11 Montrer que pour tous les problèmes indécidables dans l'exercice 10, leur indécidabilité est une conséquence immédiate du théorème de Rice. ∎

Exercice 12 On considéré la sous-classe des machines de Turing *linéairement bornées*. Sur un mot d'entrée de taille $n \in \mathbb{N}$, une telle machine utilise un espace borné par n.

- Montrer que **HALT** est décidable pour les machines linéairement bornées : il existe une machine de Turing qui termine et répond correctement au problème de l'arrêt quand on lui passe en entrée une machine linéairement bornée (on ignore ce que la machine fait pour les autres entrées).

— Montrer que le problème $\mathbf{HALT}_{\exists}$ restreint aux machines linéairement bornées reste indécidable. ▪

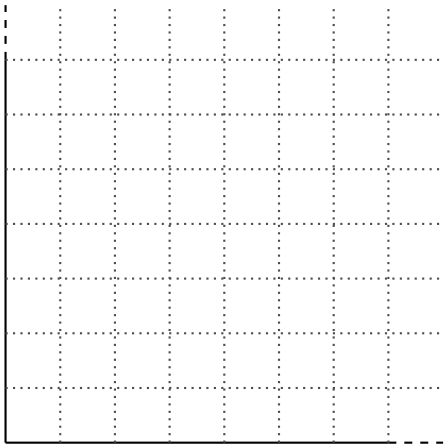
Exercice 13 Parmi les problèmes suivants, lesquels sont décidables ?

— Étant donné une machine de Turing M sur l'alphabet d'entrée $\Sigma = \{0,1\}$, est-il vrai que pendant l'exécution de M sur le mot vide "ε", les transitions déplacent toujours la tête de lecture vers la droite ?

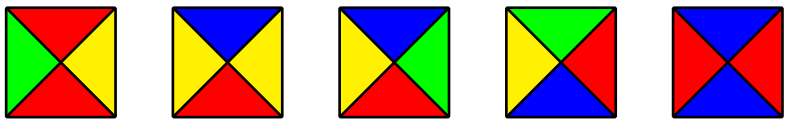
— Étant donné une machine de Turing M sur l'alphabet d'entrée $\Sigma = \{0,1\}$, est-il vrai que pendant l'exécution de M sur le mot vide "ε", une transition écrit une lettre de Σ sur la bande ?

— Étant donné une machine de Turing M sur l'alphabet d'entrée $\Sigma = \{0,1\}$, est-il vrai que pendant l'exécution de M sur le mot vide "ε", une transition écrit la lettre de "1" sur la bande ? ▪

Exercice 14 — Dominos de Wang. On considère le problème de pavage suivant. On dispose d'une grille (infinie) qui couvre le quart du plan.



Chaque "carré" dans la grille est une case vide dans laquelle on peut placer un domino de Wang. Un domino de Wang est un carré dont chaque côté possède une couleur. On donne des exemples de dominos ci-dessous.



Attention, les dominos sont orientés : on ne peut pas les tourner.

Considérons une paire (D, d_0) où D est un ensemble fini de dominos et $d_0 \in D$ est un domino spécial qu'on appelle le domino de départ. On dit qu'on peut **paver le quart de plan avec** (D, d_0) si on peut remplir chaque case de la grille (infinie) ci-dessus avec des dominos de D (on a un nombre infini de copies de chaque domino dans D) tout en satisfaisant les deux conditions suivantes :

— Une copie de d_0 doit être placée dans la première case (celle en bas à gauche).

— Étant donné deux dominos adjacents, le côté qu'ils ont en commun doit avoir la même couleur.

Montrer que le problème suivant est indécidable :

INPUT :	Une paire (D, d_0) où D est un ensemble fini de dominos et $d_0 \in D$
QUESTION :	Peut-on paver le quart de plan avec (D, d_0) ?

Suggestion : on va devoir réduire le problème de l'arrêt (ou plutôt son complémentaire) au problème de pavage. C'est-à-dire qu'étant donné une machine de Turing M, on doit expliquer comment construire un

ensemble de dominos D et un domino de départ d_0 tel que M ne s'arrête pas sur le mot vide si et seulement si on peut paver le quart de plan avec (D, d_0). ∎