Développement sur la décidabilité des formules de l'addition sur $\mathbb{N}$

22 septembre 2022

Source : *Langages formels, calculabilité et complexité*, Carton, Olivier et al., p. 178.

Niveau de la leçon et prérequis : Cette leçon peut se faire en classe prépa/licence après avoir traité les propriétés de base des automates finis.

Leçons convernées : 27. Décidabilité et indécidabilité. Exemples. et 29. Langages rationnels et automates finis. Exemples et applications.

Théorème
Le problème suivant est décidable :

> ***Entrée :*** *Une formule F du premier ordre sur le lagage $L = \{0, 1, +, =\}$.*
>
> ***Sortie :*** *La formule est vraie ou fausse dans $\mathbb{N}$.*

Exemples
Voici quelques exemples de formules du premier ordre sur le langage L ainsi que leur valeur de vérité dans $\mathbb{N}$.

1. $\forall x, \forall y, \exists z, x + y = z$ *est vraie dans* $\mathbb{N}$.
2. $\exists x, 0 = x + 1$ *est fausse dans* $\mathbb{N}$.
3. $\forall x, \neg\ x = x + 1$ *est vraie dans* $\mathbb{N}$.
4. $\forall x, \forall y, x + 1 = y + 1 \implies x = y$ *est vraie dans* $\mathbb{N}$.

Démonstration.

Idée
Pour chaque formule F de la logique du premier ordre sur le langage L, nous allons construire un automate fini $\mathcal{A}_F$ tel que le langage reconnu par l'automate est non vide si et seulement si la formule est vraie dans $\mathbb{N}$.

Soit F une formule logique sur le langage L du premier ordre que l'on suppose sous forme prénexe sans perte de généralité. La formule s'écrit alors $F = \#_1 x_1, \ldots, \#_n x_n, Q(x_1, \ldots, x_n)$ dans laquelle les $\#_i$ sont des quantificateurs et $Q(x_1, \ldots, x_n)$ est une formule sans quantificateurs dans laquelle les variables $x_1, \ldots, x_n$ sont libres. La preuve que toute formule a une formule équivalente sous forme prénexe se fait par induction sur la formule et en utilisant les règles d'inférences de la déduction naturelle.

Exemple
Si $F = (\forall x, G) \wedge H$ alors, soit y une variable non libre dans H, $F' = \forall y, G[y/x] \wedge H$ est équivalente sémantiquement à F.

Pour tout $k \in [\![1; n]\!]$ on définit $F_k = \#_{k+1} x_{k+1}, \ldots, \#_n x_n, Q(x_{k+1}, \ldots, x_n)$ avec $F = F_0$ et $Q = F_n$. On va alors construire itérativement un automate fini $\mathcal{A}_{F_k}$ dont le langage est l'ensemble de certains encodages des k-uplets d'entiers $(x_1, \ldots x_k)$ tels que $F_k(x_1, \ldots, x_k)$ est vraie.

Définition (Codage des tuples d'entiers)
Soit $(x_1, \ldots, x_k) \in \mathbb{N}^k$, l'ensemble des encodages valides pour $(x_1, \ldots, x_k)$ est $\{(0^{n_i}\overline{x_{ib}})_{i \in [\![1;k]\!]} | \forall s, n_i + |\overline{x_{ib}}| = s\}$ où $\overline{x}_b$ est l'encodage binaire classique de $x \in \mathbb{N}$ dont le bit de poids fort est un 1.

On verra un encodage de $(x_1, \ldots, x_n)$ comme un mot de taille k sur $\{0,1\}^n$.

On notera pour $k > 1$, X_k le langage des encodages de k-uplets d'entiers $(x_1, \ldots, x_k)$ tels que $F_k(x_1, \ldots, x_k)$ est vraie.

Remarques
On peut à ce point de la démonstration faire les remarques suivantes :

1. *Nous souhaitons prouver $\forall k > 1, X_k = \mathcal{L}(\mathcal{A}_{F_k})$*
2. *L'alphabet pour l'automate $\mathcal{A}_{F_k}$ est $\{0,1\}^k$ et pour $\mathcal{A}_F$ l'alphabet de l'automate est $\{0,1\}^0 = \{()\}$ le singleton avec le 0-uplet.*

Il faut maintenant consrtuire $\mathcal{A}_{F_n}$ puis $\mathcal{A}_{F_{k-1}}$ à partir de $\mathcal{A}_{F_k}$.

Pour construire $\mathcal{A}_{F_n}$ nous allons remarquer que l'on peut se limiter à des formules F_n qui sont des combinaisons booléennes de formules atomiques du type $x_i = 0$, $x_i = 1$, $x_i = x_j$ ou $x_i + x_j = x_l$ quitte à rajouter des variables.

Exemple

Par exemple, $Q(x_1, \ldots, x_3) = (x_1 + x_2 = x_3 + 1 \wedge x_1 + 1 = x_2) \implies (x_1 + x_1 = x_3)$ *peut se réécrire de façon sémantiquement équivalente à* $\exists y_1, \exists y_2 Q'(x_1, \ldots, x_3, y_1, y_2)$ *avec* $Q'(x_1, \ldots, x_3, y_1, y_2) = \neg y_1 = 1 \vee \neg y_2 = x_3 + y_1 \vee \neg y_2 = x_1 + x_2 \vee \neg x_2 = x_1 + x_1 \vee x_3 = x_1 + x_1$.

On fait donc apparaître de nouvelles variables et de nouveaux quantificateurs, on réécrit en fait toute la formule $F = \#_1 x_1, \ldots, \#_n x_3, Q(x_1, \ldots, x_3)$ *en* $F' = \#_1 x_1, \ldots, \#_3 x_3, \#_4 y_1, \#_5 y_2, Q'(x_1, \ldots, x_3, y_1, y_2)$.

Remarque

Prendre une forumle Q avec uniquement des sous formules atomiques de cette forme permet de supposer qu'il y a au moins un quantificateur.

Pour construire $\mathcal{A}_{F_n}$ il suffit alors de construire un automate fini déterministe complet pour chaque formule atomique et appliquer les transformations ensemblistes sur les automates.

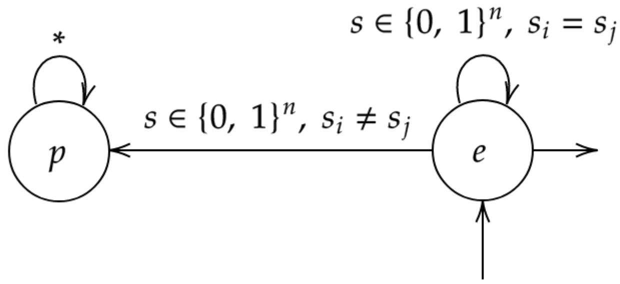

Figure 1 – Automate pour la formule $x_i = x_j$

La construction formelle de $\mathcal{A}_{F_n}$ est un peu lourde, je ne la présente pas. Il est facile de comprendre que le langage de l'automate est bien X_n.

Nous allons maintenant construire $\mathcal{A}_{F_k}$ à partir de $\mathcal{A}_{F_{k+1}}$. On a $F_k = \#_k x_k, F_{k+1}$, on doit discuter du cas où $\#_k$ est $\forall$ et $\#_k$ est $\exists$. On traîte d'abord le cas du quantificateur existenciel.

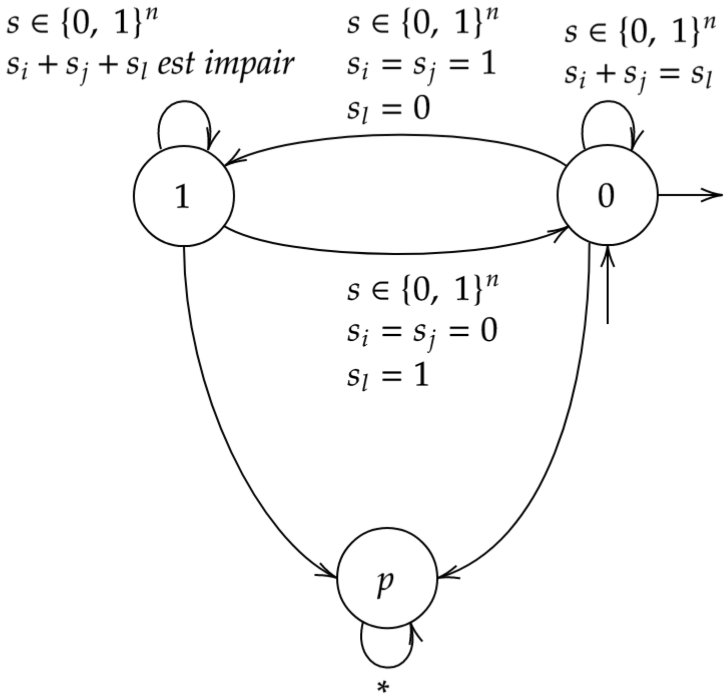

FIGURE 2 – Automate pour la formule $x_i + x_j = x_l$

On suppose que $F_k = \exists x_k, F_{k+1}$. Pour prendre en compte le quantificateur existenciel $\exists x_k$, il suffit d'"ingorer" l'indice k dans les transitions. Plus formellement, si $\mathcal{A}_{F_k} = (\{0,1\}^k, Q, \delta, I, S)$ où Q est l'ensemble fini des états, $I \subseteq Q$ est l'ensemble des états initiaux, $S \subseteq Q$ est l'ensemble des états finaux et $\delta : \{0,1\}^k \times Q \to Q$ est la fonction de transitions. Alors, on pose $\mathcal{A}_{F_k} = (\{0,1\}^{k-1}, Q, \delta', I', S)$ avec $\delta'((d_1, \ldots, d_{k-1}), q') = q$ si et seulement si $\delta((d_1, \ldots, d_{k-1}, d_k), q') = q$ et $I' = I \cup \delta((0, \ldots, 0, *), I)$ l'ensemble des états accessibles à partir des états initiaux de $\mathcal{A}_{F_{k+1}}$ en lisant des 0 sur les k premières entrées.

Montrons que le langage reconnu par $\mathcal{A}_{F_k}$ est exactement les encodages en binaires avec potentiellement des vecteurs nuls devant des k-uplets d'entiers $(x_1, \ldots x_k)$ tels que $F_k(x_1, \ldots, x_k)$ est vraie. On sait que le langage de $\mathcal{A}_{F_{k+1}}$ est exactement les encodages des $k+1$-uplets d'entiers $(x_1, \ldots x_k, x_{k+1})$ tels que $F_{k+1}(x_1, \ldots, x_k, x_{k+1})$ est vraie. Soit $(x_1, \ldots x_k)$ tels que $F_k(x_1, \ldots, x_k)$ est vraie, alors s'il existe $x_{k+1} \in \mathbb{N}$ tel que $F_{k+1}(x_1, \ldots, x_{k+1})$ est vraie, c'est à dire qu'il existe un chemin d'un état initial à un état final dans lequel les k premiers entiers qui indicent le chemin sont $(x_1, \ldots, x_k)$. Donc tout k-uplet encodé dans X_k au moins un prolongement dans X_{k+1}. De plus, si $(x_1, \ldots, x_{k+1})$ indice un chemin d'un état initial de $\mathcal{A}_{F_{k+1}}$ à un état final, cela signifie qu'il existe $x_k \in \mathbb{N}$ tel que $F_{k+1}(x_1, \ldots, x_{k+1})$ est vraie et donc $F_k(x_1, \ldots, x_k)$ est vraie. Cela signifie que tous les encodages de $(x_1, \ldots, x_k)$ doivent être dans X_k. On a bien le langage souhaité pour $\mathcal{A}_{F_k}$ grace à l'astuce d'ajouter des états initiaux.

On sait que l'on peut ré-écrire $F_k = \forall x_k, F_{k+1}$ en $F_k = \neg\exists x_k, \neg F_{k+1}$. Ainsi, traiter le cas du quantificateur existenciel permet de traiter le cas du quantificateur universel. En effet, si $L(\mathcal{A}_{\mathcal{F}_{\|+\infty}}) = X_{k+1}$ alors, le langage $\overline{X_{k+1}}$ est l'ensemble des encodages des tuples tels que F_{k+1} n'est pas vraie (ie. $\neg F_{k+1}$ est vraie), qui est reconnu par l'automate complémentaire à $\mathcal{A}_{\mathcal{F}_{\|+\infty}}$ (qui doit être déterminisé et complété si ce n'est pas encore le cas). Ainsi, si l'on sait traîter le cas existentiel et que l'on applique l'opération de complémentarisation à nouveau, on obtient un automate qui reconnait X_k.

On obtient ainsi un automate $\mathcal{A}_{\mathcal{F}}$, nous allons ici conclure en disant que $L(\mathcal{A}_{\mathcal{F}})$ est vide si et seuelement si F est fausse dans $\mathbb{N}$. En remontant une étape de construction (il y a au moins un quantificateur), on obtient un automate dont le langage est X_1 et on observe que pour tout mot dans X_1, il existe un mot de même longueur dans $\mathcal{A}_{\mathcal{F}}$. Et donc, il existe $x_1 \in \mathbb{N}$ tel que $F_1(x_1)$ est vraie alors si et seulement si X_1 est non vide, si et seulement si $L(\mathcal{A}_{\mathcal{F}})$ et si et seulement si $F = \exists x_1, F_1(x_1)$ est vraie dans $\mathbb{N}$. Il est facile de décider algorithmiquement si un automate a un langage vide ou non, et donc, le problème donné est bien décidable. $\square$