Théorème de Rice

Julien Devevey

2018-2019

Ref : Carton - Langages formels, Calculabilité et Complexité p.160 et 162

Définition 1. On pose $L_\in = \{\langle \mathcal{M}, w\rangle, w \in L(\mathcal{M})\}$, le langage d'acceptation, c'est à dire le langage qui contient les encodages des machines de Turing et des mots qu'elles acceptent.

Lemme 2. Le langage $L_\in$ est indécidable.

Démonstration.
On va raisonner par l'absurde, en supposant que $L_\in$ est décidable. Soit alors $\mathcal{A}$ qui décide $L_\in$, c'est à dire que $L(\mathcal{A}) = L_\in$ et que $\mathcal{A}$ s'arrête sur toute entrée. On construit alors un exemple problématique en la machine $\mathcal{Q}$:

Entrée: $\langle \mathcal{M}\rangle$
si $\mathcal{A}$ accepte $\langle \mathcal{M}, \langle \mathcal{M}\rangle\rangle$ **alors**
    rejeter
sinon
    accepter
fin si

On choisit de lancer $\mathcal{Q}$ sur l'entrée $\langle \mathcal{Q}\rangle$. On sait déjà que $\mathcal{Q}$ va s'arrêter, elle peut alors soit rejeter, soit accepter.

— Si $\mathcal{Q}$ accepte $\langle \mathcal{Q}\rangle$, alors $\mathcal{A}$ accepte l'entrée $\langle \mathcal{Q}, \langle \mathcal{Q}\rangle\rangle$ par définition. Pourtant, comme $\mathcal{Q}$ a accepté $\langle \mathcal{Q}\rangle$, on en déduit que $\mathcal{A}$ a rejeté l'entrée $\langle \mathcal{Q}, \langle \mathcal{Q}\rangle\rangle$: c'est une contradiction.

— Si $\mathcal{Q}$ rejette $\langle \mathcal{Q}\rangle$, alors $\mathcal{A}$ rejette l'entrée $\langle \mathcal{Q}, \langle \mathcal{Q}\rangle\rangle$ par définition. Pourtant, comme $\mathcal{Q}$ a rejeté $\langle \mathcal{Q}\rangle$, on en déduit que $\mathcal{A}$ a accepté l'entrée $\langle \mathcal{Q}, \langle \mathcal{Q}\rangle\rangle$: c'est une contradiction.

On arrive dans tous les cas à une contradiction. Alors la machine $\mathcal{A}$ n'existe pas, c'est à dire que $L_\in$ n'est pas décidable. □

Théorème 3 (Rice). Pour toute propriété non triviale P sur les langages récursivement énumérables, le problème de savoir si le langage $L(\mathcal{M})$ d'une machine de Turing $\mathcal{M}$ vérifie P ou non est indécidable.

Démonstration. On va procéder cette fois-ci en proposant une réduction de

$L_\in$ vers L_P, en posant $L_P = \{\langle \mathcal{M} \rangle, L(\mathcal{M}) \text{ vérifie } P\}$, c'est à dire le langage contenant tous les encodages de machines dont le langage vérifie P. Quitte à remplacer P par $\neg P$, on peut supposer que $\emptyset$ ne vérifie pas P. De plus, comme la propriété n'est pas triviale, on a dans les deux cas l'existence de $\mathcal{M}_0$ telle que $\langle \mathcal{M}_0 \rangle \in L_P$. Maintenant, soit $(\mathcal{M}, w)$, une machine de Turing et un mot. On pose alors $\mathcal{M}_w$ la machine définie ainsi :

Entrée: u

```
si M accepte w alors
    simuler M_0 sur u et renvoyer le résultat
sinon
    rejeter
fin si
```

Analysons ce qu'il se passe lors de l'exécution de $\mathcal{M}_w$ sur l'entrée u :

— Si $\mathcal{M}$ ne s'arrête pas sur w : $\mathcal{M}_w$ ne s'arrête sur aucune entrée et $L(\mathcal{M}_w) = \emptyset$.
— Si $\mathcal{M}$ rejette w, alors $\mathcal{M}_w$ rejette tout u et donc $L(\mathcal{M}_w) = \emptyset$.
— Si $\mathcal{M}$ accepte w, alors $u \in L(\mathcal{M}_w) \Leftrightarrow u \in L(\mathcal{M}_0)$

Comme $\emptyset$ ne vérifie pas P, on a alors l'équivalence suivante : $w \in L(\mathcal{M}) \iff L(\mathcal{M}_w)$ vérifie P .

De plus, la fonction $f : \langle \mathcal{M}, w \rangle \mapsto \langle \mathcal{M}_w \rangle$ est calculable par une machine de Turing déterministe. On a alors $L_\in \leq_m L_P$ et on sait par le lemme précédent que $L_\in$ est indécidable. D'où L_P est aussi indécidable. □

Remarque-Contre Exemple 4. Le théorème précédent concerne uniquement les propriétés sur le *langage* d'une machine de Turing, et pas sur la machine elle même, notamment sur la manière dont elle effectue ses calculs. Par exemple, le langage $L(k) = \{\langle \mathcal{M} \rangle, \mathcal{M} \text{ s'arrête en au plus } k \text{ étapes}\}$ est décidable pour n'importe quel $k \in \mathbb{N}^*$.

Démonstration.

Il suffit de simuler les k premières étapes d'une machine $\mathcal{M}$ sur une entrée w puis de s'arrêter, et d'accepter si la machine s'est arrêtée, de rejeter sinon.