Théorème de Cardinalité

Pierre GAILLARD

Décembre 2008

1 Présentation du Théorème

1.1 Oracles et Ensemble récursif

En 1987, Beigel a conjecturé un joli résultat sur les ensembles récursifs qu'on va ici présenter, puis en montrer une preuve assez élégante. On commence tout d'abord par introduire quelques notions nécessaires à la compréhension de la conjecture de Beigel et à sa preuve, à savoir les *machines à oracle* et les *ensembles récursifs*.

Une machine à oracle est une machine classique dans laquelle on peut dès que l'on souhaite interroger un oracle pour savoir dans quel état continuer. Exprimons ci-dessous la définition formelle.

Définition 1. Une *machine de Turing à oracle* $B \subset \Sigma^*$ est une machine de Turing disposant d'une bande distinguée et de trois états spéciaux $q_?$, q_{oui} et q_{non}. Si la machine entre dans l'état $q_?$ alors que le mot ω est écrit sur la bande distinguée, elle effectue une transition vers l'état q_{oui} si $\omega \in B$ et vers q_{non} sinon.

On pourra remarquer que la machine doit pouvoir, en un temps fini, obtenir la réponse à la question d'appartenance à B. Bien entendu, une machine à oracle peut ne jamais utiliser son oracle B, elle fonctionnera alors comme une machine ordinaire.

Exemple 1. Le langage $\left\{0^i1^i0^i1^j0^j1^j0 \mid i,j \geq 1\right\}$ est reconnu par une machine avec l'oracle $\{a^nb^nc^n \mid n \geq 1\}$ en temps linéaire.

Définition 2. Un ensemble A est dit *récursif* si sa fonction caractéristique, notée χ_A est récursive, c'est à dire calculable par une machine de Turing. Il est dit B-récursif si la machine de Turing est avec l'oracle B.

Déterminer si des ensembles sont récursifs est donc intimement lié à la théorie de la décidabilité. Le théorème de Cardinalité fait parti de ces résultats forcéments intéressants, qui conclut à la récursivité de certains ensembles. Donnons en premièrement l'idée intuitive. Les oracles permettent d'avancer beaucoup plus vite dans les calculs mais ils permettent surtout de

construire des machines extrêmement puissantes, allant jusqu'à reconnaître des langages indécidables. On s'intéresse ici à savoir si un ensemble est récursif selon le nombre de requêtes à un oracle. Il est assez intuitif qu'une machine posant peu de questions à un oracle aura tendance à reconnaître un langage plus simple qu'une machine lui en posant beaucoup.

1.2 La conjecture de Beigel

Un ensemble récursif est clairement B-récursif. Le théorème de Cardinalité s'intéresse à la réciproque : à quelles conditions un ensemble B-récursif est-il récursif ? En 1986, Beigel a démontré le théorème suivant.

Théorème 1 (Non-Speedup)**.** *Soient A et B deux parties de $\mathbb{N}$. Si la fonction de 2^n variables $(\chi_A(x_1), ..., \chi_A(x_{2^n}))$ peut-être calculée par un algorithme qui fait au plus n appels à l'oracle B, alors A est récursif.*

L'année d'après, il a conjecturé un résultat un peu plus fort. Nous allons voir qu'il est vrai. On note $\#A$ le cardinal d'un ensemble A. Pour $A \subset \mathbb{N}$ et $n \geq 1$, on définit la fonction suivante :

$$\forall x_1, ..., x_n \in \mathbb{N} \qquad \#_n^A(x_1, ..., x_n) = \#\{i, x_i \in A\} = \sum_{i=1}^{n} \chi_A(x_i)$$

Conjecture (Beigel)**.** *Soient A et B deux parties de $\mathbb{N}$ et $n \geq 1$. Si $\#_{2^n}^A$ peut être calculée par un algorithme faisant au plus n appels à l'oracle B alors A est récursif.*

1.3 Le théorème de Cardinalité

La conjecture de Beigel n'est cependant pas très pratique à montrer directement. On va donc montrer ici un théorème un peu plus fonctionnel. On commence par en donner l'énoncé, puis on montrera que la conjecture précédente n'en est qu'une conséquence.

1.3.1 Enoncé du théorème

Considérons une énumération $(W_i)_{i \in \mathbb{N}}$ des parties de $\mathbb{N}$ récursivement énumérables.

Théorème 2 (de Cardinalité)**.** *Soient $A \subset \mathbb{N}$ et $m \geq 1$. Supposons qu'il existe une fonction calculable $g : \mathbb{N} \to \mathbb{N}$ telle que pour tout $(x_1, ..., x_m) \in \mathbb{N}^m$ deux à deux distincts, on ait :*

1. $W_{g(x_1,...,x_m)} \subset \{0, ..., m\}$ *(L'inclusion est propre)*
2. $\#_m^A(x_1, ..., x_m) \in W_{g(x_1,...,x_m)}$

Alors A est récursif.

Cela revient en fait à faire un calcul approché de $\#_m^A$. On ne sait pas calculer précisément la valeur de $\#_m^A(x_1, ..., x_m)$ mais on sait qu'elle appartient à un ensemble $W_{g(x_1,...,x_m)}$ récursivement énumérable d'au plus m entiers. On voit bien toute la puissance du théorème, dans lequel on déduit la récursivité d'un ensemble A en ne supposant que de faibles hypothèses sur $\#_m^A$.

1.3.2 Lien avec la conjecture

Nous supposons ici que le théorème est démontré et nous allons en déduire la véracité de la conjecture de Beigel. Pour voir le lien entre le théorème précédent et la conjecture, nous avons besoin d'un petit lemme.

Lemme 1. *Si une fonction $f : \Sigma^* \to \Gamma^*$ peut être calculée avec n appels à un oracle B alors il existe un ensemble S d'au plus 2^n fonctions calculables telles que :*

$$\forall x \in \Sigma^* \quad \exists h \in S \qquad h(x) = f(x)$$

Démonstration. La preuve est assez intuitive, il suffit de considérer toutes les séries de réponses que pourrait donner l'oracle aux n interrogations que lui fait la machine de Turing quand elle a le mot x en entrée, il y en a au plus 2^n. Plus formellement, supposons f déterminée par la machine de Turing avec oracle M^B telle que pour toute entrée ω, M^B fait au plus n appels à l'oracle B. Pour chaque mot $\omega \in \{0,1\}^n$, on définit la fonction h_ω qui exécute $M^{(.)}$ en utilisant consécutivement les bits de ω en réponse aux appels à un éventuel oracle. Au moins une des séquences est correcte : la séquence des réponses si B avait été utilisé comme oracle. □

Soit $A \subseteq \mathbb{N}$ tel que $\#_{2^n}^A$ est calculables avec moins de n requêtes à un oracle $B \subseteq \mathbb{N}$. Il existe d'après le lemme un ensemble S d'au plus 2^n fonctions calculables telles que

$$\forall (x_1, ..., x_{2^n}) \in \mathbb{N}^{2^n} \quad \exists h \in S \qquad h(x_1, ..., x_{2^n}) = \#_n^A(x_1, ..., x_{2^n})$$

On définit alors une machine de Turing M qui exécute les unes après les autres toutes les fonctions $h \in S$ sur l'entrée $(x_1, ..., x_{2^n})$. On obtient ainsi un ensemble récursivement énumérable de 2^n entiers qui contient $\#_n^A(x_1, ..., x_{2^n})$ On reconnaît les hypothèses de théorème et on conclut que A est récursif.

2 Preuve du Théorème

2.1 Réduction du problème à un arbre

2.1.1 Notations et définitions

Commençons par rappeler quelques définissions et notations sur les arbres et les chaînes de caractères. On note ϵ la chaîne vide ; $|s|$ la longueur d'une

chaîne de caractères s, par exemple $|\epsilon| = 0$; $s \sqsubseteq t$, pour signifier que s est préfixe de t et pour tout $n < |s|$, $s(n)$ le $(n+1)$ème symbole de s. Un *arbre* est une partie de $\{0,1\}^*$ qui est close par prefixe. $s \in T$ est un noeud de T et $t \in \{0,1\}^{\mathbb{N}}$ est une *branche* de T si tout prefixe fini de t est un noeud de T. Les branches vont en fait nous servir de fonction indicatrice pour des parties de $\mathbb{N}$, puisque pour tout $n \in \mathbb{N} \quad t(n) \in \{0,1\}$. On dit alors intuitivement qu'une branche t d'un arbre est *récursive* si $\{i \in \mathbb{N}, t(i) = 1\}$ est récursif.

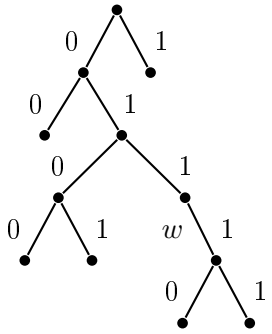

FIGURE 1 – Arbre $T = \{0, 00, 01, 010, 0100, 0101, 011, 0111, 01110, 01111, 1\}$. Le noeud $w = 0111 \in T$ est tel que $w(0) = 0$ et $w(2) = 1$

2.1.2 L'arbre intéressant

Soient $A \subseteq \mathbb{N}$, $m \geq 1$ et $g : \mathbb{N}$ vérifiant les hypothèses du théorème précédent. L'arbre récursivement énumérable qui nous intéresse ici est celui des possibilités défini par :

$$T_g = \left\{ t \in \{0,1\}^* \quad \middle| \quad \forall 0 \leq x_1 < ... < x_m < |t| \quad \sum_{i=1}^{m} t(x_i) \in W_{g(x_1,...,x_m)} \right\}$$

En effet, on peut remarquer qu'il s'agit bien d'un arbre binaire mais essentiellement que $\sum_{i=1}^{m} \chi_A(x_i) = \#_m^A(x_1, ..., x_m) \in W_{g(x_1,...,x_m)}$. χ_A est donc une branche de T_g. On veut montrer que A est récursif, c'est à dire que χ_A est récursive. Pour cela, notre preuve reposera en deux étapes :

- On cherchera tout d'abord sous quelles conditions les branches d'un arbre récursivement énumérable sont récursives.
- On montrera ensuite que T_g vérifie ces conditions.

2.2 Condition sur un arbre pour que ses branches soient récursives

2.2.1 Plongement et rang

Notons B_n l'arbre binaire complet de hauteur n et $\cdot$ la concaténation des chaînes de caractères.

Définition 3. Soit T un arbre. On dit que $f : B_n \to T$ est un *plongement* de B_n dans T si et seulement si

$$\forall s \in \{0,1\}^{n-1} \qquad f(s) \cdot 0 \sqsubseteq f(s \cdot 0) \qquad \& \qquad f(s) \cdot 1 \sqsubseteq f(s \cdot 1)$$

On dit que B_n *se plonge* dans T sous un noeud $e \in T$, s'il existe un plongement f de B_n dans T tel que $e \sqsubseteq f(\epsilon)$, ie $f(\epsilon) = e \cdot u$. Enfin, le *rang* d'un arbre T est : $rg(T) = sup\{n \geq 0,\ B_n \text{ se plonge dans } T\} \in \mathbb{N} \cup \{\infty\}$.

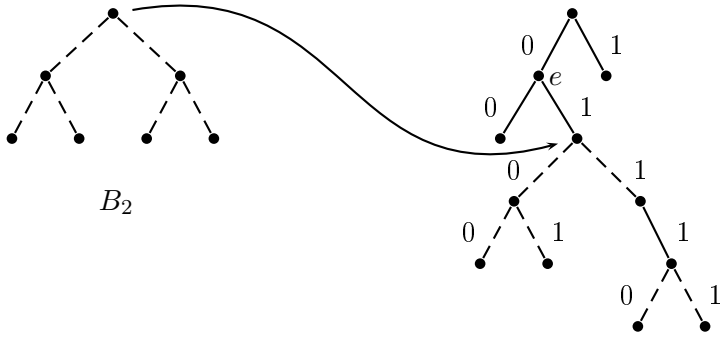

FIGURE 2 – Exemple de plongement d'un arbre binaire B_2 dans un arbre T sous un noeud e, $rg(T) = 2$.

2.2.2 La condition

Lemme 2. *Si T est un arbre récursivement énumérable de rang fini alors chacune de ses branches est récursive.*

Démonstration. Soient T un tel arbre, t une de ses branches et $k_0 = sup\{n \geq 0 \,|\, B_n \text{ se plonge dans } T \text{ sous chaque noeud } e \in T\}$, alors $k_0 \leq rg(T)$. On choisit alors un noeud $e_0 \sqsubseteq t$ tel que B_{k_0+1} ne se plonge pas sous e_0. Alors t est calculable par l'algorithme suivant :

Si $x > |e_0|$, alors on énumère T et on cherche un plongement f de B_{k_0} tel que $|f(\epsilon)| > x$. On renvoie alors $f(\epsilon)(x)$. En effet, $f(\epsilon) \sqsubseteq t$ pour tout plongement $f : B_{k_0} \to T$, sinon on raisonne par l'absurde et on suppose le contraire. Soit alors $d \in T$, le préfixe maximal commun à $f(\epsilon)$ et t. On a $e_0 \sqsubseteq d$. Commme B_{k_0} se plonge sous $d \cdot t(d)$, d est la racine d'un plongement B_{k_0+1}, ce qui contredit la définition de e_0.

Sinon $x \leq |e_0|$, on cherche alors $t(x)$ parmi un ensemble fini. □

2.3 T_g vérifie cette condition

Montrons maintenant que $rg(T_g) < \infty$. C'est la partie un peu plus combinatoire de la preuve. Pour cela, nous allons voir deux lemmes.

Lemme 3. *Pour tout arbre B_{2k} bicolore, il existe un plongement g de B_k dans B_{2k} tel que tous les noeuds de $g(B_k)$ soient de la même couleur.*

Démonstration. Soit B_N, un arbre noir et blanc. Montrons, par récurrence sur la hauteur N de B_N, que pour tout $k \in \{0, ...N\}$ B_N contient soit un plongement noir B_k, soit un plongement blanc B_{N-k}.

- Si $N = 1$: c'est immédiat.
- Sinon, soient $N \geq 1$ et $0 \leq k \leq N$. On peut supposer $1 \leq k \leq N-1$. Supposons de plus par exemple que la racine de B_N soit noire (l'autre cas étant symétrique). Si l'un des fils, droit ou gauche, possède un plongement blanc B_{N-k}, c'est fini. Sinon, ils possèdent tous les deux un plongement noir B_{k-1}, B_N contient donc un plongement noir B_k.

□

On montre maintenant dans le dernier lemme que pour un arbre dont le rang est suffisament grand, on peut trouver n entiers $x_1, ..., x_n$ et $n+1$ noeuds $t_1, ..., t_{n+1}$ tels que $\sum_{i=1}^{n} t_j(x_i)$ prend toutes les valeurs de $\{0, ..., n\}$ quand j varie de 1 à $n+1$. La figure 3 illustre cette propriété. Cela permettra d'aboutir à une contradiction avec les hypothèses du théorème si le rang de T_g est infini. Le lemme énonce en fait un résultat un peu plus fort, qui est innutile pour la démonstration du théorème de Cardinalité mais qui pourrait servir à montrer des variantes.

Construisons deux fonctions utiles pour le lemme. Pour $n \geq 1$ et $1 \leq i \leq 2n-1$, on pose $h(n, 2n-1) = 0$ et $h(n, i-1) = 2(h(n, i) + 1)$. On pose alors $k(n) = h(n, 0) = 4^n - 2$. C'est le rang à partir duquel on peut faire une telle construction.

Lemme 4. *Pour tout $n \geq 1$ et tout arbre T tel que $B_{k(n)}$ se plonge dans T, il existe $t_1, ..., t_n$, $n+1$ noeuds de T, des entiers $x_1 < ... < x_n$ et $b \in \{0, 1\}$ tels que :*

$$\forall i = 1, ..., n \quad \forall j = 1, ..., n+1 \qquad t_j(x_i) = \begin{cases} 1-b & si\ i < j \\ b & si\ i \geq j \end{cases}$$

En particulier, $\{\sum_{i=1}^{n} t_j(x_i) \,|\, 1 \leq j \leq n+1\} = \{0, 1, ..., n\}$.

Démonstration. Soient $n \in \mathbb{N}^*$ fixé et T un arbre tel que $B_{k(n)}$ se plonge dans T par f_0. On va construire par récurrence sur $0 \leq i \leq 2n-1$ deux mot w_i et s_i de T, $b_i \in \{0, 1\}$ et un plongement $f_i : B_{h(n,i)} \to T$.

On suppose qu'on dispose d'un plongement $f_{i-1} : B_{h(n,i-1)} \to T$. L'idée est de choisir la feuille dont l'image s_i par le plongement est la plus profonde dans l'arbre. On pourra alors en utilisant le lemme 3, trouver un plongement plus petit tel qu'on ne garde que les niveaux de l'arbre où les arêtes de cette branche ne contiennent que des 0 ou que des 1, on note alors b_i cette couleur et w_i le début du plongement. A la fin de la preuve, on disposera

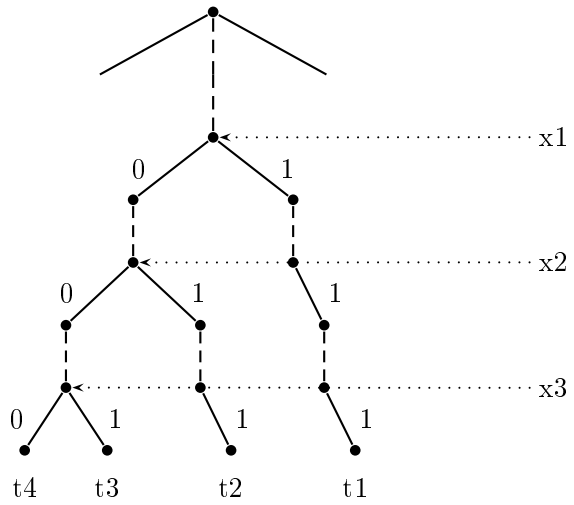

FIGURE 3 – Illustration du lemme 4 avec $b = 1$. On remarque qu'on a bien $\left\{\sum_{i=1}^{3} t_j(x_i) \,|j = 1,..,4\right\} = \{0,1,2,3,4\}$

de $w_1, ..., w_{2n-1}$, on pourra donc en utilisant le principe des tiroirs en choisir n de la même couleur (0 ou 1) et on aura notre construction. Par exemple, sur la figure 3, on a pu avoir $s_1 = t_1$ et $w_1 = x_1$, on n'a alors gardé que les niveaux de T où s_1 ne contenait que des 1 et on a continué. On verra que c'est possible car on a choisit le rang de l'arbre suffisamment grand. Formalisons maintenant cette idée.

Soit s une feuille de $B_{h(n,i-1)}$ telle que $f_{i-1}(s)$ soit de longeur maximale. On pose $s_i = f_{i-1}(s)$. s_i colorie $B_{h(n,i-1)}$ de deux couleurs de la manière suivante :

- chaque noeud interne e de $B_{h(n,i-1)}$ est de couleur $s_i(|f_{i-1}(e)|)$ qui existe par définition de s_i.
- chaque feuille est arbitrairement coloriée en 0.

D'après le lemme 2 et la définition de h, il existe un plongement g de $B_{h(n,i)+1}$ dans $B_{h(n,i-1)}$ tel que $g\left(B_{h(n,i)+1}\right)$ ne soit rempli que de 0 ou que de 1. On pose alors $w_i = f_{i-1}(g(\epsilon))$ et $b_i = s_i(|w_i|)$. Puis, on définit le plongement suivant, $f_i : B_{h(n,i)} \to T$ par $f_i(s) = f_{i-1}(g((1-b_i)\cdot))$ pour $|s| \leq h(n,i)$. On peut alors remarquer qu'on a bien :

- $f_i\left(B_{h(n,i)}\right) \subseteq f_{i-1}\left(B_{h(n,i-1)}\right)$: en effet à chaque étape on ne fait que restreindre le plongement à la partie qui nous intéresse.
- $w_i \cdot (1-b_i) \sqsubseteq w_{i+1} \sqsubseteq s_{i+1}$
- $\forall i \geq j \quad s_j(|w_i|) = b_j$

Par le principe des tirroirs, il existe $b \in \{0,1\}$ et n indices $1 \leq i_1 < ... < i_n \leq 2n-1$ tels que $\forall m = 1, ..., n \quad s_{i_m}(|w_{i_m}|) = b$.

On pose alors pour tout $m = 1, ..., n \quad t_m = s_{i_m}$ et $x_m = |w_{i_m}|$, puis $t_{n+1} = w_{i_n} \cdot (1-b)$. Le lemme en découle aisément. □

Pour finir la preuve du théorème de cardinalité, il ne reste plus maintenant qu'à conclure en montrant que le rang de T_g est fini. On raisonne par l'absurde et on suppose qu'il est infini. En particulier, pour tout $m \geq 1$ il est supérieur à $k(m)$. On peut donc appliquer le lemme 4. Il existe donc $t_1, ..., t_{m+1} \in T_g$ et m entiers $x_1, ..., x_m$, tels que $\sum_{i=1}^{m} t_j(x_i) = \{0, 1, ..., m\}$. On en déduit par définition de T_g que $\{0, 1, ..., m\} \subseteq W_{g(x_1,...,x_m)}$ ce qui est en contradiction avec la première hypothèse du théorème. D'après le lemme 1, chaque branche de T_g est donc récursive, en particulier χ_A. Finalement, A est récursif. Le théorème est donc démontré.

Références

[1] Martin Kummer. *A proof of Beigel's cardinality conjecture*, 1991.