

Qu'est-ce qu'une machine ? (I/III)

Eric OLIVIER^{1 2}

Résumé. – La théorie des machines de Turing reformule et clarifie un certain nombre de questions portant sur les fondements (logiques) des mathématiques. Ainsi la question "Qu'est-ce qu'une machine ?" est-elle équivalente à la question "Qu'est-ce qu'un calcul ?". Richard Feynman résume cela en affirmant que *n'importe quelle procédure de calcul à laquelle on pourrait penser, est équivalente au calcul d'une machine de Turing – les fonctions récursives générales sont Turing-calculables et vice-versa – et on peut donc prendre "Turing-calculable" pour un synonyme effectif de "calculable"*. Notons enfin que le calcul automatique (i.e. le calcul effectué par une machine de Turing) distingue la notion de *proposition démontrable* de celles de *proposition vraie*, *décidable*, *indécidable* : cela éclaire les travaux révolutionnaires de Gödel sur la *complétude* et la *consistance* des théories mathématiques.

1. ALPHABETS ET LANGAGES

Un alphabet $\mathcal{A}$ est un ensemble fini ou infini dénombrable dont les éléments sont appelés lettres, chiffres, symboles ou encore digits ; on note $\mathcal{A}^{(0)} := \{\phi\}$ (ϕ est le mot vide) et pour tout entier $n \geq 1$ l'ensemble $\mathcal{A}^{(n)}$ est constitué des mots finis composés de (la concaténation de) n lettres prises dans $\mathcal{A}$. (Pour tout $n \geq 1$, il est facile de mettre en bijection $\mathcal{A}^{(n)}$ avec le produit cartésien $\mathcal{A}^n = \mathcal{A} \times \cdots \times \mathcal{A}$ ayant n facteurs égaux à $\mathcal{A}$.) Le langage associé à $\mathcal{A}$ est

$$\mathcal{A}^* = \bigcup_{n=0}^{\infty} \mathcal{A}^{(n)}.$$

(Ici on utilise un cas particulier de la notation de Kleene). L'application $(w, m) \mapsto wm$, définie sur $\mathcal{A}^* \times \mathcal{A}^*$ et à valeur dans $\mathcal{A}^*$, est appelée la concaténation : elle munit $\mathcal{A}^*$ d'une structure de monoïde (non commutatif) dont l'élément neutre est le mot vide. Soit w et m deux mots de $\mathcal{A}^*$; on dit que m est un préfixe (resp. suffixe) de w s'il existe $w' \in \mathcal{A}^*$ tel que $w = mw'$ (resp. $w = w'm$) ; s'il existe deux mots w' et w'' dans $\mathcal{A}^*$ tels que $w = w'mw''$, alors on dit que m factorise w . La longueur d'un mot $w \in \mathcal{A}^*$, notée $|w|$, est le nombre de lettres qui le composent (avec la convention que $|\phi| = 0$) ; il est alors facile de voir que $w \mapsto |w|$ est un morphisme de monoïde de $\mathcal{A}^*$ sur le monoïde additif $(\mathbb{N}, +)$ des nombres entiers positifs ou nuls. Dans la suite nous supposerons toujours les alphabets considérés totalement ordonnés ; cela permet de définir sur $\mathcal{A}^*$ un ordre total naturel appelé *ordre canonique*. Pour w et m deux mots distincts de $\mathcal{A}^*$ on note $w < m$ lorsque $|w| < |m|$ ou lorsque w précède m dans l'ordre lexicographique, dans le cas où $|w| = |m|$ (notons que l'ordre canonique diffère de l'ordre lexicographique). Un sous-ensemble $\mathcal{L}$ de $\mathcal{A}^*$ est appelé un *langage* sur $\mathcal{A}$.

1. GDAC-I2M UMR 7373 CNRS Université d'Aix-Marseille

2. eric.olivier@univ-amu.fr

2. MACHINES DE TURING

Le concept de *machine* introduit par Turing dans son article fondateur de 1936 [Tur36] peut se décrire comme un *mécanisme abstrait* pouvant se trouver dans un nombre fini d'*états internes*, dont on note l'ensemble $\mathcal{Q} = \{q_1 = I, \dots, q_r = F\}$. La machine est pourvue d'une tête de lecture/écriture permettant de lire/écrire sur un *ruban* des lettres prises dans un alphabet $\mathcal{A}$ – *fini* – appelé *alphabet d'exécution* ; le ruban lui-même est formé d'une infinité (linéaire bilatérale) de cases. Les cases du ruban sont vides, à l'exception d'un nombre *fini* d'entre elles où figurent des lettres de l'alphabet d'exécution ; par convention, l'alphabet d'exécution contient le symbole $\star$ permettant d'indiquer qu'une case est vide. Après chaque lecture/écriture, la tête peut se déplacer d'une case à droite (action R) ou à gauche (action L) ou encore rester immobile (action S). Ici les états internes $q_1 = I$ et $q_r = F$ sont respectivement l'état initial et l'état final (I et F seront toujours supposés distincts). Initialement, la machine est positionnée dans son état initial I, la tête étant prête à lire la lettre de la case en position de lecture/écriture. Plus précisément, c'est la *table de transition* qui définit les actions de la machine en fonction des lettres figurant sur le ruban : on la représente comme une application

$$T : \mathcal{Q} \times \mathcal{A} \rightarrow \mathcal{Q} \times \mathcal{A} \times \{L, S, R\}.$$

Par exemple (en supposant que 0 et 1 appartiennent à $\mathcal{A}$) la transition $T(I, 1) = (F, 0, L)$, signifie que T étant dans l'état interne I et lisant la lettre 1 sur la case courante du ruban, écrit la lettre 0 sur cette même case, puis se positionne dans l'état interne F tout en déplaçant la tête de lecture/écriture d'une case à gauche. Un couple $(Q, a) \in \mathcal{Q} \times \mathcal{A}$ est appelé une *configuration* de T ; par convention on impose que

$$(1) \quad T(Q, a) = (Q, a, S) \iff Q = F$$

de sorte que les seules *configurations d'arrêt* de la machine sont de la forme (F, a) . Dans la suite $\mathfrak{T}(\mathcal{A})$ désigne l'ensemble (dénombrable) des machines de Turing dont l'alphabet d'exécution est $\mathcal{A}$.

Il y a beaucoup de définitions possibles (et équivalentes) de la machine de Turing : pour un approfondissement, citons l'introduction heuristique de Feynman [Fey99] ainsi que les présentations plus systématiques dans les livres de Minsky [Min67], Bianco [Bia79] ou encore chez Yablonski [Yab79]. On trouvera aussi un survol complet des apports scientifiques et mathématiques de Turing dans [Mar13] ; il y a aussi la belle biographie de Lassègue [Las98].

3. ASPECT FONCTIONNEL DES MACHINES DE TURING

La composition (fonctionnelle) des machines de Turing est définie de la façon suivante : pour $i = 0, 1$ soit T_i une machine de $\mathfrak{T}(\mathcal{A})$ définie avec un alphabet d'états internes $\mathcal{Q}_i$ et d'état initial et final I_i et F_i . En supposant de plus que $\mathcal{Q}_0 \cap \mathcal{Q}_1 = \emptyset$ (il est toujours possible de se ramener à cette situation), on définit la composée $T_1 \circ T_0$ comme la machine de $\mathfrak{T}(\mathcal{A})$ dont les états internes sont dans $\mathcal{Q} := \mathcal{Q}_0 \sqcup \mathcal{Q}_1$, d'état initial et final I_0 et F_1 , la

table de transition étant définie pour tout $(Q, a) \in Q \times \mathcal{A}$ en posant :

$$T_1 \circ T_0(Q, a) = \begin{cases} T_0(Q, a) & \text{si } Q \in Q_0 \setminus \{F_0\} ; \\ (I_1, a, S) & \text{si } Q = F_0 ; \\ T_1(Q, a) & \text{si } Q \in Q_1. \end{cases}$$

Pour $T \in \mathfrak{T}(\mathcal{A})$ une machine de Turing (d'états internes Q), l'itération (fonctionnelle) de T (composition successive avec elle-même) donne

$$T^0(Q, a) = (Q, a), \quad T^1(Q, a) = T(Q, a), \quad T^2(Q, a) = T \circ T(Q, a), \dots$$

Afin d'éviter toute ambiguïté d'écriture on supposera toujours que $Q \cup \{[,]\}$ est disjoint de $\mathcal{A}$. L'ensemble $\text{St}(T)$ des *états courants* de T est constitué des mots de la forme $w[Q]m$, où w et m sont dans $\mathcal{A}^*$ et $Q \in Q$. Ainsi, $\dots \star \star \star w m \star \star \star \dots$ est la suite des symboles écrit sur le ruban, la première lettre de m étant le symbole de la case lue par la tête de lecture/écriture (si m est le mot vide le symbole lu est $\star$) ; enfin, Q est l'état interne de la machine. Par abus de notation

$$T : \text{St}(T) \rightarrow \text{St}(T)$$

est l'application telle que $T(w[Q]m)$ soit l'état courant obtenu par application d'un cycle de T initialement dans l'état courant $w[Q]m$. L'ensemble des entrées (resp. sorties) de T est le sous-ensemble de $\text{St}(T)$ formé des états courants de la forme $w[I]m$ (resp. $w[F]m$). On note $\text{Out}(T)$ le sous ensemble de $\text{St}(T)$ qui sont des *sorties* (ou encore des *états terminaux*) de T : plus précisément,

$$w[Q]m \in \text{Out}(T) \iff T(w[Q]m) = w[Q]m \iff Q = F.$$

Par définition, le temps d'arrêt d'un état courant $w[Q]m \in \text{St}(T)$ est

$$\text{Stop}(T, w[Q]m) = \min \left\{ n \in \mathbb{N} ; T^n(w[Q]m) \in \text{Out}(T) \right\}.$$

avec $\text{Stop}(T, w[Q]m) = +\infty$ si $T^n(w[Q]m) \notin \text{Out}(T)$ pour tout $n \in \mathbb{N}$. On définit aussi l'application $T^* : \text{St}(T) \rightarrow \text{Out}(T) \cup \{\omega\}$ telle que

$$(2) \quad T^*(w[Q]m) = \begin{cases} T^{\text{Stop}(T, w[Q]m)}(w[Q]m) & \text{si } \text{Stop}(T, w[Q]m) < +\infty ; \\ \omega & \text{si } \text{Stop}(T, w[Q]m) = +\infty. \end{cases}$$

Remarque 3.1. Le fait que $\text{Stop}(\cdot, \cdot)$ soit définie par une minimisation déterminée par l'arrêt de la machine est à mettre en rapport avec le schéma de minimisation (introduit par Kleene dans [Kle36]) permettant de définir les fonctions récursives (voir l'identification des fonctions Turing-calculables et des fonctions récursives dans la démonstration du Théorème 5.4).

La fonction $\text{Stop}(\cdot, \cdot)$ permet aussi de retrouver une interprétation fonctionnelle de la composition des machines de Turing introduite en début de paragraphe ; en effet, étant données S et T deux machines dans $\mathfrak{T}(\mathcal{A})$, on a :

$$(3) \quad S \circ T^*(w[I]m) = \begin{cases} S^* \left(T^{\text{Stop}(T, w[I]m)}(w[I]m) \right) & \text{si } \text{Stop}(T, w[I]m) < +\infty ; \\ \omega & \text{si } \text{Stop}(T, w[I]m) = +\infty. \end{cases}$$

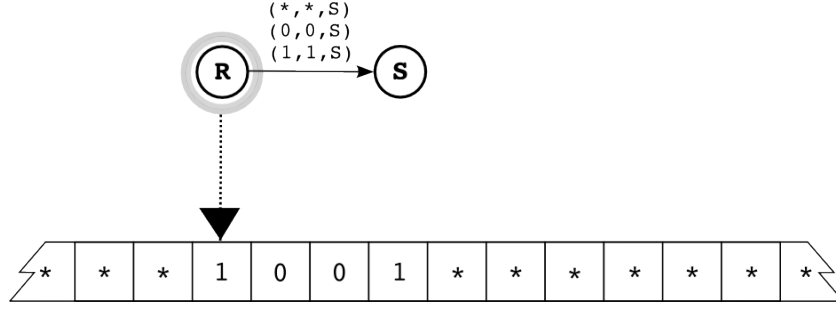


FIGURE 1. Représentation de Minsky d'une machine identité I (dans $\mathfrak{T}\{\star, 0, 1\}$) : ici on a $I^*([I]1001) = [F]1001$.

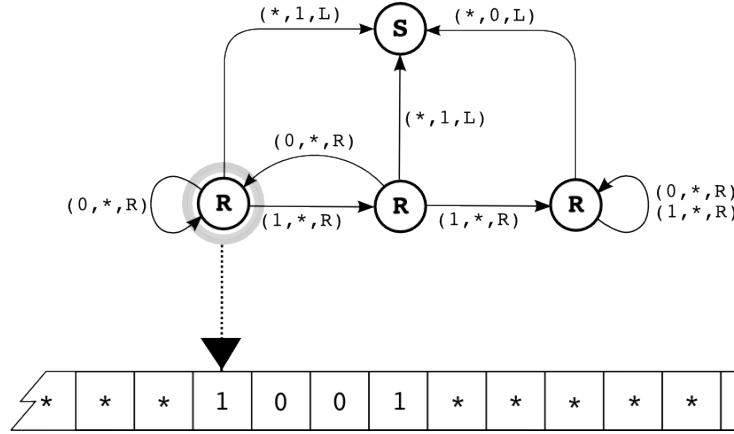


FIGURE 2. Représentation de Minsky d'une machine de Turing F reconnaissant les mots (de Fibonacci) qui ne sont pas factorisables par 11 : ici $F^*([I]1001) = [F]1$.

4. REPRÉSENTATION DES MACHINES DE TURING : GRAPHE DE MINSKY

Afin de donner une représentation graphique simple d'une machine de Turing de table de transition $T : \mathcal{Q} \times \mathcal{A} \rightarrow \mathcal{Q} \times \mathcal{A} \times \{L, S, R\}$, nous utiliserons *le graphe de Minsky* (voir [Min67]) : suivant la configuration de la machine on choisit (le plus judicieusement) une application $\Delta : \mathcal{Q} \ni Q \mapsto (Q, D_Q) \in \mathcal{Q} \times \{L, S, R\}$ qui fixe une direction privilégiée pour chaque état interne ; le graphe de Minsky est alors le graphe orienté dont l'ensemble des noeuds (resp. labels) est $\Delta(\mathcal{Q})$ (resp. $\mathcal{A} \times \mathcal{A} \times \{L, S, R\}$) et défini par l'application

$$G : \Delta(\mathcal{Q}) \times \Delta(\mathcal{Q}) \times (\mathcal{A} \times \mathcal{A} \times \{L, S, R\}) \rightarrow \{0, 1\}$$

telle que pour la transition $T((Q, a)) = (P, b, D)$,

$$G\left((Q, D_Q), (P, D_P), (a, b, D)\right) = 1 \iff (P, b, D) \neq (Q, b, D_Q).$$

Autrement dit, le graphe de Minsky ne représente pas une transitions qui ne changent ni l'état interne Q , ni le symbole de la case courante et pour laquelle la tête de lecture/écriture se déplace dans la direction privilégiée D_Q . En pratique, un noeud d'un graphe de Minsky ne portent que le symbole de sa direction privilégiée.

Donnons comme exemple une machine F permettant de reconnaître les mots de *Fibonacci* c'est-à-dire les mots binaires qui ne sont pas factorisables par 11. L'ensemble des

états internes de F est $\mathcal{Q} = \{I, U, V, F\}$ et l'alphabet d'exécution $\mathcal{A} = \{\star, 0, 1\}$. Le fonctionnement complet de la machine est donné par la table de transition suivante :

	$\star$	0	1
I	(F, 1, L)	(I, $\star$, R)	(U, $\star$, R)
U	(F, 1, L)	(I, $\star$, R)	(V, $\star$, R)
V	(F, 0, L)	(V, $\star$, R)	(V, $\star$, R)
F	(F, $\star$, S)	(F, 0, S)	(F, 1, S)

Ainsi on aura par exemple $F^*([I]1001) = [F]1$ (ce qui signifie que 1001 est un mot de Fibonacci) alors que $F^*([I]1011) = [F]0$ (ce qui signifie que 1011 n'est pas un mot de Fibonacci). Une représentation du graphe de Minsky de F (appliquée au mot 1001) est donnée dans la Figure 2.

5. LANGAGE RÉCURSIVEMENT ÉNUMÉRABLES ET LANGAGES RÉCURSIFS

Supposons $\{[,], |, \#\}$ disjoint de l'alphabet $\mathcal{A}$; la machine $G \in \mathfrak{T}(\mathcal{B})$, avec $\mathcal{B} \supset \mathcal{A} \cup \{[, \#]\}$, est un *générateur* du langage $\mathcal{L} \subset \mathcal{A}^*$, lorsqu'il existe une suite strictement croissante d'entiers $n_0, n_1, \dots$ (cette suite étant finie lorsque $\mathcal{L}$ est fini) et telle que

$$G^{n_k}([I]m_0|w_0\#) = [I]m_k|w_0\#w_1\#\dots\#w_k\#$$

et où $w_0, w_1, \dots$ forment une liste complète des mots dans $\mathcal{L}$. Le symbole $|$ (dont on s'assure qu'il ne possède qu'une seule occurrence au cours des calculs de G) délimite la zone d'affichage de la liste des mots de $\mathcal{L}$ (à droite) d'avec la zone de calcul (à gauche) où sont affichés les mots $m_0, m_1, \dots$ (ce sont des mot liés au fonctionnement de G).

Définition 5.1. Le langage $\mathcal{L} \subset \mathcal{A}^*$ est dit *rékursivement énumérable* s'il existe un générateur G qui engendre tous les mots de $\mathcal{L}$ (et uniquement ceux-là); il est dit *rékursif* s'il existe un générateur G qui engendre tous les mots de $\mathcal{L}$ dans l'ordre canonique sur $\mathcal{A}^*$.

Soit $\mathcal{L}$ un sous-langage de $\mathcal{A}^*$ et $w \in \mathcal{A}^*$; un problème classique consiste à déterminer si oui ou non w appartient à $\mathcal{L}$. Si le langage $\mathcal{L}$ est rékursif, on pourra répondre à cette question dans tous les cas en un temps fini (en général on ne connaît pas le temps d'attente). Par contre, si $\mathcal{L}$ n'est que rékursivement énumérable, et que w n'est pas dans $\mathcal{L}$, alors on a pas accès à cette information en temps fini.

Le *langage reconnu* (on dit aussi *accepté*) par une machine $T \in \mathfrak{T}(\mathcal{A} \sqcup \{\star\})$, noté $\text{Ackn}(T)$, est le sous-ensemble de $\mathcal{A}^*$ constitué des mots w pour lesquels le temps d'arrêt $\text{Stop}(T, [I]w)$ est fini ($\text{Ackn}(\cdot)$ vient du terme anglais *acknowledgement*).

Théorème 5.2. Soit $\mathcal{L} \subset \mathcal{A}^*$ un langage; alors (i) : $\mathcal{L}$ est rékursivement énumérable si et seulement si il est reconnu par une machine de Turing $T \in \mathfrak{T}(\mathcal{A} \cup \{\star\})$ et (ii) : $\mathcal{L}$ est rékursif si et seulement si $\mathcal{L}$ et $\mathcal{A}^* \setminus \mathcal{L}$ sont rékursivement énumérables.

Preuve. Exercice. □

Dans la suite $\mathcal{N}_2 = \{0\} \cup 1\{0,1\}^*$ désigne le langage des écritures binaires des nombres entiers naturels, de sorte qu'en utilisant (grossièrement) l'ordre naturel sur les

entiers $\mathcal{N}_2 = \{0, 1, 10, 11, 100, \dots\}$. L'ensemble $\mathfrak{T}(\mathcal{A} \sqcup \{\star\})$ des machines de Turing sur $\mathcal{A}$ étant dénombrable on peut l'écrire sous la forme d'une liste injective, soit encore :

$$T_0, T_1, T_{10}, T_{11}, T_{100}, \dots$$

Proposition 5.3. *Le langage*

$$(4) \quad \mathcal{D} := \left\{ w \in \mathcal{N}_2 ; w \notin \text{Ackn}(T_w) \right\} = \left\{ w \in \mathcal{N}_2 ; \text{Stop}(T_w, [\mathbf{I}]w) = +\infty \right\}.$$

n'est pas récursivement énumérable.

Preuve. Supposons par l'absurde $\mathcal{D}$ récursivement énumérable : comme $\mathcal{D} \subset \{0, 1\}^*$ et que $T_0, T_1, T_{10} \dots$ est la liste complète des machines de $\mathfrak{T}\{\star, 0, 1\}$, d'après le Théorème 5.2 il existe $w_0 \in \mathcal{N}_2$ tel que $\mathcal{D} = \text{Ackn}(T_{w_0})$. La contradiction vient du fait que $w_0 \in \mathcal{D}$ entraîne $w_0 \notin \text{Ackn}(T_{w_0}) = \mathcal{D}$ alors que $w_0 \in \mathcal{N}_2 \setminus \mathcal{D}$ entraîne $w_0 \in \text{Ackn}(T_{w_0}) = \mathcal{D}$. $\square$

Preuve (bis). Par l'absurde, supposons que $\mathcal{D} = \text{Ackn}(T_w)$ avec $w \in \mathcal{N}_2$; la contradiction vient du fait qu'on a :

$$w_A \in \mathcal{D} \Rightarrow \begin{cases} \text{Stop}(T_w, [\mathbf{I}]w) = \infty & \text{par définition de } \mathcal{D} \\ \text{Stop}(T_w, [\mathbf{I}]w) < \infty & \text{par définition de } T_w \end{cases}$$

et

$$w \in \mathcal{N}_2 \setminus \mathcal{D} \Rightarrow \begin{cases} \text{Stop}(T_w, [\mathbf{I}]w_A) < \infty & \text{par définition de } \mathcal{D} \\ \text{Stop}(T_w, [\mathbf{I}]w_A) = \infty & \text{par définition de } T_w \end{cases}$$

$\square$

6. MACHINE DE TURING UNIVERSELLE

Une machine de Turing est dite *universelle* si le ruban de lecture/écriture peut être programmée afin de *simuler* les calculs de n'importe quelle machine de Turing. En suivant la présentation guidée de Feynman³ dans [Fey99], la construction (abstraite) effective d'une machine universelle est un exercice abordable et même amusant ! Le théorème suivant, résume les propriétés de la machine universelle que nous utiliserons.

Théorème 6.1. *Il existe une machine de Turing $U \in \mathfrak{T}\{\star, 0, 1\}$ – dite universelle – associée à une application $(T, w[\mathbf{Q}]m) \mapsto \text{Pr}(T, w[\mathbf{Q}]m) \in \{\star, 0, 1\}^*$ définie pour tout $T \in \mathfrak{T}\{\star, 0, 1\}$ et tout $w[\mathbf{Q}]m \in \text{St}(T)$, et telle que pour tout couple $(T, w[\mathbf{Q}]m)$, il existe un rang $n \geq 1$ pour lequel*

$$(5) \quad U^n([\mathbf{I}]\text{Pr}(T, w[\mathbf{Q}]m)) = [\mathbf{I}]\text{Pr}(T, T(w[\mathbf{Q}]m)) ;$$

de plus, en posant $\text{Pr}(T, \omega) = \omega$, on a dans tous les cas

$$(6) \quad U^*([\mathbf{I}]\text{Pr}(T, w[\mathbf{Q}]m)) = [\mathbf{I}]\text{Pr}(T, T^*(w[\mathbf{Q}]m)) ;$$

d'autre part, il existe une machine $\Delta \in \mathfrak{T}\{\star, 0, 1\}$ (une décodeur) telle que

$$(7) \quad (\Delta \circ U)^*([\mathbf{I}]\text{Pr}(T, w[\mathbf{Q}]m)) = T^*(w[\mathbf{Q}]m).$$

3. Cette présentation est issue des discussions entre Minsky et Feynman à propos des machines et qui ont participé à jeter les bases de ce qu'on appelle maintenant l'*ordinateur quantique*.

7. L'AUTORÉFÉRENCE ET LE PROBLÈME DE L'ARRÊT

Soit $U \in \mathfrak{T}\{\star, 0, 1\}$ une machine universelle donnée par le Théorème 6.1. Toute machine $T \in \mathfrak{T}\{\star, 0, 1\}$ est associée au code $\text{Pr}(T, w[\mathbb{Q}]m) \in \{\star, 0, 1\}^*$ de simulation par U du calcul de T sur l'entrée $w[\mathbb{Q}]m$: si $T^k(w[\mathbb{Q}]m) = w_k[\mathbb{Q}_k]m_k$ alors il existe une suite d'entiers $0 = n_0, n_1, \dots$ strictement croissante telle que

$$U^{n_k}([I]\text{Pr}(T, w[\mathbb{Q}]m)) = [I]\text{Pr}(T, w_k[\mathbb{Q}_k]m_k).$$

Rappelons que $\mathcal{N}_2 := \{0\} \sqcup 1\{0, 1\}^*$ est l'ensemble des entiers binaires : ce langage est ordonné canoniquement en prenant $0 < 1$. Chacune machine $T \in \mathfrak{T}\{\star, 0, 1\}$ est affecté un entier binaire unique noté w_T qui est le *codage binaire (standard) du rang de $\text{Pr}(T, [I])$ dans l'ordre canonique sur $\{0, 1\}^*$* (le code $\text{Pr}(T, [I])$ décrit complètement, et de manière univoque, la table de transition de T) ; on notera aussi $T = T_{w_T}$, de sorte que $T_0, T_1, T_{10}, \dots$ est la liste complète des machines dans $\mathfrak{T}\{\star, 0, 1\}$. Cette numérotation des machines précise celle (plus vague) qui nous a permis de définir le langage $\mathcal{D}$ introduit en (4). Sachant (Proposition 5.3) que $\mathcal{D}$ n'est pas récursivement énumérable, le théorème suivant établit l'existence d'un langage strictement récursivement énumérable.

Proposition 7.1. *Le langage $\mathcal{N}_2 \setminus \mathcal{D} = \{w \in \mathcal{N}_2 ; \text{Stop}(T_w, [I]w) < +\infty\}$ est strictement récursivement énumérable (i.e. récursivement énumérable et non récursif).*

Preuve du Théorème 7.1. D'après la Proposition 5.3, nous savons que $\mathcal{D}$ n'est pas récursivement énumérable : il reste donc à démontrer que $\mathcal{N}_2 \setminus \mathcal{D}$ est récursivement énumérable. Par construction de la machine universelle U , le langage

$$\mathcal{E}_U := \left\{ \text{Pr}(T, w[\mathbb{Q}]m) ; T \in \mathfrak{T}\{\star, 0, 1\} \text{ et } w[\mathbb{Q}]m \in \text{St}(T) \right\}.$$

est récursif : le langage $\mathcal{N}_2$ étant lui aussi récursif, il en est de même de

$$\left\{ \text{Pr}(T_w, [I]w) ; w \in \mathcal{N}_2 \right\}.$$

Ainsi, il existe une machine S telle que $\text{Stop}(S, [I]w) = \infty$ lorsque $w \in \{0, 1\}^* \setminus \mathcal{N}_2$ et $S^*([I]w) = [F]\text{Pr}(T_w, [I]w)$, pour tout $w \in \mathcal{N}_2$. On obtient $\mathcal{N}_2 \setminus \mathcal{D} = \text{Ackn}(U \circ S)$ du fait que $\text{Stop}(U \circ S, [I]w) = \infty$, pour tout $w \in \{0, 1\}^* \setminus \mathcal{N}_2$ et que $\text{Stop}(T_w, [I]w) < \infty$ dès que $w \in \mathcal{N}_2$ (i.e. $w \in \mathcal{N}_2 \setminus \mathcal{D}$) si et seulement si $\text{Stop}(U \circ S, [I]w) < \infty$. □

Insistons sur le fait que l'existence d'un langage strictement récursivement énumérable est un sous-produit du *problème de l'arrêt* des machines de Turing. Nous donnons maintenant un deuxième argument pour le Théorème 7.1 : celui-ci est essentiellement basé sur le fait que la machine universelle U permet de conclure que

$$(8) \quad \mathcal{F}_U := \left\{ \text{Pr}(T, [I]w) ; T \in \mathfrak{T}\{\star, 0, 1\} \text{ et } w \in \{0, 1\}^* \text{ t.q. } \text{Stop}(T, [I]w) < \infty \right\}$$

est un langage récursivement énumérable. Le problème de l'arrêt porte sur la récursivité de $\mathcal{F}_U$, c'est-à-dire qu'il pose la question de l'existence d'une machine – appelée un *Oracle* – prédisant (pour tout $T \in \mathfrak{T}\{\star, 0, 1\}$ et tout $w \in \{0, 1\}^*$) si $\text{Stop}(T, [I]w) < \infty$.

Théorème 7.2. *Le langage $\mathcal{F}_U$ est strictement récursivement énumérable.*

Preuve. Supposons par l'absurde qu'il existe un oracle $O \in \mathfrak{T}\{\star, 0, 1\}$ décidant le problème de l'arrêt des machines dans $\mathfrak{T}\{\star, 0, 1\}$, c'est-à-dire tel que pour tout $T \in \mathfrak{T}\{\star, 0, 1\}$ et pour tout $w \in \{0, 1\}^*$,

$$O^*([I]\text{Pr}(T, [I]w)) = \begin{cases} [F]1 & \text{si } \text{Stop}(T, [I]w) < \infty \\ [F]0 & \text{si } \text{Stop}(T, [I]w) = \infty \end{cases}$$

Les langages $\mathcal{N}_2$ et $\mathcal{E}_U$ étant rékursifs, le langage $\{\text{Pr}(T_w, [I]w) ; w \in \mathcal{N}_2\}$ est lui aussi rékursif. Il existe donc une machine S pour laquelle $\text{Stop}(S, [I]w) = \infty$ lorsque $w \in \{0, 1\}^* \setminus \mathcal{N}_2$ et telle que $S^*([I]w) = [F]\text{Pr}(T_w, [I]w)$, pour tout entier binaire w . Par suite, pour tout $w \in \{0, 1\}^*$,

$$(O \circ S)^*([I]w) = \begin{cases} [F]1 & \text{si } w \in \mathcal{N}_2 \text{ et } \text{Stop}(T_w, [I]w) < \infty \\ [F]0 & \text{si } w \in \mathcal{N}_2 \text{ et } \text{Stop}(T_w, [I]w) = \infty \\ [F]0 & \text{si } w \in \{0, 1\}^* \setminus \mathcal{N}_2 \end{cases}$$

ce qui contredit le fait (Théorème 7.1) que $\mathcal{N}_2 \setminus \mathcal{D}$ n'est pas rékursif. □

La théorie des langages rékursivement énumérables est un point clef du dixième problème de Hilbert [Hil02] résolu par Yuri Matiyasevich (voir [Rob52, Dav53, DPR61, Mat70, DH73, Mat99]). Suivant une définition de Julia Robinson, une partie A de l'ensemble $\mathbb{N}$ des entiers naturels positifs ou nuls, est dite *diophantienne* s'il existe un polynôme $P \in \mathbb{Z}[X_0, \dots, X_N]$ tel que $n \in A$ si et seulement si l'équation $P(n, x_1, \dots, x_N) = 0$ possède une solution dans $\mathbb{Z}^{N+1}$. En d'autres termes, A est la projection sur $\mathbb{N}$ d'une courbe algébrique de $\mathbb{Z}^{N+1}$. Par exemple, l'ensemble $A = \{0, 1, 4, 9, \dots\}$ des carrés d'entiers est diophantien puisque $n \in A$ si et seulement si $P(n, n^2) = 0$ avec

$$P(X, Y) = X^2 - Y.$$

Pour $A \subset \mathbb{N}$ soit $\mathcal{L}_A$ le sous-ensemble de $\{0, 1\}^*$ des écritures binaires des éléments de A . L'ensemble A est alors dit rékursivement énumérable (resp. rékursif) si le langage $\mathcal{L}_A$ l'est. Si A est diophantien alors A est rékursivement énumérable. La réciproque répond à la question de Hilbert portant sur le calcul des solutions des équations diophantiennes⁴.

Théorème 7.3. [Robinson-Matijasevic] Une partie de $\mathbb{N}$ est diophantienne si et seulement si elle est rékursivement énumérable.

En particulier, il existe des parties diophantiennes de $\mathbb{N}$ qui sont strictement rékursivement énumérable : cela rend le calcul systématique (mécanique) des solutions d'équations diophantiennes impossible (voir [Oli14, § 5]).

8. COMPLEXITÉ DE KOLMOGOROV

Dans ce paragraphe, tout nombre entier naturel n est identifié à son développement binaire, c'est-à-dire à un mot du langage $\mathcal{N}_2 = \{0\} \cup 1\{0, 1\}^*$; dans la suite, $\ell(w)$ désigne

4. Est-il possible de trouver une procédure *mécanique* permettant de trouver les solutions des équations diophantiennes ? D'après le Théorème de Robinson-Matijasevic, la réponse est non.

la longueur d'un mot de w relativement à son alphabet de référence⁵ : en particulier,

$$\log_2 n \leq \ell(n) \leq 1 + \log_2(\max\{1, n\}).$$

Admettons provisoirement l'existence d'une famille $\{K^{(n)} : \mathbb{N}^n \rightarrow \mathbb{N} ; n \geq 1\}$ t.q.

(K0) : il existe une infinité d'entiers n t.q. $K^{(1)}(n) \geq \ell(n)$ (n est dit K -aléatoire) ;

(K1) : si $f : \mathbb{N}^p \rightarrow \mathbb{N}^q$ est calculable alors il existe $C_f > 0$ t.q.

$$K^{(q)}(f(x_1, \dots, x_p)) \leq 4(\ell(x_1) + \dots + \ell(x_p)) + C_f ;$$

Théorème 8.1 (Euclide). *Il existe une infinité de nombres premiers.*

Preuve (Chaitin-Kolmogorov [Kol65][Cha66, Cha75]). D'après **(K0)**, il existe une suite strictement croissante $x_1, x_2, \dots$ d'entiers K -aléatoires, de sorte que pour tout rang k ,

$$(9) \quad K^{(1)}(x_k) \geq \ell(x_k).$$

Supposons qu'il existe un nombre fini de nombre premiers, soient $p_1, \dots, p_N$ et écrivons

$$x_k = p_1^{\alpha_1(x_k)} \dots p_N^{\alpha_N(x_k)} =: f(\alpha_1(x_k), \dots, \alpha_N(x_k))$$

la décomposition de x_k en facteurs premiers (ici les exposants $\alpha_i(x_k) \in \mathbb{N}$ sont des entiers positifs où nuls). En particulier, du fait que $x_k \geq 2^{\alpha_1(x_k) + \dots + \alpha_N(x_k)}$, on tire l'inégalité

$$(10) \quad \log_2 x_k \geq \alpha_1(x_k) + \dots + \alpha_N(x_k).$$

L'application $f : (a_1, \dots, a_N) \mapsto p_1^{a_1} \dots p_N^{a_N}$ étant calculable, avec **(K1)** il vient :

$$\begin{aligned} K^{(1)}(x_k) &= K^{(1)}(f(\alpha_1(x_k), \dots, \alpha_N(x_k))) \\ &\leq 4[\ell(\alpha_1(x_k)) + \dots + \ell(\alpha_N(x_k))] + C_f \\ &\leq 4[\log_2(\max\{1, \alpha_1(x_k)\}) + \dots + \log_2(\max\{1, \alpha_N(x_k)\})] + 4N + C_f \\ &\leq 4\log_2[\alpha_1(x_k) + \dots + \alpha_N(x_k)] + 4N + C_f ; \end{aligned}$$

en combinant (9) et (10) on obtient la suite – contradictoire – d'inégalités

$$\log_2 x_k \leq \ell(x_k) \leq K^{(1)}(x_k) \leq 4\log_2(\log_2 x_k) + 4N + C_f.$$

□

La notion de *complexité algorithmique* de Kolmogorov [Kol65], permet la construction effective d'une famille d'applications $K^{(n)}$ ($n \geq 1$) satisfaisant **(K0)**, **(K1)** et **(K2)**. Pour $T \in \mathfrak{T}\{\star, 0, 1\}$ la T -complexité d'un mot $w \in \mathcal{A}^*$ pour $\emptyset \neq \mathcal{A} \subset \{\star, 0, 1\}^*$ est

$$K_T(w) = \min \left\{ \ell(m) ; m \in \mathcal{A}^* \text{ et } T^*([I]m) = [F]w \right\}$$

avec la restriction que $K_T(w) = +\infty$ s'il n'existe pas de mot binaire m tel que $T^*([I]m) = [F]w$. Un premier moyen pour contourner la dépendance relativement à la machine T (qui entraîne en particulier que $K_T(w)$ peut-être infini) est de définir le minimum $\underline{K}(w)$ des $K_T(w)$ pour T décrivant l'ensemble des machines dans $\mathfrak{T}\{\star, 0, 1\}$. En considérant la machine identité, il est évident que $\underline{K}(w) \leq \ell(w)$. Cette dernière remarque mène au concept de *nombre compressible* : pour $0 < \gamma < 1$ on dit que w est γ -compressible si $\underline{K}(w) \leq \gamma\ell(w)$. Pour un tel w il existe donc une machine T telle $K_T(w) \leq \gamma\ell(w)$; ce critère de

5. Nous utilisons $\ell(w)$ et non $|w|$ – comme d'habitude – afin d'éviter de possibles confusions avec la valeur absolue.

compressibilité ne tient pas compte de la structure interne de la machine⁶ T , ni de la longueur d'un calcul du type $T^*([I]m) = [F]w$. Un moyen pour prendre en compte la complexité des machines (mais pas de la longueur du calcul⁷) est de définir un analogue de $\underline{K}(w)$ au moyen d'une machine universelle (voir Définition 8.2 ci-dessous).

Dans l'idée de Kolmogorov, le mot w est (*algorithmiquement*) *aléatoire* s'il est *peu* compressible. Dans ce paragraphe $U \in \mathfrak{T}\{\star, 0, 1\}$ est une machine universelle fixée donnée par le Théorème 6.1. En particulier U permet de simuler les calculs de toute machine $T \in \mathfrak{T}\{\star, 0, 1\}$. Nous utiliserons aussi le décodeur Δ défini en (7) et $U_0 := \Delta \circ U$.

Définition 8.2. (i) : La complexité de Kolmogorov (relative à la machine universelle $U_0 = \Delta \circ U$) d'un mot $w \in \mathcal{A}^*$ pour $\emptyset \neq \mathcal{A} \subset \{\star, 0, 1\}^*$ est

$$K_{U_0}(w) = \min \left\{ \ell(m) ; m \in \mathcal{A}^* \text{ et } U_0^*([I]m) = w \right\} \quad (< +\infty) ;$$

(ii) : pour $n \geq 1$ la complexité de Kolmogorov d'un n -uplet $(x_1, \dots, x_n) \in \mathbb{N}^n (\equiv \mathcal{N}_2^n)$ est

$$K^{(n)}(x_1, \dots, x_n) := K_{U_0}(x_1 \star \dots \star x_n).$$

Pour $w \in \{\star, 0, 1\}^*$ donné, il existe un couple optimal machine/mot (T, m) tel que $T^*([I]m) = w$ et avec $\underline{K}(w) = K_T(w) = \ell(m)$. Comme $U_0^*([I]\text{Pr}(T, [I]m)) = T^*([I]m) = w$, il vient $K_{U_0}(w) \leq \ell(\text{Pr}(T, [I]m))$: il est évident que $\underline{K}(w) \leq K_{U_0}(w)$ et raisonnable d'imaginer que $K_{U_0}(w)$ et $\underline{K}(w)$ sont – en un sens – comparables. En fait, si U est bien construite, U_0 doit pouvoir engendrer w de manière optimale en ce sens que U_0 ne peut pas faire beaucoup moins bien qu'une machine non universelle T et optimisée pour engendrer w . Le Lemme suivant précise cette remarque.

Lemme 8.3. Il existe une machine universelle U (au sens du Théorème 6.1) optimale pour la complexité de Kolmogorov en ce sens que pour toute machine $T \in \mathfrak{T}\{\star, 0, 1\}$ il existe $\theta \geq 1$ (dépendant de U) et $C_T > 0$ (dépendant de U et de T) t.q. pour tout mot $w \in \{\star, 0, 1\}^*$,

$$K_{U_0}(w) \leq \theta K_T(w) + C_T.$$

Preuve heuristique. Supposons que $K_T(w) < +\infty$ et soit m minimal t.q. $T^*([I]m) = [F]w$; en particulier cela signifie que $\ell(m) = K_T(w)$. Par définition de la machine $U_0 = \Delta \circ U$ il vient $U_0^*([I]\text{Pr}(T, [I]m)) = [F]w$ et donc $K_{U_0}(w) \leq \ell(\text{Pr}(T, [I]m))$. Le détail de la construction de U montre qu'il est possible d'assurer que $\ell(\text{Pr}(T, [I]m))$ est une fonction affine de $\ell(m)$: pour une bonne machine universelle U , il existe alors deux constantes $\theta > 1$ (dépendant de U_0) et $C_T > 0$ (dépendant de U_0 et de T) telles que $\ell(\text{Pr}(T, [I]m)) \leq \theta \ell(m) + C_T$. Les définitions de θ et C_T doivent prendre en compte le système de simulation de U (implémenté sur le ruban de U) ainsi que la description de T dans le programme de simulation de T (inscrit sur le ruban de U) ; il est possible de construire U t.q. $\theta = 2$. □

6. La machine T en question peut donc être très *complexe* et difficile à trouver : c'est tout le problème de la *compression des données*.

7. Pour prendre en compte la longueur du calcul $T^*([I]m) = [F]w$ on pourrait par exemple considérer

$$\sum_{k=0}^{\text{Stop}(T, [I]m)} \ell(T^k([I]m)).$$

Proposition 8.4. *Les applications $K^{(n)}$ ($n \geq 1$) vérifient **(K0)**; plus précisément, pour tout entier $n \geq 1$, l'ensemble $\{0, 1\}^n$ contient au moins un mot K -aléatoire.*

Preuve. Le nombre de mots dans $\{0, 1\}^*$ de longueur au plus $n - 1$ est $1 + 2 + \dots + 2^{n-1} = 2^n - 1$. Par suite il existe au moins un mot x , parmi les 2^n mots de $\{0, 1\}^n$, pour lequel $U_0^*([I]m) = [F]x$ entraîne $\ell(m) \geq n$: cela signifie que $K^{(1)}(x) \geq \ell(x) = n$. □

Remarque 8.5. (1) : La Proposition 8.4 est une porte ouverte sur le monde de la théorie des générateurs de nombres aléatoires (voir [BH10]).

(2) : Soit $w \in \{\star, 0, 1\}$ et soit (T, m) un couple machine/mot optimal qui engendre w , i.e. $T^*([I]m) = [F]w$ et $\underline{K}(w) = \ell(m)$. Alors $U_0^*([I]\text{Pr}(T, [I]m)) = [F]w$ et donc $K_{U_0}(w) \leq \ell(\text{Pr}(T, [I]m))$. Il est troublant d'imaginer l'existence d'un mot $x \in \{\star, 0, 1\}$ (non prévu par la syntaxe de programmation de U_0) t.q. $U_0^*([I]x) = [F]w$: cette possibilité (fonctionnement de U or du ce qui est prévu) est d'autant plus troublante qu'il est possible que $\ell(x) \leq \ell(\text{Pr}(T, [I]m))$.

Proposition 8.6. *La condition **(K1)** est vérifiée, c'est-à-dire, que pour toute fonction $f : \mathbb{N}^p \rightarrow \mathbb{N}^q$ Turing-calculable, il existe une constante C_f (dépendant de la machine universelle utilisée) t.q.*

$$K^{(q)}(f(x_1, \dots, x_p)) \leq \theta^2(\ell(x_1) + \dots + \ell(x_p)) + C_f.$$

(et où la constante $\theta \geq 1$ donnée par le Lemme 8.3 peut prendre la valeur 2).

Preuve. Supposons que $f(x_1, \dots, x_p) = (y_1, \dots, y_q)$ et soit $m \in \{\star, 0, 1\}$ la plus petite entrée t.q. $U_0^*([I]m) = [F]x_1 \star \dots \star x_p$; alors

$$\ell(m) = K^p(x_1, \dots, x_p) = K_{U_0}(x_1 \star \dots \star x_p)$$

et en notant T_{Id} la machine identité de $\mathfrak{T}\{\star, 0, 1\}$ le Lemme 8.3 donne :

$$(11) \quad \ell(m) \leq K_{T_{Id}}(x_1 \star \dots \star x_p) + C_{T_{Id}} = \theta(\ell(x_1) + \dots + \ell(x_p)) + p + C_{T_{Id}}.$$

Puisque f est calculable, il existe un machine T_f telle que $T_f^*([I]x_1 \star \dots \star x_p) = [F]y_1 \star \dots \star y_q$. Par composition des machines $(T_f \circ U_0)^*([I]m) = [F]y_1 \star \dots \star y_q$ et par suite

$$K_{T_f \circ U_0}(y_1 \star \dots \star y_q) \leq \ell(m) ;$$

avec la machine $T_f \circ U_0$ et la constante $C_{T_f \circ U_0}$ donnée par le Lemme 8.3

$$\begin{aligned} K^{(q)}(f(x_1, \dots, x_p)) &= K_{U_0}(y_1 \star \dots \star y_q) \\ &\leq \theta K_{T_f \circ U_0}(y_1 \star \dots \star y_q) + C_{T_f \circ U_0} \\ &\leq \theta \ell(m) + C_{T_f \circ U_0} \\ &= \theta^2(\ell(x_1) + \dots + \ell(x_p)) + (\theta(p + C_{T_{Id}}) + C_{T_f \circ U_0}). \end{aligned}$$

□

RÉFÉRENCES

- [BH10] L. Bienvenu and M. Hoyrup. Une brève introduction à la théorie effective de l'aléatoire. *La Gazette des mathématiciens (SMF)*, 123 :35–47, 2010.
- [Bia79] E. Bianco. *Informatique fondamentale : de la machine de Turing aux ordinateurs modernes*. Basel, Boston, Stuttgart ISR 70, 1979.
- [Cha66] G. Chaitin. On the lenght of programs for computing finite binary sequences. *J. of Ass. For Computing Machinery*, 13 :547–569, 1966.

- [Cha75] G. Chaitin. A theory of program size formally identical to information theory. *J. of Ass. For Computing Machinery*, 22 :329–340, 1975.
- [Dav53] M. Davis. Arithmetical problems and recursively enumerable predicates. *Journal of Symbolic Logic*, 18-(1) :33–41, 1953.
- [DH73] M. Davis and R. Hersh. Hilbert’s tenth problem. *Scientific American*, 229 :84–91, 1973.
- [DPR61] M. Davis, H. Putnam, and J. Robinson. The decision problem for exponential Diophantine equations. *Annals of Mathematics, Second Series*, 74-(3) :425–436, 1961.
- [Fey99] R. Feynman. *Lectures on Computation*. Penguin Books Ltd (New edition), 1999.
- [Hil02] D. Hilbert. Lecture delivered before the International Congress of Mathematicians at Paris in 1900 by Professor David Hilbert, (translated into english by Dr. Maby Winton Newson, with the author’s permission). *Bulletin of the American Mathematical Society*, 8 :437–479, 1902.
- [Kle36] S. Kleene. General recursive functions of natural numbers. *Mathematische Annalen*, 112 :727–729, 1936.
- [Kol65] A. Kolmogorov. Three approaches to quantitative definition of information. *Information and Control*, 1 :1–7, 1965.
- [Las98] J. Lassègue. *Turing*. Les Belles Lettres, Paris, 1998.
- [Mar13] M. Margenstern. Ce qu’Alan Turing nous a laissé. *La Gazette des mathématiciens (SMF)*, 135 :17–31, 2013.
- [Mat70] Y. Matiyasevich. Enumerable sets are Diophantine (in Russian). *Doklady Akademii Nauk SSSR* 191 : 279–282. *English translation in Soviet Mathematics*, 11-(2) :354–357, 1970.
- [Mat99] Yuri Matiyasevich. *Le dixième problème de Hilbert : que peut-on faire avec les équations diophantiennes ? La recherche de la vérité* (M. Serfati ed.). A.C.L. Paris, 1999.
- [Min67] Marvin Lee Minsky. *Computation : Finite and Infinite Machines*. Prentice Hall, 1967.
- [Oli14] E. Olivier. Qu’est-ce-qu’une machine (II/III). *BIAA*, 98 :45–56, 2014.
- [Rob52] J. Robinson. Existential definability in arithmetic. *Transactions of the American Mathematical Society*, 72-(3) :437–449, 1952.
- [Tur36] A. Turing. On Computable Numbers, with an application to the Entscheidungsproblem (correction ibid. (1967) 43, p. 544–546). *Proc. Lond. Math. Soc.*, (2)-42 :230–265, 1936.
- [Yab79] S. Yablonski. *Introduction au machines discrètes*. MIR - Moscou (traduit du russe par D. Embarek), 1979.