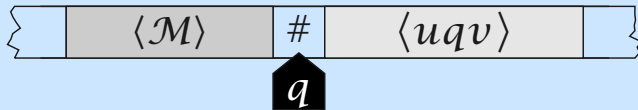


Calculabilité & Complexité

Calculabilité (4/5) : problèmes indécidables

Nicolas Ollinger (LIFO, Université d'Orléans)

M1 info, Université d'Orléans — S2 2025/2026



1. Dans l'épisode précédent...

Limites du calcul

Proposition Il existe des (*multitudes de*) langages **non rékursifs**.

Il suffit de compter :

- les MT sont **dénombrables** (on peut les coder) ;
- les langages sur $\{a, b\}$ ne le sont **pas**.


Définition Un ensemble X est **dénombrable** s'il existe une application injective de X dans \mathbb{N} .

Idée utiliser l'**argument diagonal** de Cantor

Diagonalisation

Supposons que (L_i) énumère les langages sur $\{a, b\}$.

	ε	a	b	aa	ab	ba	bb	aaa	aab	\dots
L_0	0	0	0	0	0	0	0	0	0	\dots
L_1	0	1	0	1	1	0	0	1	1	\dots
L_2	1	1	1	1	1	1	1	1	1	\dots



$$L'(k) = 1 - L_i(k)$$

Construisons $L' = \{\langle k \rangle \mid \langle k \rangle \notin L_k\}$ qui est distinct de chaque (L_i) .
Contradiction !

Problème de l'arrêt

entrée : *une machine de Turing \mathcal{M} et un mot u*
question : *est-ce que \mathcal{M} s'arrête sur l'entrée u ?*

2. Indécidabilité et limites du calcul

Problème de décision

Définition Un **problème de décision** \mathcal{P} est un prédicat sur un ensemble récursif d'entrées E , les instances du problème.

Le **langage associé** au problème est le langage des codages des **instances positives**, i.e. $L_{\mathcal{P}} = \{\langle x \rangle \mid x \in E \wedge \mathcal{P}(x)\}$.

Problème de l'arrêt

entrée : *une machine de Turing \mathcal{M} et un mot u*

question : *est-ce que \mathcal{M} s'arrête sur l'entrée u ?*

Définition Un problème de décision est **décidable** si le langage associé est récursif, **indécidable** sinon.

Le problème de l'arrêt

Problème de l'arrêt

entrée : *une machine de Turing \mathcal{M} et un mot u*

question : *est-ce que \mathcal{M} s'arrête sur l'entrée u ?*

Le langage associé est $K = \{\langle \mathcal{M}, u \rangle \mid \mathcal{M} \text{ s'arrête sur } u\}$.

Proposition K est **récursivement énumérable**.

Idée Il suffit de modifier légèrement la machine de Turing **universelle** pour qu'elle détecte l'arrêt.

Un problème indécidable

Théorème Le **problème de l'arrêt** est **indécidable**.

Par l'absurbe, en supposant K reconnu par une MT totale \mathcal{H} , on construit une MT Δ qui sur l'entrée $\langle \mathcal{M} \rangle$:

1. exécute \mathcal{H} sur l'entrée $\langle \mathcal{M}, \mathcal{M} \rangle$
2. si \mathcal{H} accepte alors Δ **diverge** ;
3. si \mathcal{H} rejette alors Δ **s'arrête**.

Par construction, la MT Δ possède un **codage** $\langle \Delta \rangle$. Étudions $\Delta(\langle \Delta \rangle)$!

Corollaire le langage K n'est **pas co-récurсивement énumérable**.

Des problèmes indécidables

Une technique pour dériver de nombreux problèmes indécidables à partir d'un premier consiste à utiliser des **réductions** : des pré-ordres sur les langages qui **préservent la récursivité**.

Définition Soient $A \subseteq \Sigma^*$ et $B \subseteq \Gamma^*$ deux langages. Une **réduction** (many-one) de A à B est une fonction totale calculable $f : \Sigma^* \rightarrow \Gamma^*$ telle que

$$u \in A \Leftrightarrow f(u) \in B \quad \forall u \in \Sigma^* .$$

Le langage A **se réduit** au langage B , noté $A \leq_m B$ s'il existe une réduction de A à B .

Proposition La relation \leq_m est une relation de **pré-ordre**.

Réductions *many-one*

Proposition Si $A \leq_m B$ alors $\overline{A} \leq_m \overline{B}$.

Idée L'équivalence passe au complémentaire.

Proposition Si $A \leq_m B$ avec B **récursivement énumérable** alors A est **récursivement énumérable**.

Idée Construire un reconnaisseur pour A à partir d'un reconnaisseur pour B .

Corollaire Si \mathcal{P} est **indécidable** et si $L_{\mathcal{P}} \leq_m L_{\mathcal{P}'}$ alors \mathcal{P}' est **indécidable**.

Exemple de réduction

Problème de l'arrêt sur l'entrée vide

entrée : *une machine de Turing \mathcal{M}*

question : *est-ce que \mathcal{M} s'arrête sur l'entrée vide ε ?*

Le langage associé est $K_0 = \{\langle \mathcal{M} \rangle \mid \mathcal{M} \text{ s'arrête sur } \varepsilon\}$.

Montrons que $K_0 \leq_m K$ et $K \leq_m K_0$!

Exemple de réduction (bis)

Égalité de fonctions

entrée : *deux machines de Turing \mathcal{M} et \mathcal{N}*

question : *est-ce que \mathcal{M} et \mathcal{N} calculent la même fonction ?*

Le langage associé est $E = \{\langle \mathcal{M}, \mathcal{N} \rangle \mid f_{\mathcal{M}} \doteq f_{\mathcal{N}}\}$.

Montrons que $K_0 \leq_m E$ et $K_0 \leq_m \bar{E}$!

Théorème de Rice

Une **MT** est un objet **syntaxique** qui décrit la manière d'organiser un calcul.

Le **langage reconnu** par la machine est de nature **sémantique**, c'est le résultat du calcul.

Le **théorème de Rice** montre que tout problème qui voudrait caractériser une propriété purement sémantique des machines est **indécidable** ou **trivial**.

Remarque La démonstration se fait par **réduction**.

Formalisation du théorème de Rice

Définition Une propriété \mathfrak{P} des langages est **non triviale** s'il existe deux machines de Turing \mathcal{M} et \mathcal{M}' telles que $L(\mathcal{M})$ vérifie \mathfrak{P} et $L(\mathcal{M}')$ ne vérifie pas \mathfrak{P} .

Théorème Soit \mathfrak{P} une propriété des langages **non triviale**. Le problème d'appartenance à \mathfrak{P} est **indécidable**.

Problème d'appartenance à \mathfrak{P}

entrée : *une machine de Turing \mathcal{M}*

question : *est-ce que $L(\mathcal{M})$ vérifie \mathfrak{P} ?*