Le théorème de cardinalité

François-Régis André

Mercredi 23 décembre 2009

Table des matières

Un langage L est décidable si une machine de Turing qui termine toujours permet de calculer si un mot est dans le langage. Cette notion occupe une place centrale en théorie de la calculabilité, car c'est souvent une condition minimale à imposer a un langage pour pouvoir l'étudier. La définition n'est pas toujours facile à vérifier, et il est donc utile d'avoir des conditions nécessaires ou suffisantes à vérifier pour qu'un langage soit décidable. Le premier critère simple est que si on peut réduire grâce à une machine de Turing le test d'appartenance à L à un nombre fini de tests dans un autre langage L' que l'on sait décidable, alors L est décidable. Il n'est cependant pas toujours facile de trouver un lien avec un autre langage décidable. En 1986, Beigel a montré le théorème de non-accélération : si on peut réduire un grand nombre de tests pour L à un petit nombre de tests dans un langage L', alors L est décidable. On démontre ici ce théorème ainsi qu'une conjecture de Beigel qui en est une amélioration.

1 Machines de Turing avec oracle

1.1 Définition

Le but de cette étude est d'étudier des parties de $\mathbb{N}$ qui se caractérisent simplement par une machine de Turing à l'aide d'une autre partie.

Définition 1. Une machine de Turing avec oracle est une machine de Turing qui a de plus une bande particulière, l'oracle, qui lui permet de calculer instantanément une fonction à valeurs dans $\{0,1\}$.

L'oracle permet donc de reconnaitre un ensemble quelconque de mots.

1.2 Théorème de non accélération et conjecture de Beigel

Soient A une partie de $\mathbb{N}$ et $n \in \mathbb{N}$. On note :

$$\chi_{A,n} : \mathbb{N}^n \to \{0,1\}^n \ , \ (x_1, x_2, \ldots, x_n) \mapsto (\chi_A(x_1), \ldots, \chi_A(x_n))$$
$$\#_{A,n} : \mathbb{N}^n \to \mathbb{N} \, , \, (x_1, x_2, \ldots, x_n) \mapsto \sum_{i=1}^{n} \chi_A(x_i) = \#\{i \in \{1, \ldots, n\} \,|x_i \in A\}$$

où χ_A désigne la fonction caractéristique de A.

Théorème 2. *(théorème de non-accélération) Si $\chi_{A,2^n}$ peut être calculée par une machine de Turing avec oracle qui ne fait pas plus de n appels à cet oracle, alors A est récursif.*

En 1987, Beigel énonce une conjecture qui renforce ce théorème :

Proposition 3. *Si $\#_{A,2^n}$ peut être calculée par une machine de Turing avec oracle qui ne fait pas plus de n requêtes à cet oracle, alors A est récursif.*

1.3 Fonctions calculables avec un nombre borné de requêtes à un oracle

Lemme 4. *Si une fonction f peut être calculée grâce à moins de n requêtes à un oracle pour un certain entier n, alors il existe un ensemble S d'au plus 2^n fonctions partielles récursives tel que :*

$$\forall x \in \mathbb{N}, \exists g \in S, f(x) = g(x)$$

Démonstration. On suppose que f est calculée par la machine de Turing avec oracle M qui ne fait pas plus de n requêtes à son oracle et ce, pour toute entrée, mais aussi quelques soient les réponses de l'oracle. Pour tout mot $w \in \{0,1\}^n$, on considère la machine de Turing M_w qui fonctionne comme M en supposant que l'oracle réponde en suivant les lettres de w lors des n premières requêtes, puis éventuellement ne réponde que 0. On obtient donc une machine de Turing au sens habituel en simulant les réponses de l'oracle. On note g_w la fonction partielle récursive calculée par M_w. Pour toute entrée x, l'oracle de M répond en suivant les lettres d'un certain mot w, donc M effectue le même calcul que M_w, et $f(x) = g_w(x)$. On en déduit que $S = \{g_w | w \in \{0,1\}^n\}$ convient. □

2 Arbres binaires

On démontre un résultat plus fort que la conjecture de Beigel, le théorème de cardinalité. On a besoin pour cela de quelques résultats sur les arbres binaires.

2.1 Définitions

On considère des arbres binaires éventuellement infinis. Un nœud est représenté par le chemin parcouru depuis la racine pour y accéder.

Définition 5.

1. Un arbre est une partie T de $\{0,1\}^*$ close par préfixe. Les nœuds de T sont ses éléments. Une suite $t \in \{0,1\}^{\mathbb{N}}$ est appelée branche de T lorsque tous ses préfixes finis sont des nœuds de T. On note lorsque t est un nœud ou une branche et $n \in \mathbb{N}$, $t(n)$ le n+1-ème terme de la suite définie par t. On note la concaténation de suites avec $*$. On note $s \preceq t$ lorsque s est un préfixe de t.
2. Les arbres considérés peuvent être infinis et ne peuvent donc pas être comparés par leur taille. On choisit ici un critère de densité donné par la notion de plongement :
   Soient T_1 et T_2 deux arbres, on dit que $f : T_1 \to T_2$ est un plongement de T_1 dans T_2 lorsque pour tout $t_1 \in T_1$,t_1 est un sous-mot de $f(t_1)$. On peut donner une définition équivalente par récurence :
   - La fonction de l'arbre vide dans T est un plongement.
   - En notant $r_i = \epsilon$ la racine de T_i, et pour un nœud s, $f_g(s)$ et $f_d(s)$ ses fils gauche et droit, f est un plongement de T_1 dans T_2 si f se restreint en un plongement de $f_g(r_1)$ dans $f_g(f(r_1))$ et en un plongement de $f_d(r_1)$ dans $f_d(f(r_1))$.

On pourra identifier un plongement avec son image. On dit que T_1 se plonge dans T_2 sous un nœud $s \in T_2$ s'il existe un plongement f tel que $s \preceq f(r_1)$. On note B_n l'arbre complet de hauteur n et on note $rg(T)$ le rang de T définit comme le supremum des n tels que B_n se plonge dans T.

2.2 Arbres récursivement énumérables de rang fini

Lemme 6. *Si T est un arbre récursivement énumérable de rang fini alors toutes les branches de T sont récursives.*

Démonstration. Soit t une branche de T et soit k_0 le supremum des k tels que B_k puisse être plongé dans T sous tous les nœuds $s \preceq t$ (k_0 est fini, inférieur à $rg(T)$). Soit s_0 un noeud de t tel que B_{k_0+1} ne puisse pas être plongé dans T sous s_0. Pour montrer que t est récursive, on donne un algorithme qui calcule en fonction d'un entier x un nœud s de t de profondeur supérieure à x. Les nœuds de t de profondeur inférieure à x seront donc les préfixes de s.
On initialise T' comme l'arbre vide. A chaque étape où T' est modifié, on clos T' par préfixe de façon à obtenir un arbre fini, qui sera par construction un sous-arbre de T. On énumère les éléments de T que l'on ajoute à T' jusqu'à obtenir s_0. Puis s'il n'existe pas de plongement f de B_{k_0} dans T' sous s_0 tel que $|f(\epsilon)| \geq x$, on continue de rajouter les éléments de T obtenus par énumération jusqu'à ce que l'on trouve f, plongement de B_{k_0} dans T' sous s_0 tel que $|f(\epsilon)| \geq x$. Le nœud cherché est alors $f(\epsilon)$.
En effet, si on suppose par l'absurde que $f(\epsilon)$ n'est pas un nœud de t, on note g un plongement de B_{k_0} dans T sous s_0 tel que $g(\epsilon) \in t$ et on note r le plus grand préfixe commun de $f(\epsilon)$ et $g(\epsilon)$. $f(\epsilon)$ est dans un fils de r tandis que $g(\epsilon)$ est dans l'autre par définition de r. Mais dans ce cas, $\{r\} \cup f(B_{k_0}) \cup g(B_{k_0})$ est un plongement de B_{k_0+1} dans T sous s_0, ce qui contredit la définition de s_0. □

2.3 Propriétés combinatoires

Lemme 7. *Pour tout $n \in \mathbb{N}$ et tout 2-coloriage $c : B_{2n} \to \{0,1\}$, il existe un plongement g monochromatique de B_n dans B_{2n}.*

Démonstration. On montre par récurence sur $m+n$, $m,\ n \geq 1$ que pour tout 2-coloriage de B_{m+n}, il existe un plongement monochromatique de B_m de couleur 0 ou un plongement monochromatique de B_n de couleur 1.

- Si $m = n = 0$, le résultat est clair.
- Soit c un 2-coloriage de B_{m+n}, $n \geq 1$. On suppose par exemple que $g(\epsilon) = 1$. Par hypothèse de récurence, soit $f_g(\epsilon)$ ou $f_d(\epsilon)$ admet un plongement monochromatique de B_m de couleur 0 et donc B_{m+n} également, soit $f_g(\epsilon)$ et $f_d(\epsilon)$ admettent respectivement un plongement monochromatique g_1 et g_2 de $B_{(n-1)}$ de couleur 1 et alors $\epsilon \cup g_1(f_g(\epsilon)) \cup g_2(f_d(\epsilon))$ est un plongement de B_n dans B_{m+n} de couleur 1.

□

Lemme 8. *Pour tout $n \geq 1$ et tout arbre T de rang $rg(T) \geq 4^n - 2$, il existe des nœuds $t_1, \ldots, t_{(n+1)}$ de T, des entiers $x_1 \leq \ldots \leq x_n$ et $b \in \{0,1\}$ tels que :*

$$\text{pour tous } i = 1, \ldots, n,\ j = 1, \ldots, n+1 \ : t_j(x_i) = \begin{cases} b & \text{si } i \geq j \\ 1-b & \text{si } i < j \end{cases}$$

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

Démonstration. On définit par récurence pour $n \in \mathbb{N}$ et $1 \leq i \leq 2n-1$:

- $h(n,\ 2n-1) = 0$
- $h(n,\ i-1) = 2(h(n,\ i)+1)$

En particulier, pour tout n : $h(n,0) = 4^n - 2$. On suppose que f_0 est un plongement de $B_{h(n,0)}$ dans T. On définit par récurence sur $i = 1, \ldots, 2n-1$, des nœuds $w_i,\ s_i \in T$, $b_i \in \{0,1\}$ et $f_i \ : B_{h(n,\ i)} \to T$.
Soit s une feuille de $B_{h(n,\ i-1)}$ telle que $f_{(i-1)}(s)$ ait une longueur maximale. On pose $s_i = f_{(i-1)}(s)$. On construit un 2-coloriage de $B_{h(n,\ i-1)}$: chaque nœud intérieur e est colorié par $s_i\left(|f_{(i-1)}(e)|\right)$ par maximalité de s_i. On colorie les feuilles avec la couleur 0 par exemple. Par le lemme **7**, et comme $h(n,\ i) = 2(h(n,\ i-1)+1)$, il existe un plongement g de $B_{(h(n,\ i-1)+1)}$ dans $B_{h(n,\ i)}$ dont l'image est monochromatique. On pose :

$$w_i = f_{(i-1)}(g(\epsilon)) \text{ et } b_i = s_i(|w_i|).$$

On définit f_i par :

$$f_i(s) = f_{(i-1)}(g((1-b_i) * s)) \text{ pour } s \in B_{h(n,\ i)}.$$

Cette construction permet d'obtenir les propriétés suivantes :

1. $f_i\left(B_{h(n,\ i)}\right) \subseteq f_{(i-1)}\left(B_{h(n,\ i-1)}\right)$
2. $w_i * (1-b_i) \preceq w_{(i+1)}$
3. $s_j(|w_i|) = b_j$ pour $i \geq j$

Par le principe des tiroirs, comme $b_1, \ldots, b_{2n-1} \in \{0,1\}$, n d'entre eux au moins sont égaux : il existe $b \in \{0,1\}$ et n indices $1 < i_1 < \ldots < i_n \leq 2n-1$ tels que pour tout $m = 1, \ldots, n$, $s_{i_m}(|w_{i_m}|) = b$. On obtient alors le résultat voulu avec pour $1 \leq m \leq n$:

- $t_m = s_{i_m}$
- $x_m = |w_{i_m}|$
- $t_{(n+1)} = w_{i_n} * (1-b)$

□

Cette preuve construit par récurence les x_i grâce aux fonctions f_i qui garantissent que le procédé pourra se poursuivre jusqu'à son terme.

3 Le théorème de cardinalité

3.1 Énoncé

On désigne par $\mathcal{P}^*(E)$ l'ensemble des parties strictes de l'ensemble E. A est une partie de $\mathbb{N}$.

Théorème 9. *Si pour $m \in \mathbb{N}$, il existe une fonction G récursive $\mathbb{N}^m \to \mathcal{P}^*(\{0,\ldots,m\})$, telle que pour tous $(x_1,\ldots,x_m) \in \mathbb{N}$:*

$$\#_{A,m}(x_1,\ldots,x_m) \in G(x_1,\ldots,x_m)$$

alors A est récursive.

Avec l'hypothèse de la conjecture de Beigel,on obtient grâce au lemme **4**, dans le cas où m est de la forme 2^n, m fonctions partielles récursives $f_1,\ldots,f_m$ prenant leurs valeurs entre 0 et m. On est alors dans les hypothèses du théorème avec :

$$G(x_1,\ldots,x_m) = \{f_i(x_1,\ldots,x_m) \,|\, i = 1,\ldots,m\}.$$

On en déduit que le théorème de cardinalité implique la conjecture de Beigel.

3.2 Preuve du théorème

Démonstration. D'après l'hypothèse on peut construire un arbre récursivement énumérable :

$$T_G = \left\{ t \in \{0,1\}^* \,|\, \forall x_1 \leq \ldots \leq x_m \leq |t|, \sum_{i=1}^{m} t(x_i) \in G(x_1,\ldots,x_m) \right\}$$

Par hypothèse, χ_A est une branche de T_G. Par le lemme **6**, il suffit de monter que T_G est de rang fini. Mais si on suppose par l'absurde que $rg(T_G) \geq 4^m - 2$, par le lemme **8**, on trouve $t_1,\ldots,t_{(m+1)} \in T_G$ et des entiers $x_1 \leq \ldots \leq x_m$ tels que :

$$\left\{ \sum_{i=1}^{m} t_j(x_i) \,|\, 1 \leq j \leq m+1 \right\} = \{0,\ldots,m\}$$

Par définition de T_G, on trouve que $\{0,\ldots,m\} \subseteq G(x_1,\ldots,x_m)$, ce qui contredit l'hypothèse du théorème. □