Le théorème de cardinalité

Travail de rédaction LFCC

Miguel ACOSTA

6 janvier 2011

1 Machines avec oracle

On fixe l'alphabet $\{0,1\}$. On rappelle que l'ensemble des langages décidables, ou récursifs, de $\{0,1\}^*$ est l'ensemble des langages reconnus par une machine de Turing qui s'arrête sur chaque calcul qu'elle effectue. On identifiera par la suite $\{0,1\}^*$ et $\mathbb{N}$ par l'écriture en base 2. On parle alors de sous-ensembles de $\mathbb{N}$ décidables ou récursifs. Une fonction est récursive, ou calculable, s'il existe une machine de Turing qui s'arrête sur tous les calculs et qui la calcule. Finalement, un langage est récursivement énumérable s'il est le langage reconnu par une machine de Turing.

Dans la suite, on considère uniquement des machines déterministes.

On s'interesse à élargir la classe des langages décidables de $\{0,1\}^*$. Pour le faire, on utilise des machines avec oracle :

Définition 1.1. Soit $B \subseteq \mathbb{N}$. Une machine à oracle $M^{(B)}$ est une machine de Turing à plusieurs bandes, dont une bande "oracle", qui est initialisée avec $\chi_B(i)$ sur la $i^{\text{ème}}$ position et sur laquelle la machine n'écrit pas.

En d'autres mots, $M^{(B)}$ est une machine de Turing qui, au cours d'un calcul, répond instantanément pour les instances de B.

Ayant les machines avec oracle, on peut s'interesser aux fonctions calculables par ces machines :

Définition 1.2. Soit $B \subseteq \mathbb{N}$. Une fonction f est calculée avec l'oracle B s'il existe une machine avec oracle $M^{(B)}$ qui, avec l'entrée x, écrit sur sa bande de sortie $f(x)$ et arrive dans un état final.

On aura alors des classes plus larges que celle des langages décidables : avec un ensemble B fixé, on peut considérer la classe des fonctions calculables par une machine $M^{(B)}$. On peut aussi étudier les langages L décidables par une machine à oracle $M^{(B)}$. Une telle classe contient strictement les langages décidables si et seulement si B n'est pas décidable. De plus, par un argument de dénombrabilité, on voit que, quelque soit B, il existe un langage qui n'est reconnu par aucune machine $M^{(B)}$. Ceci permet alors de donner une hiérarchie sur les langages de $\{0,1\}^*$.

2 La conjecture de cardinalité de Beigel

On peut cependant se demander dans quelle mesure la présence de l'oracle pour décider un problème détermine l'indécidabilité du problème en question. À ce sujet, Beigler a donné une conjecture, qui dit que, si on identifie le coût d'un calcul au nombre d'appels de l'oracle, il y a un

coût minimal pour décider, à l'aide d'une machine $M^{(B)}$, un langage non récursif A. C'est-à-dire, étant donné un entier m et un ensemble A, si une machine à oracle $M^{(B)}$ qui décide le nombre d'élements dans A parmi 2^m entiers distincts avec un coût faible ($\leq m$), alors A est décidable.

Définition 2.1. Soient $A \subseteq \mathbb{N}$ et $n \in \mathbb{N}$. On note

$$\#_n^A : \begin{array}{l} \mathbb{N}^n \to \mathbb{N} \\ (x_1, \ldots, x_n) \mapsto Card\{i \in [|1, n|] \mid x_i \in A\} = \sum_{i=0}^{n} \chi_A(x_i) \end{array} .$$

Conjecture 2.2 (Conjecture de Cardinalité de Beigel)**.** Soient $A, B \subseteq \mathbb{N}$ et $n \in \mathbb{N}$. Si $\#_{2^n}^A$ est calculable par une machine de Turing $M^{(B)}$ en faisant au plus n appels à l'oracle, alors A est récursif.

Cela veut dire que, si on n'utilise l'oracle qu'un nombre assez faible de fois por calculer le nombre d'éléments dans A dans un 2^n-uplet, alors A est récursif.

On remarque que, dans la conjecture, n'importe quelle valeur de n convient, et on ne fait pas d'hypothèses sur B ce qui fait un résultat assez général. On se propose de montrer cette conjecture en montrant un résultat plus général : le théorème de cardinalité. Pour le faire, il faut introduire d'abord quelques notations.

3 Notations

On rappelle l'existence d'une machine de Turing universelle M, et d'un codage récursif des machines de Turing tel que, pour tout indice i et tout mot $w \in \{0,1\}^*$, M reconnaît le code du couple (i, w) si et seulement si la machine d'indice i reconnaît w. Ce fait nous sert à énumérer les parties récursives de $\mathbb{N}$.

Définition 3.1. Soit $i \in \mathbb{N}$. On note W_i le langage (identifié à une partie de $\mathbb{N}$) reconnu par la machine de Turing d'indice i.

On veut maintenant donner un codage pour les arbres en termes de mots sur $\{0,1\}^*$. Il faut fixer d'abord les notations, puis donner une définition naturelle d'un arbre binaire (fini ou infini) dans ce langage.

Définition 3.2.
1. On note ϵ le mot vide et, si $s \in \{0,1\}^*$, $|s|$ dénote la longueur de s, de sorte que $|\epsilon| = 0$.
2. Si $s, t \in \{0,1\}^*$ on note $s \sqsubseteq t$ si s est un préfixe de t, et $s * t$ la concaténation de s et t.
3. Si $s \in \{0,1\}^*$ et $n \leq |s|$, $s(n)$ dénote la $n^{\text{ème}}$ lettre de s.

Définition 3.3. Un arbre est un sous-ensemble T de $\{0,1\}^*$ stable par préfixes. $t \in T$ est un noeud de T, $u \in \{0,1\}^{\mathbb{N}}$ est une branche de T si tout segment initial de u est dans T.

Définition 3.4. On note, pour $k \in \mathbb{N}$, B_k l'arbre complet de hauteur k.

On utilisera dans la suite les plongements d'un arbre T dans un arbre $\tilde{T}$. C'est-à-dire, on prend une partie d'un arbre T isomorphe à un arbre $\tilde{T}$ par $s \mapsto \tilde{s}$ au sens où s est un fils de t si et seulement si $\tilde{s}$ est un descendant de $\tilde{t}$. Ou encore, qu'en effaçant des noeuds de T on obtient $\tilde{T}$.

Définition 3.5. Soient $k \in \mathbb{N}$ et T un arbre. Un plongement de B_k dans T est une fonction $f : B_k \to T$ telle que, pour tout $s \in B_k$ tel que $|s| < k$ on ait $f(s)*0 \sqsubseteq f(s*0)$ et $f(s)*1 \sqsubseteq f(s*1)$. B_k est plongeable au-dessus de $e \in T$ s'il existe un plongement f de B_k dans T tel que $f(\epsilon) = e$

Exemple 3.6. Dans la figure 1, on voit un plongement de B_3 dans un arbre T : en effaçant des noeuds de T on obtient une copie de B_3 au dessus de ϵ. De plus, on a des plongements de B_2 au-dessus de 0 (brance gauche) et 11 (branche droite).

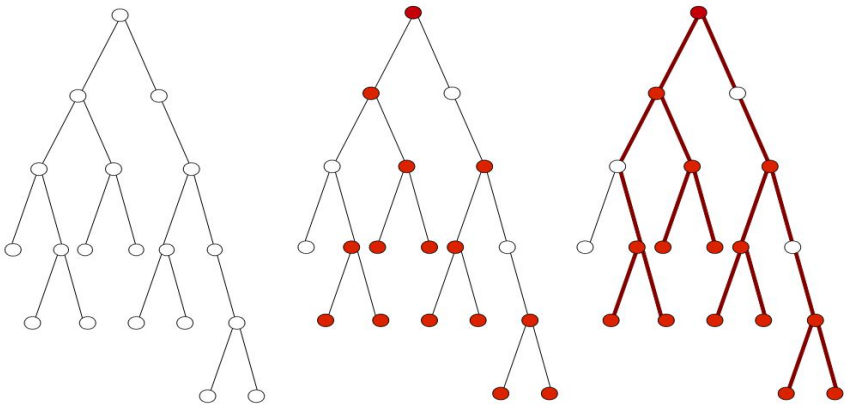

FIGURE 1 – Plongement de B_3 dans T

Définition 3.7. Soit T un arbre. On appelle rang de T le sup, dans $\mathbb{N} \cup \{\infty\}$, de l'ensemble des $n \in \mathbb{N}$ tels que B_n se plonge dans T. Si T est de rang fini, on le note $\text{rk}(T)$.

Exemple 3.8. L'arbre de la figure 1 est fini, donc de rang fini. Dans le cas de la figure, $\text{rk}(T) = 3$

Exemple 3.9. On considère l'arbre suivant : $T = \{s \in \{0,1\}^* \mid s(0) = 0, \forall n \in \mathbb{N}\ s(n) = s(n+1) \implies |s| = n+1\} \cup \{\epsilon\}$. Alors T est de rang 1. On voit bien, dans la figure 2, qu'on peut plonger B_1 mais pas B_2 dans T.

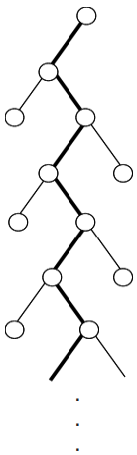

FIGURE 2 – Un arbre infini de rang 1

4 Le Théorème de Cardinalité

On montre ici un résultat plus général que la conjecture de Beigel, qui est dû à Kummer.

Théorème 4.1 (Kummer 1992)**.** *Soient $A \subseteq \mathbb{N}$ et $m \geq 1$. On suppose qu'il existe une fonction récursive $g : \mathbb{N}^m \to \mathbb{N}$ telle que, pour tout m-uplet $(x_1, \ldots, x_m)$ d'entiers naturels distincts :*

1. $W_{g(x_1,\ldots,x_m)} \subsetneq [|0, m|]$
2. $\#_m^A(x_1, \ldots, x_m) \in W_{g(x_1,\ldots,x_m)}$

Alors A est récursif.

Remarque 4.2. En d'autres termes, si on peut construire une machine de Turing calculant une fonction $\psi : \mathbb{N}^m \to \mathbb{N}^m$ telle que pour tout $(x_1, \ldots, x_m) \in \mathbb{N}^m$ on ait $\#_m^A(x_1, \ldots, x_n) \in \psi(x_1, \ldots, x_m)$ alors A est récursif.

De manière équivalente, on a un résultat un peu plus général : Si $f : \mathbb{N}^m \to \mathbb{N}$ est telle qu'on peut construire une machine de Turing calculant une fonction $\phi : \mathbb{N}^m \to \mathbb{N}^m$ telle que pour tout $(x_1, \ldots, x_m) \in \mathbb{N}^m$ on ait $f(x_1, \ldots, x_n) \in \phi(x_1, \ldots, x_m)$ alors f est récursive.

En d'autres mots, si de manière récursive on peut touver m élements parmi lesquels se trouve $f(x_1, \ldots, x_m)$ pour tout m-uplet, alors f est récursive.

Remarque 4.3. Le théorème implique que la conjecture de cardinalité est vraie.

Démonstration. Supposons le résultat du théorème vrai. Soient $A, B \subseteq \mathbb{N}$ et $n \in \mathbb{N}$ tels que $\#_{2^n}^A$ est calculable par une machine de Turing $M^{(B)}$ en faisant au plus n appels à l'oracle. Posons $m = 2^n$.

Pour chaque $w \in \{0,1\}^n$, on définit une machine de Turing M_w, calculant h_w, qui simule $M^{(B)}$ mais qui, au $k^{\text{ème}}$ appel de l'oracle, répond par $w(k)$. Il existe donc une fonction récursivement énumérable g telle que $W_{g(x_1,\ldots,x_m)} = \{h_w(x_1, \ldots, x_m) \mid w \in [|0, m|]\} \cap [|0, m|]$.

Or, comme une des séquences de réponses est celle donnée par B, pour chaque entrée $(x_1, \ldots, x_m)$ il existe $w \in \{0,1\}^n$ tel que $\#_{2^n}^A(x_1, \ldots, x_m) = h_w(x_1, \ldots, x_m)$.

On a alors, pour tout m-uplet $(x_1, \ldots, x_m)$, que $W_{g(x_1 \ldots x_m)} \subsetneq [|0, m|]$ (par cardinalité) et $\#_m^A(x_1, \ldots x_m) \in W_{g(x_1 \ldots x_m)}$.

D'après le théorème de cardinalité, A est récursif. □

5 Lemmes préliminaires

Pour montrer le théorème, on a besoin de trois résultats intermédiaires. On montre d'abord qu'un arbre récursivement énumérable de rang fini a toutes ses branches récursives. Ensuite, à l'aide d'un résultat combinatoire sur le coloriage d'un arbre complet on donne une condition sur un arbre qui servira à borner son rang. On appliquera finalement ces résultats sur un arbre défini à partir de la fonction g de l'énoncé du théorème.

Lemme 5.1. *Soit T un arbre récusivement énumérable de rang fini. Alors toute branche de T est récursive.*

Démonstration. Soit T un tel arbre. Soit t une branche de T. Soit k_0 le max des entiers n tels que B_n est plongeable au-dessus de e pour tout $e \sqsubseteq t$. On a ainsi $k_0 \leq rk(T)$. Soit e_0 un noeud de t tel que B_{k+1} ne soit pas plongeable au-dessus de e_0.

On a un algorithme déterministe pour calculer la fonction t :

– Si $x \leq |e_0|$, la valeur $t(x)$ est stockée dans un tableau fini. Il suffit donc de le consulter.

– Si $x > |e_0|$, comme T est récursivement énumérable, on peut lancer un énumérateur de T. À chaque étape, sur les noeuds déjà énumérés, on essaie de trouver un plongement f de B_{k_0} au-dessus de e_0 tel que sa racine vérifie $|f(\epsilon)| > x$. Par le choix de k_0, cette étape termine. On renvoie alors $f(\epsilon)(x)$.

Il reste à monter que l'algorithme est correct. Soit $x \in \mathbb{N}$. Il suffit de prouver que le plongement f trouvé vérifie $f(\epsilon) \sqsubseteq t$.

Supposons, par l'absurde, que $f(\epsilon) \not\sqsubseteq t$. Soit d le plus long préfixe commun de t et $f(\epsilon)$. On a clairement $e_0 \sqsubseteq d$. Par le choix de k_0, il existe un plongement $g_0 : B_{k_0} \to T$ au-dessus de $d * t(|d|)$, et, par l'algorithme, il existe un plongement $g_1 : B_{k_0} \to T$ au-dessus de $d * (1 - t(|d|))$.

Donc $g : \begin{array}{l} \epsilon \mapsto e_0 \\ 0 * s \mapsto g_0(s) \\ 1 * s \mapsto g_1(s) \end{array}$ est un plongement de B_{k_0+1} dans T au-dessus de e_0, comme on voit dans la figure 3 Absurde.

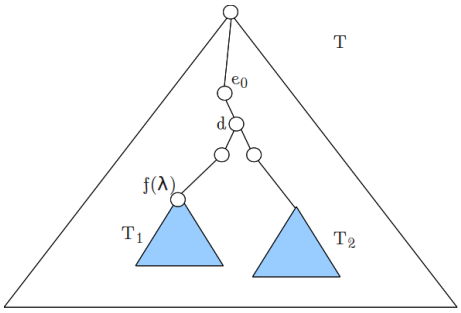

FIGURE 3 – Plongement de B_{k_0+1} au-dessus de e_0

□

Lemme 5.2. *Soit $k \in \mathbb{N}^*$. Pour tout 2-coloriage de B_{2k}, il existe un plongement g de B_k dans B_{2k} tel que tous les noeuds de $g(B_k)$ aient la même couleur.*

Démonstration. Montrons, par récurrence sur $m + n$, que, pour tout coloriage en rouge et bleu de B_{m+n} il existe soit un plongement de B_m où tous les noeuds sont rouges, soit un plongement de B_n où tous les neuds sont bleus.

– Si $m + n = 1$ le résultat est clair.
– Soit $k \in \mathbb{N}$ tel que, pour tout coloriage de B_k et pour tous m, n tels que $m + n = k$ il existe un plongement rouge de B_m ou un plongement bleu de B_n. Soient $m, n \in \mathbb{N}$ tels que $m + n = k + 1$. On considère un coloriage de B_{k+1}
  – Si $m = 0$ ou $n = 0$, le résultat est clair. On suppose $m > 0$ et $n > 0$.
  – On considère $f_0 : \begin{array}{l} B_k \to B_{k+1} \\ s \mapsto 0 * s \end{array}$ et $f_1 : \begin{array}{l} B_k \to B_{k+1} \\ s \mapsto 1 * s \end{array}$. Si la racine de l'arbre est rouge, par hypothèse de récurrence, dans $f_0(B_k)$ et dans $f_1(B_k)$, il existe un plongement de B_{m-1} rouge ou un plongement bleu de B_n. Si un des deux plongements bleus existe, on a un plongement de B_n bleu dans B_{k+1}. Sinon, on a un plongement rouge de B_{m-1} dans $f_0(B_k)$ et un dans $f_1(B_k)$. En considérant la racine du B_{k+1} colorié, on a un plongement rouge de B_m.
  – Le cas où la racine est bleue est analogue.

□

Définition 5.3. Pour $n \in \mathbb{N}$ et $i \in [|1, 2n-1|]$ on définit récursivement h par :
- $h(n, 2n-1) = 0$
- $h(n, i-1) = 2(h(n,i)+1) = 2(2^{n-i}-1) = 2^{2n-(i-1)}$

Puis $k(n) = h(n,0) = 4^n - 2$.

Lemme 5.4. *Soit T un arbre tel que $B_{k(n)}$ se plonge dans T par f_0. Alors il existe des noeuds $t_1, \dots, t_{n+1}$ de T, $x_1 < \cdots < x_n$ et $b \in 0,1$ tels que pour tous $j \in [|1,n|]$, $k \in [|1, n+1|]$:*
$t_k(x_j) = \begin{cases} 1-b & \text{si} \quad j < k \\ b & \text{si} \quad j \geq k \end{cases}$

Démonstration. On définit récursivement, pour $i \in [|1, 2n-1|]$, un plongement $f_i : B_{h(n,i)} \to T$, des noeuds $w_i, s_i \in T$ et $b_i \in \{0,1\}$.

Par hypothèse de récurrence, on suppose donné un plongement $f_{i-1} : B_{h(n,i-1)} \to T$. Soit $s \in B_{h(n,i-1)}$ tel que $f_{i-1}(s)$ soit de longueur maximale. On pose $s_i = f_{i-1}(s)$. s_i induit alors un 2-coloriage de $B_h(n,i)$: chaque noeud intérieur e est colorié par $s_i(|f_{i-1}(e)|)$, les feuilles étant coloriées arbitrairement, par exemple par 0.

Par le lemme 5.2 , puisque $h(n,i-1) = 2(h(n,i)+1)$, il existe un plongement monochromatique $g : B_{h(n,i)+1} \to B_{h(n,i-1)}$. On pose $w_i = f_{i-1}(g(\epsilon))$, la nouvelle racine du plongement $f \circ g$, et $b_i = s_i(|w_i|)$, la couleur du plongement. On pose finalement $f_i : \begin{matrix} B_{h(n,i)} \to T \\ s \mapsto f_{i-1}(g((1-b_i) * s)) \end{matrix}$

On remarque que $f_i(B_{h(n,i)}) \subseteq f_{i-1}(B_{h(n,i-1)})$ et que $w_i * (1-b_i) \sqsubseteq w_{i+1} \sqsubseteq s_{i+1}$. De plus, $s_k(|w_j|) = b_j$ si $j \geq k$.

Par le principe des tiroirs, il existe des indices $1 \leq i_1 < \cdots < i_n \leq 2n-1$ et $b \in \{0,1\}$ tels que $s_{i_m}(|w_{i_m}|) = b$ pour tout $m \in [|1,n|]$.

Pour $m \in [|1,n|]$, on pose $t_m = w_{i_m}$ et $x_m = |w_{i_m}|$, puis $t_{n+1} = w_{i_n} * (1-b)$.

Soient $j \in [|1,n|], k \in [|1, n+1|]$. On sait alors que, si $j \geq k$, $t_k(x_j) = 1-b$ par la construction des f_i. En outre, $t_k(x_j) = s_{i_k}(|w_{i_j}|) = s_{i_j}(|w_{i_j}|) = b$ si $j < k$. □

6 Preuve du théorème

On procède maintenant, en se servant des lemmes prouvés dans la section précédente, à la preuve du théorème de cardinalité :

Théorème 6.1 (Kummer 1992)**.** *Soient $A \subseteq \mathbb{N}$ et $m \geq 1$. On suppose qu'il existe une fonction récursive $g : \mathbb{N}^m \to \mathbb{N}$ telle que, pour tout m-uplet $(x_1, \dots, x_m)$ d'entiers naturels distincts :*

1. $W_{g(x_1,\dots,x_m)} \subsetneq [|0,m|]$
2. $\#_m^A(x_1, \dots, x_m) \in W_{g(x_1,\dots,x_m)}$

Alors A est récursif.

Démonstration. On considère l'arbre de possibilités défini par :

$$T_g = \left\{ t \in \{0,1\}^* \mid \forall x_1, \dots, x_m \left(x_1 < \cdots < x_m < |t| \implies \sum_{i=1}^{m} t(x_i) \in W_g(x_1, \dots, x_n) \right) \right\}$$

T_g est bien un arbre qui est récursivement énumérable : en effet, il suffit d'énumérer en parallèle les $W_{g(x_1,\dots,x_m)}$ avec $x_1 < \cdots < x_m < |t|$ et de vérifier la condition à chaque étape pour accepter les $t \in T_g$.

D'après (2), χ_A est une branche de T_g. Donc, par le lemme 5.1 , il suffit de montrer que T_g est de rang fini pour conclure.

Supposons, par l'absurde, que T_g est de rang infini. En particulier, on peut plonger $B_{k(m)}$ dans T_g. Donc, d'après le lemme 5.4 il existe des noeuds $t_1, \ldots, t_{m+1}$ de T_g, $x_1 < \cdots < x_m$ et $b \in 0, 1$ tels que pour tous $j \in [|1, m|]$, $k \in [|1, m+1|] : t_k(x_j) = \begin{cases} 1-b & \text{si} \quad j < k \\ b & \text{si} \quad j \geq k \end{cases}$

Or, d'après (1), il existe $k \in [|0, m|]$ tel que $k \notin W_{g(x_1,\ldots x_m)}$. Mais alors, si $b = 0$, $\sum_{j=1}^{m} t_k(x_j) = k$, et, si $b = 1$, $\sum_{j=1}^{m} t_{m+1-k}(x_j) = k$. Absurde.

Donc $\mathrm{rk}(T_g) \leq m$, d'où A est récursif. □

Bibliographie

1. M. KUMMER, *A proof of Beigel's cardinality conjecture*, The journal of Symbolic Logic, Vol 57, No. 2 (Jun 1992)
2. P. ODIFREDDI, *Classical recursion theory*, North-Holland, Amsterdam, 1989