Équivalence entre logique monadique du second ordre et automates

Antoine Amarilli

23 décembre 2009

Résumé

L'objet de ce rapport est de démontrer le théorème affirmant qu'un langage est définissable dans la logique monadique du second ordre si et seulement s'il est régulier. Pour ce faire, on introduira d'abord le formalisme de la logique du premier ordre. On démontrera ensuite le théorème affirmant qu'un langage est définissable dans la logique du premier ordre si et seulement s'il est sans étoile. Ceci nous amènera à présenter la logique du second ordre, et à prouver qu'elle permet de définir tous les langages réguliers.

Table des matières

1 Logique du premier ordre

1.1 Définitions

Les définitions de cette section sont adaptées de [2] et [3]. On ne définit pas ici la notion de fonction, car on n'en a pas besoin pour les théorèmes qui nous intéressent.

1.1.1 Formules

On définit d'abord les langages en logique du premier ordre. Notons que ce concept n'est pas le même que celui utilisé dans le cours de langages formels. D'une certaine manière, ce que l'on appelle ici langage se rapproche plus des alphabets du cours de langages formels. En effet, on fournit ici les symboles de base qui vont nous permettre de construire termes et formules.

Définition 1 (Langage)**.** Un **langage** $\mathcal{L}$ est un couple formé d'un ensemble de constantes $c_1, ..., c_n, ...$ et d'un ensemble de relations $P_1, ..., P_n, ...$ où chaque symbole de relation a une arité. On suppose que l'on dispose du symbole de relation "$=$", d'arité 2, que l'on notera de façon infixe pour une meilleure lisibilité, et que l'on dispose du symbole de relation "T", d'arité 0, correspondant à l'énoncé toujours vrai.

Définissons à présent les termes et les formules pour un langage $\mathcal{L}$. On suppose que l'on dispose d'un ensemble de variables $x_1, ..., x_n, ...$

Définition 2 (Terme)**.** L'ensemble des **termes** du langage $\mathcal{L}$ est l'ensemble des variables et des constantes.

Définition 3 (Formule atomique)**.** L'ensemble des **formules atomiques** du langage $\mathcal{L}$ est l'ensemble des expressions de la forme $P(t_1, ..., t_k)$ pour $t_1, ..., t_k$ des termes de $\mathcal{L}$ et P un symbole de relation d'arité k de $\mathcal{L}$.

Définition 4 (Formule)**.** L'ensemble des **formules** du langage $\mathcal{L}$ est le plus petit ensemble contenant les formules atomiques et tel que pour toutes formules ϕ_1, ϕ_2 et toute variable x, $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\exists x\, \phi_1$ et $\forall x\, \phi_1$ sont des formules.

On définit à présent la notion de variable libre. De façon intuitive, les variables libres toutes celles qui ne sont pas quantifiées par un symbole $\exists$ ou $\forall$.

Définition 5 (Variable libre)**.** L'ensemble des variables libres d'un terme ou d'une formule sur le langage $\mathcal{L}$ est défini inductivement par les règles suivantes :

- Les variables libres d'un terme ou d'une formule atomique est l'ensemble des variables apparaissant dans ce terme ou dans cette formule atomique.
- Une formule de la forme $\phi_1 \wedge \phi_2$ ou $\phi_1 \vee \phi_2$ avec ϕ_1 et ϕ_2 des formules a pour variables libres celles de ϕ_1 et ϕ_2.
- Une formule de la forme $\neg\phi$ a pour variables libres celles de ϕ.
- Une formule de la forme $\exists x\, \phi$ ou $\forall x\, \phi$ avec x une variable et ϕ une formule a pour variables libres celles de ϕ à l'exception de x.

Définition 6 (Énoncé)**.** Un **énoncé** est une formule sans variables libres.

La définition des types, qui interviendra ultérieurement, va utiliser la notion de profondeur maximale de quantificateurs d'une formule. Celle-ci, indépendante de toute sémantique, peut être définie dès maintenant.

Définition 7 (Profondeur maximale de quantificateurs)**.** Soit ϕ une formule. On définit inductivement la **profondeur maximale de quantificateurs** de ϕ, notée $\mathrm{qr}(\phi)$, par les règles suivantes :
- $\mathrm{qr}(\phi) = 0$ pour une formule atomique ϕ.
- $\mathrm{qr}(\phi_1 \wedge \phi_2) = \mathrm{qr}(\phi_1 \vee \phi_2) = \max(\mathrm{qr}(\phi_1), \mathrm{qr}(\phi_2))$ pour des formules ϕ_1 et ϕ_2.
- $\mathrm{qr}(\neg\phi) = \mathrm{qr}(\phi)$ pour une formule ϕ.
- $\mathrm{qr}(\exists x\, \phi) = \mathrm{qr}(\forall x\, \phi) = \mathrm{qr}(\phi) + 1$ pour une formule ϕ et une variable x.

On utilise la notation FO[k] pour désigner toutes les formules de la logique du premier ordre dont la profondeur maximale de quantificateurs est inférieure ou égale à k.

1.1.2 Structures

Les notions définies jusqu'à présent ne concernent que des propriétés syntaxiques des formules logiques et ne leur confèrent aucune valeur sémantique. Pour leur en fournir une, on définit à présent le concept de structure.

Définition 8 (Structure)**.** Soit $\mathcal{L}$ un langage. Une **$\mathcal{L}$-structure** est un triplet $\mathfrak{A} = (A, \{c_i^{\mathfrak{A}}\}, \{P_i^{\mathfrak{A}}\})$ formé d'un ensemble A appelé **univers**, et de deux ensembles qui sont l'**interprétation** des éléments du langage dans la $\mathcal{L}$-structure : on demande qu'à toute constante c_i de $\mathcal{L}$ corresponde un élément $c_i^{\mathfrak{A}} \in A$ et qu'à toute relation P_i de $\mathcal{L}$ d'arité k corresponde une relation $P_i^{\mathfrak{A}} \in A^k$ représentée par son graphe. On suppose aussi que $=^{\mathfrak{A}}$ est l'égalité usuelle dans A, c'est-à-dire que le graphe de $=^{\mathfrak{A}}$ est $\{(x, x) \,|\, x \in A\}$.

Définition 9 (Sous-structure)**.** Soit $\mathfrak{A}$ une $\mathcal{L}$-structure d'univers A, et $B \subset A$ contenant les interprétations des constantes de $\mathcal{L}$ dans $\mathfrak{A}$. On peut alors définir $\mathfrak{B}$ la **sous-structure de $\mathfrak{A}$ d'univers** B en interprétant les constantes de $\mathcal{L}$ de la même manière que dans $\mathfrak{A}$ et en interprétant les relations de $\mathcal{L}$ par la restriction à B de leur interprétation dans $\mathfrak{A}$.

On va maintenant être en mesure de définir la satisfaction d'une formule dans une structure. Pour ce faire, définissons d'abord l'interprétation des termes dans une structure.

Définition 10 (Interprétation)**.** L'**interprétation** dans une $\mathcal{L}$-structure $\mathfrak{A}$ d'un terme $t(\underline{x})$ de variables libres $\underline{x} = (x_1, ..., x_n)$ pour $\underline{a} = (a_1, ..., a_n) \in A^n$, notée $t^{\mathfrak{A}}(\underline{a})$, est définie de la façon suivante :
- L'interprétation d'une constante c est $c^{\mathfrak{A}}$.
- L'interprétation d'une variable x_i est a_i.

Définissons à présent la satisfaction d'une formule ϕ.

Définition 11 (Satisfaction d'une formule)**.** Pour $\phi(\underline{x})$ une formule sur $\mathcal{L}$ de variables libres $\underline{x} = (x_1, ..., x_n)$, pour $\underline{a} = (a_1, ..., a_n) \in A^n$ où $\mathfrak{A}$ est une $\mathcal{L}$-structure, on dit que $\underline{a}$ **satisfait** ϕ **dans** $\mathfrak{A}$, ce que l'on note $\mathfrak{A} \models \phi(\underline{a})$, dans les cas suivants :

- Si ϕ est de la forme $P(t_1, ..., t_k)$ où $t_1, ..., t_k$ sont des termes et P un symbole de relation d'arité k de $\mathcal{L}$, on a $\mathfrak{A} \models \phi(\underline{a})$ si et seulement si $(t_1^{\mathfrak{A}}(\underline{a}), ..., t_k^{\mathfrak{A}}(\underline{a})) \in P^{\mathfrak{A}}$.
- Si ϕ est de la forme $\neg\phi_1$ (resp. $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$) où ϕ_1 et ϕ_2 sont des formules, on a $\mathfrak{A} \models \phi(\underline{a})$ si et seulement si l'on n'a pas $\mathfrak{A} \models \phi_1(\underline{a})$ (resp. si l'on a $\mathfrak{A} \models \phi_1(\underline{a})$ et $\mathfrak{A} \models \phi_2(\underline{a})$, si l'on a $\mathfrak{A} \models \phi_1(\underline{a})$ ou $\mathfrak{A} \models \phi_2(\underline{a})$).
- Si ϕ est de la forme $\exists x\, \phi_1(x, \underline{x})$ (resp. $\forall x\, \phi_1(x, \underline{x})$) où ϕ_1 est une formule et x une variable, on a $\mathfrak{A} \models \phi(\underline{a})$ si et seulement s'il existe un $x \in A$ tel que (resp. pour tout $x \in A$ on a) $\mathfrak{A} \models \phi_1(x, \underline{a})$.

1.1.3 Types

À l'aide du concept de profondeur maximale de quantificateurs, définissons à présent la notion de type. (On ne s'intéresse qu'aux types pour des énoncés, car c'est tout ce dont nous aurons besoin par la suite.)

Définition 12 (Type)**.** On fixe un langage $\mathcal{L}$. Soit $\mathfrak{A}$ une $\mathcal{L}$-structure d'univers A, k un entier. Le **type de** $\mathfrak{A}$ de rang k, noté $\mathrm{tp}_k(\mathfrak{A})$, est défini comme l'ensemble $\{\phi \in \mathrm{FO}[k] \mid \mathfrak{A} \models \phi\}$.

On note $\mathfrak{A}_1 \equiv_k \mathfrak{A}_2$ si $\mathrm{tp}_k(\mathfrak{A}_1) = \mathrm{tp}_k(\mathfrak{A}_2)$. Par définition du type, on a alors, pour tout énoncé $\phi \in \mathrm{FO}[k]$, $\mathfrak{A}_1 \models \phi$ si et seulement si $\mathfrak{A}_2 \models \phi$.

Démontrons que dans un langage fini $\mathcal{L}$, pour tout k, il n'existe qu'un nombre fini de types de rang k (à équivalence logique près). On démontre d'abord le théorème suivant (théorème 3.13 p. 33 de [2]).

Théorème 13. *Pour un langage $\mathcal{L}$ fini fixé, pour tout k, pour tout m,* $\mathrm{FO}[k]$ *ne contient qu'un nombre fini de formules ayant $\underline{x} = x_1, ..., x_m$ pour variables libres, à équivalence logique près.*

Démonstration. On procède par récurrence sur k.

Cas de base. On fixe m. Les éléments de FO[0] sont les combinaisons booléennes de formules atomiques. Comme le nombre de variables, de constantes et de relations possibles est fini (on a m variables libres, et $\mathcal{L}$ est fini), il n'y a qu'un nombre fini de formules atomiques possibles. Il n'y a donc, à équivalence logique près, qu'un nombre fini de combinaisons booléennes de ces formules atomiques. En effet, il n'y a qu'un nombre fini de disjonctions non équivalentes des formules atomiques et de leurs négations (on exploite le fait que $a \vee a \equiv a$). Il n'y a donc qu'un nombre fini de conjonctions non équivalentes de telles disjonctions (puisque $a \wedge a \equiv a$). Comme toutes les combinaisons booléennes des formules atomiques peuvent s'écrire en forme normale conjonctive, le résultat est établi.

Récurrence. Soit k un entier naturel. Supposons que $\mathrm{FO}[k]$ est fini pour tout m. Fixons m, et démontrons le résultat pour $\mathrm{FO}[k+1]$. Les formules de $\mathrm{FO}[k+1]$ sont les combinaisons booléennes de formules de $\mathrm{FO}[k]$ et de formules du type $Qx_{m+1}\, \phi(\underline{x}, x_{m+1})$ pour $\phi \in \mathrm{FO}[k]$. Mais, par hypothèse de récurrence, il n'existe qu'un nombre fini de formules de $\mathrm{FO}[k]$ à k ou $k+1$ variables libres à équivalence près. Par le même raisonnement que pour le cas de base, il n'existe donc qu'un nombre fini de combinaisons booléennes de telles formules à équivalence près. Ainsi, $\mathrm{FO}[k+1]$ est également fini. □

On en déduit le résultat suivant (théorème 3.15 p. 34 de [2]) :

Théorème 14. *Pour un langage $\mathcal{L}$ fini fixé, pour tout k, le nombre de types de rang k est fini.*

Démonstration. Soient $\mathcal{L}$ un tel langage et k un entier. D'après le théorème précédent, FO[k] est fini à équivalence logique près : notons $\phi_1, ..., \phi_n$ des représentants des classes d'équivalence. Notons que par définition des types, si pour une formule ϕ, $\phi \in \mathrm{tp}_k(\mathfrak{A})$ alors $\psi \in \mathrm{tp}_k(\mathfrak{A})$ pour toute formule ψ équivalente à ϕ. Pour cette raison, pour toute $\mathcal{L}$-structure $\mathfrak{A}$, $\mathrm{tp}_k(\mathfrak{A})$ est entièrement défini par l'ensemble $K_{\mathrm{tp}_k(\mathfrak{A})}$ des ϕ_i qui y appartiennent. Mais il n'existe qu'un nombre fini de tels ensembles, vu que le nombre de ϕ_i est fini, donc le nombre de types de rang k possibles est également fini. □

On a également les corollaires suivants :

Corollaire 15. *Pour tout type τ de rang k, il existe une formule $\psi_\tau \in$ FO[k] telle que pour toute $\mathcal{L}$-structure $\mathfrak{A}$, $\mathfrak{A} \models \psi_\tau$ si et seulement si $\mathrm{tp}_k(\mathfrak{A}) = \tau$.*

Démonstration. Avec les notations précédentes, on pose :

$$\psi_\tau = \bigwedge_{i \in K_\tau} \phi_i \wedge \bigwedge_{i \notin K_\tau} \neg \phi_i$$

D'après ce qui précède, comme K_τ caractérise le type τ parmi les types de rang k, c'est également le cas de ψ_τ. Notons enfin que comme les ϕ_i sont dans FO[k], ψ_τ l'est également puisque l'on n'a pas rajouté de quantificateur. □

Corollaire 16. *Toute formule ϕ de* FO[k] *est équivalente à une disjonction de formules de la forme ψ_τ.*

Démonstration. Avec les notations précédentes, toute formule de FO[k] est équivalente à une formule ϕ_i, et celle-ci est la disjonction des formules ψ_τ du corollaire précédent pour tous les types τ dont l'ensemble K_τ est tel que $i \in K_\tau$. □

Ces corollaires nous amènent à définir la notion de type cohérent dont nous aurons besoin par la suite.

Définition 17 (Type cohérent)**.** Pour une formule ϕ de FO[k], l'ensemble des **types cohérents avec** ϕ est l'ensemble des types τ tels que $\phi \in \tau$. Ce sont exactement les types τ_i tels que $\phi = \bigvee_i \psi_{\tau_i}$.

1.1.4 Définissabilité

On définit enfin la notion de définissabilité d'un langage formel (d'après [2], p. 122).

Soit Σ un alphabet. On définit le langage $\mathcal{L}_\Sigma$ (au sens de la logique) comme comprenant le symbole binaire $<$ et des symboles de relation unaire P_a pour tout $a \in \Sigma$. À tout mot $u \in \Sigma^*$ de longueur n, on associe une $\mathcal{L}_\Sigma$-structure $\mathfrak{M}_u$ dont l'univers est $\{1, ..., n\}$. On y interprète $<$ comme l'ordre sur les entiers naturels, et P_a comme la relation dont le graphe est l'ensemble des positions où la lettre a apparaît dans u.

Dans la suite, on écrira parfois $x \leq y$ comme abréviation pour $x < y \vee x = y$.

Exemple 18. Considérons $\Sigma = \{a, b, c\}$ et $u = aba$. La structure $\mathfrak{M}_u$ a pour univers $\{1, 2, 3\}$, on a $<^{\mathfrak{M}_u} = \{(1, 2), (1, 3), (2, 3)\}$, $P_a^{\mathfrak{M}_u} = \{1, 3\}$, $P_b^{\mathfrak{M}_u} = \{2\}$ et $P_c^{\mathfrak{M}_u} = \emptyset$.

À un énoncé Φ sur $\mathcal{L}_\Sigma$, on associe alors le langage formel $L(\Phi) = \{u \in \Sigma^* \mid \mathfrak{M}_u \models \Phi\}$. On dit qu'un langage formel L est **définissable** s'il existe un énoncé Φ tel que $L = L(\Phi)$.

Exemple 19. Posons $\Sigma = \{a, b, c\}$. Posons $\Phi \equiv \exists x\, P_c(x)$. On a alors $L(\Phi) = \Sigma^* \backslash (a + b)^*$.

Exemple 20. Sur le même alphabet, posons :

$$\Phi \equiv \exists x\, \exists y\ (P_a(x) \wedge P_b(y) \wedge x < y \wedge \neg\,(\exists z\,(x < z \wedge z < y)))$$

On a alors $L(\Phi) = \Sigma^* ab \Sigma^*$.

1.2 Définissabilité des langages sans étoile

La logique du premier ordre permet déjà de rendre certains langages définissables. Plus précisément, elle permet exactement de définir les langages sans étoile. Énonçons ce résultat (théorème 7.26 p. 126 de [2]) et démontrons-le.

Théorème 21. *Un langage est définissable dans la logique du premier ordre si et seulement s'il est sans étoile.*

Démonstration. On note Σ l'alphabet. Montrons dans un premier temps comment construire à partir d'un langage formel sans étoile L un énoncé Φ de $\mathcal{L}_\Sigma$ qui définit L. Montrons ensuite que les langages formels que définissent les énoncés de $\mathcal{L}_\Sigma$ de la logique du premier ordre sont sans étoile.

Définissabilité des langages sans étoile. Procédons par induction sur le langage à partir de la définition des langages sans étoile.

Cas de base. Le langage vide est définissable par $\neg T$ et pour toute lettre $a \in \Sigma$, le langage $\{a\}$ est définissable par la formule $(\forall x\, \forall y\, (x = y)) \wedge (\forall x\, P_a(x))$.

Complémentation. Si le langage L est définissable par un énoncé Φ, le langage $\overline{L}$ est définissable par $\neg\Phi$.

Union. Si L_1 et L_2 sont définissables respectivement par des énoncés Φ_1 et Φ_2, le langage $L_1 + L_2$ est définissable par $\Phi_1 \vee \Phi_2$.

Produit. Supposons que L_1 et L_2 sont définissables respectivement par des énoncés Φ_1 et Φ_2. Construisons une formule permettant de définir $L = L_1 \cdot L_2$.

Soit x une variable n'apparaissant pas dans Φ_1 ou Φ_2. On pose $\phi_1(x)$ la formule obtenue à partir de Φ_1 en remplaçant par induction toutes les sous-formules de la forme $\exists y\, \phi$ ou $\forall y\, \phi$ par $\exists y\, ((y \leq x) \wedge \phi)$ ou $\forall y\, ((y \leq x) \wedge \phi)$ respectivement. Il est clair que pour tout mot u sur Σ, pour tout p, $1 \leq p \leq |u|$, une structure $\mathfrak{M}_u$ est telle que $\mathfrak{M}_u \models \phi_1(p)$ si et seulement si $\mathfrak{M}_u^{\leq p} \models \Phi_1(p)$, où $\mathfrak{M}_u^{\leq p}$ désigne la sous-structure de $\mathfrak{M}_u$ sur le domaine $\{1, ..., p\}$.

De la même manière, on pose $\phi_2(x)$ la formule obtenue à partir de Φ_2 en ajoutant à chaque quantificateur la condition $y > x$. Comme dans le cas précédent, pour tout mot u sur Σ, pour tout p, $1 \leq p \leq |u|$, une structure $\mathfrak{M}_u$

est telle que $\mathfrak{M}_u \models \phi_2(p)$ si et seulement si $\mathfrak{M}_u^{>p} \models \Phi_2(p)$, où $\mathfrak{M}_u^{>p}$ désigne la sous-structure de $\mathfrak{M}_u$ sur le domaine $\{p+1, ..., |u|\}$.

Or un mot u appartient à L si et seulement s'il peut s'écrire sous la forme $u = vw$ avec $v \in L_1$ et $w \in L_2$, ou, de façon équivalente, s'il existe une position p telle que $u_{1\cdots p} \in L_1$ et $u_{p+1\cdots|u|} \in L_2$. Pour cette raison, la formule $\Phi = \exists x\, \phi_1(x) \wedge \phi_2(x)$ est telle que $L(\Phi) = L$.

Caractère sans étoile des langages définissables. Soit Φ un énoncé de $\mathcal{L}_\Sigma$ définissant un langage $L(\Phi)$. On veut montrer que $L(\Phi)$ est sans étoile. On raisonne par récurrence sur la profondeur maximale de quantificateurs de Φ. Pour amorcer la récurrence, on ajoute au langage une constante max, à interpréter dans les $\mathfrak{M}_s$ comme le plus grand élément de l'univers. Elle est définissable par la logique du premier ordre, donc cela ne change pas les langages définissables par la logique du premier ordre.

Cas de base. Les seuls énoncés de profondeur maximale de quantificateurs 0 sont T et $\neg T$, définissant le langage vide et le complémentaire du langage vide qui sont bien sans étoile, ainsi que des combinaisons booléennes de formules atomiques de la forme $P_a(max)$. Comme $P_a(max)$ définit A^*a pour tout $a \in \Sigma$, qui est sans étoile ($A^*a = \overline{\emptyset} \cdot \{a\}$), et que les langages sans étoile sont clos par union, intersection et complémentation, on obtient bien des langages sans étoile.

Récurrence. Soit k un entier. Supposons le résultat acquis pour les énoncés de FO$[k]$. Considérons un énoncé de FO$[k+1]$. Il s'écrit comme une combinaison booléennes de formules de FO$[k]$ et de formules du type $Qx\, \phi(x)$ pour $\phi \in$ FO$[k]$. Comme les langages sans étoile sont stables par union, par intersection et par complémentation, il suffit de démontrer que chaque terme de la combinaison booléenne définit un langage sans étoile. C'est clair pour les formules de FO$[k]$ par hypothèse de récurrence. Pour les autres, comme les langages sans étoile sont stables par complémentation, on peut, quitte à étudier la négation, supposer que le quantificateur est existentiel. Ainsi, sans perte de généralité, on suppose que $\Phi = \exists x\, \phi(x)$ avec $\phi \in$ FO$[k]$.

Notons $\tau_0, ..., \tau_n$ les types de rang k de $\mathcal{L}_\Sigma$ (il n'en existe qu'un nombre fini d'après le théorème 14). Notons $\Psi_0, ..., \Psi_n$ les énoncés de FO$[k]$ les caractérisant (qui existent d'après le corollaire 15). Par hypothèse de récurrence, les $L(\Psi_i)$ sont sans étoile. On pose :

$$S_\Phi = \left\{ (\tau_i, \tau_j) \,\middle|\, \exists s \in \Sigma^*, \exists p \in \mathbb{N}, \left((\mathfrak{M}_s \models \phi(p)) \wedge \left(\operatorname*{tp}_k(\mathfrak{M}_s^{\leq p}) = \tau_i \right) \wedge \left(\operatorname*{tp}_k(\mathfrak{M}_s^{>p}) = \tau_j \right) \right) \right\}$$

Attention : les symboles $\exists$ et $\wedge$ sont uniquement utilisés comme abréviations ici. La condition imposée sur (τ_i, τ_j) n'est pas une formule de la logique du premier ordre.

On va montrer que $\mathfrak{M}_s \models \Phi$ si et seulement s'il existe une position p de s telle que $\operatorname{tp}_k(\mathfrak{M}_s^{\leq p}) = \tau_i$ et $\operatorname{tp}_k(\mathfrak{M}_s^{>p}) = \tau_j$ pour $(\tau_i, \tau_j) \in S_\Phi$ (autrement dit, $\mathfrak{M}_s^{\leq p} \models \Psi_i$ et $\mathfrak{M}_s^{>p} \models \Psi_j$). Notons que ce résultat entraîne le résultat demandé. En effet, on aurait alors :

$$L(\Phi) = \bigcup_{(\tau_i, \tau_j) \in S_\Phi} L(\Psi_i) L(\Psi_j)$$

Ainsi, $L(\Phi)$ serait sans étoile comme union de concaténations de langages sans étoile.

Supposons que $\mathfrak{M}_s \models \Phi$. Alors, comme Φ s'écrit $\exists x\, \phi(x)$, il existe une position p telle que $\mathfrak{M}_s \models \phi(p)$, et l'existence d'une paire (τ_i, τ_j) vérifiant les conditions demandées est immédiate (il suffit de prendre $\tau_i = \mathrm{tp}_k(\mathfrak{M}_s^{\leq p})$ et $\tau_j = \mathrm{tp}_k(\mathfrak{M}_s^{>p})$).

Réciproquement, supposons qu'il existe une position p telle que $\mathrm{tp}_k(\mathfrak{M}_s^{\leq p}) = \tau_i$ et $\mathrm{tp}_k(\mathfrak{M}_s^{>p}) = \tau_j$ pour $(\tau_i, \tau_j) \in \mathfrak{M}_\Phi$. Par définition de S_Φ, il existe une chaîne s' et une position p' telle que $\mathrm{tp}_k(\mathfrak{M}_{s'}^{\leq p'}) = \tau_i$ et $\mathrm{tp}_k(\mathfrak{M}_{s'}^{>p'}) = \tau_j$. Mais on a alors $\mathrm{tp}_k(\mathfrak{M}_s^{\leq p}) = \mathrm{tp}_k(\mathfrak{M}_{s'}^{\leq p'})$ et $\mathrm{tp}_k(\mathfrak{M}_s^{>p}) = \mathrm{tp}_k(\mathfrak{M}_{s'}^{>p'})$, soit $\mathfrak{M}_s^{\leq p} \equiv_k \mathfrak{M}_{s'}^{\leq p'}$ et $\mathfrak{M}_s^{>p} \equiv_k \mathfrak{M}_{s'}^{>p'}$. En appliquant le lemme 3.7 de [2], on en déduit que $(\mathfrak{M}_s, p) \equiv_k (\mathfrak{M}_{s'}, p')$. Ainsi, comme $\phi \in \mathrm{FO}[k]$, comme $\mathfrak{M}_{s'} \models \phi(p')$, on a bien $\mathfrak{M}_s \models \phi(p)$ d'où $\mathfrak{M}_s \models \Phi$. □

Ce résultat montre en particulier que la logique du premier ordre ne suffit pas à définir tous les langages rationnels, puisque l'on sait d'après le théorème de Schützenberger (théorème 1.146 p. 67 de [1]) qu'il existe des langages qui ne sont pas sans étoile. Cela nous amène à définir une logique plus expressive : la logique du second ordre.

2 Logique du second ordre

2.1 Définitions

Les définitions de cette section sont adaptées de [2].

La logique du second ordre va nous permettre de quantifier non seulement sur les éléments de l'univers, mais aussi sur les sous-ensembles de l'univers ou les sous-ensembles de puissances de l'univers. Définissons-la formellement.

2.1.1 Formules

La notion de langage est inchangée par rapport à la logique du premier ordre. On fixe un langage $\mathcal{L}$. On suppose que l'on dispose d'un ensemble de variables du premier ordre $x_1, ..., x_n, ...$ et que pour tout $k > 0$, il existe un ensemble de variables du second ordre $X_1^k, ..., X_n^k, ...$ représentant des relations d'arité k.

Définition 22 (Formule en logique du second ordre)**.** On définit les **formules** de la logique du second ordre de la façon suivante :

- Les termes de la logique du second ordre sont ceux de la logique du premier ordre.
- Les formules atomiques de la logique du second ordre sont les formules atomiques de la logique du premier ordre et les expressions de la forme $X_i^k(t_1, ..., t_k)$ où $t_1, ..., t_k$ sont des termes et X_i^k une variable du second ordre d'arité k.
- L'ensemble des formules de la logique du second ordre est le plus petit ensemble contenant les formules atomiques et tel que pour toutes formules ϕ_1, ϕ_2, pour toute variable du premier ordre x et pour toute variable du second ordre X, $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\exists x\, \phi_1$, $\forall x\, \phi_1$, $\exists X\, \phi_1$ et $\forall X\, \phi_2$ sont des formules.

Définition 23 (Variables libres en logique du second ordre)**.** Les **variables libres du premier ordre** d'une formule de la logique du second ordre sont

obtenus avec les mêmes règles que pour les formules de la logique du premier ordre.

Les **variables libres du second ordre** d'une formule de la logique du second ordre sont définies de la façon suivante :

- Une formule atomique de la logique du premier ordre n'a pas de variable libre du second ordre.
- Une formule atomique de la forme $X_i^k(t_1, ..., t_k)$ où $t_1, ..., t_k$ sont des termes et X_i^k une variable du second ordre d'arité k a pour variable libre du second ordre X_i^k.
- Une formule de la forme $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\exists x\, \phi_1$, $\forall x\, \phi_1$ avec ϕ_1 et ϕ_2 des formules et x une variable du premier ordre a pour variables libres du second ordre celles de ϕ_1 et de ϕ_2.
- Une formule de la forme $\neg\phi$ a pour variables libres celles de ϕ.
- Une formule de la forme $\exists X\, \phi$ ou $\forall X\, \phi$ avec X une variable du second ordre et ϕ une formule a pour variables libres celles de ϕ à l'exception de X.

2.1.2 Structures

La notion de structure est inchangée par rapport à la logique du premier ordre. La notion d'interprétation d'un terme est également inchangée.

Définissons à présent la satisfaction d'une formule ϕ.

Définition 24 (Satisfaction d'une formule en logique du second ordre)**.** Pour $\phi(\underline{x}, \underline{X})$ une formule du second ordre sur $\mathcal{L}$ de variables libres du premier ordre $\underline{x} = (x_1, ..., x_n)$ et de variables libres du second ordre $\underline{X} = (X_1, ..., X_m)$ d'arités $n_1, ..., n_m$ respectivement, pour $\underline{b} = (b_1, ..., b_n) \in A^n$, pour $\underline{B} = (B_1, ..., B_m)$ avec $B_i \subset A^{n_i}$ pour tout i, où $\mathfrak{A}$ est une $\mathcal{L}$-structure, on dit que $(\underline{a}, \underline{B})$ **satisfait** ϕ **dans** $\mathfrak{A}$, ce que l'on note $\mathfrak{A} \models \phi(\underline{a}, \underline{B})$, dans les cas suivants :

- Si $\phi(\underline{x}, X)$ est de la forme $X(t_1, ..., t_k)$ où $t_1, ..., t_k$ sont des termes et X une variable du second ordre d'arité k de $\mathcal{L}$, on a $\mathfrak{A} \models \phi(\underline{a}, B)$ si et seulement si $(t_1^{\mathfrak{A}}(\underline{a}), ..., t_k^{\mathfrak{A}}(\underline{a})) \in B$.
- Si $\phi(\underline{x}, \underline{X})$ est de la forme $\exists Y\, \psi(\underline{x}, Y, \underline{X})$ (resp. $\forall Y\, \psi(\underline{x}, Y, \underline{X})$) où Y est une variable du second ordre d'arité k de $\mathcal{L}$, on a $\mathfrak{A} \models \phi(\underline{a}, \underline{B})$ si et seulement s'il existe un $C \subset A^k$ tel que (resp. pour tout $C \subset A^k$ on a) $\mathfrak{A} \models \psi(\underline{a}, C, \underline{B})$.
- Dans les autres cas, on applique les mêmes règles qu'en logique du premier ordre.

En fait, on verra que la logique monadique du second ordre est un sous-ensemble suffisamment fort de la logique du second ordre pour rendre définissables tous les langages réguliers. En voici la définition.

Définition 25 (Logique monadique du second ordre)**.** La **logique monadique du second ordre** est définie comme la restriction de la logique du second ordre où les variables du second ordre sont toutes d'arité 1.

Dans la suite, on ne travaillera qu'avec la logique monadique du second ordre.

2.1.3 Types

Dans l'ensemble, la notion de type en logique monadique du second ordre ne change guère par rapport à la logique du premier ordre ; on se contente

de donner de nouvelles notations. La définition de la profondeur maximale de quantificateurs est inchangée (les quantificateurs du second ordre comptent de la même manière que ceux du premier ordre). On note MSO[k] les formules de profondeur maximale de quantificateurs inférieure ou égale à k. La définition des types est inchangée, si ce n'est que l'on travaille à présent avec des formules de la logique monadique du second ordre. On note mso-tp$_k(\mathfrak{A})$ le MSO-type de rang k de $\mathfrak{A}$.

Les théorèmes 13 et 14, ainsi que les corollaires 15 et 16 s'étendent à la logique monadique du second ordre. Ils se démontrent de la même manière que dans le cas de la logique du premier ordre.

2.1.4 Définissabilité

La notion de définissabilité est bien sûr inchangée, à ceci près que l'on utilise à présent des énoncés de la logique monadique du second ordre.

2.2 Définissabilité des langages réguliers

Il est maintenant temps de démontrer le théorème qui nous intéresse (théorème 7.21 p. 122 de [2]).

Pour ce faire, on a besoin du lemme suivant (lemme 7.11 p. 116 de [2]), que l'on admettra.

Lemme 26. *Soit $\mathcal{L}$ un langage, soient $\mathfrak{A}_1, \mathfrak{A}_2, \mathfrak{B}_1, \mathfrak{B}_2$ des $\mathcal{L}$-structures, et soient $\mathfrak{A}$ l'union disjointe de $\mathfrak{A}_1$ et de $\mathfrak{A}_2$, $\mathfrak{B}$ l'union disjointe de $\mathfrak{B}_1$ et de $\mathfrak{B}_2$. Alors, si $\mathfrak{A}_1 \equiv_k \mathfrak{B}_1$ et $\mathfrak{A}_2 \equiv_k \mathfrak{B}_2$, alors $\mathfrak{A} \equiv_k \mathfrak{B}$.*

Théorème 27 (Büchi)**.** *Un langage est définissable dans la logique monadique du second ordre si et seulement s'il est régulier.*

Démonstration. On note Σ l'alphabet. Montrons dans un premier temps comment construire à partir d'un langage régulier L un énoncé Φ de $\mathcal{L}_\Sigma$ qui définit L. Montrons ensuite que les langages que définissent les énoncés de $\mathcal{L}_\Sigma$ de la logique monadique du second ordre sont réguliers.

Définissabilité des langages réguliers. Soit L un langage régulier, et posons $\mathcal{A} = (Q, \delta, q_0, F)$ un automate déterministe reconnaissant L (où $Q = q_0, ..., q_{m-1}$ est l'ensemble des états, δ désigne la fonction de transition, q_0 l'état initial, F les états finaux).

Démarche générale. Construisons à présent une formule logique Φ telle que $L(\Phi) = L$. Nous allons utiliser des variables du second ordre $X_0, ..., X_{m-1}$ pour représenter l'ensemble des positions du mot où l'automate se trouve dans l'état 0, ..., $m-1$ respectivement (ce sont bien des sous-ensembles de l'univers, puisque l'univers des $\mathcal{L}_\Sigma$-structures que nous allons considérer est justement l'ensemble des positions du mot qu'elles représentent). Un mot est alors reconnu par l'automate si et seulement s'il existe des valeurs de $X_0, ...X_{m-1}$ représentant une exécution de l'automate sur son entrée qui respecte les règles de fonctionnement de l'automate et se termine à un état final, ce que nous allons imposer avec une formule logique ϕ.

On pose donc :

$$\Phi \equiv \exists X_0 \cdots \exists X_{m-1}\, \phi$$

Pour que les X_i représentent une exécution correcte de l'automate, il faut imposer que l'exécution commence en q_0, que le passage d'un état à un autre respecte la fonction de transition, que l'exécution se termine dans un état final, et que les X_i forment une partition des positions (ie. dans chaque position, on est à un et un seul état).

Construction détaillée de la formule. On construit donc :

$$\begin{aligned}
\phi &\equiv \phi_{\text{part}} \wedge \phi_{\text{init}} \wedge \phi_{\text{trans}} \wedge \phi_{\text{accepte}} \\
\phi_{\text{part}} &\equiv \forall x \left(\left(\bigvee_{i=0}^{m-1} X_i(x) \right) \wedge \neg \left(\bigvee_{0 \leq i < j \leq m-1} X_i(x) \wedge X_j(x) \right) \right) \\
\phi_{\text{init}} &\equiv \forall x \left((\forall y \, (x \leq y)) \Rightarrow \bigwedge_{a \in \Sigma} \left(P_a(x) \Rightarrow X_{\hat{\delta}(q_0,a)}(x) \right) \right) \\
\phi_{\text{trans}} &\equiv \forall x \, \forall y \left((x \prec y) \Rightarrow \bigwedge_{i=0}^{m-1} \bigwedge_{a \in \Sigma} \left(X_i(x) \wedge P_a(y) \Rightarrow X_{\delta(q_i,a)}(y) \right) \right) \\
\phi_{\text{accepte}} &\equiv \forall x \left((\forall y \, (y \leq x)) \Rightarrow \bigvee_{q_i \in F} X_i(x) \right)
\end{aligned}$$

On a utilisé les abréviations suivantes :

$$\begin{aligned}
x \Rightarrow y &\equiv \neg(x \wedge \neg y) \\
\hat{\delta}(q_i, a) &= j \text{ tel que } \delta(q_i, a) = q_j \\
x \prec y &\equiv x < y \wedge \neg(\exists z \, (x < z \wedge z < y))
\end{aligned}$$

Explications. La formule ϕ_{part} impose que les X_i forment une partition des positions. Elle peut se lire : « pour toute position, l'automate après la lecture de la lettre à cette position se trouve en un état et un seul ».

La formule ϕ_{init} impose que l'automate commence à l'état initial. Elle peut se lire : « pour toute position, si cette position est la première, alors pour toute lettre, si cette lettre est à la position considérée, l'automate se trouve dans l'état où il doit se trouver après être parti de l'état initial et avoir lu cette lettre », soit de façon plus concise « après avoir lu la première lettre, l'automate se trouve dans l'état correspondant à la lecture de cette lettre depuis l'état initial ».

La formule ϕ_{trans} impose que l'automate respecte les transitions. Elle peut se lire de façon concise : « pour tout couple de positions consécutives, si l'automate se trouve dans l'état q_i à la première position du couple et a lu la lettre a, alors, à la deuxième position du couple, il se trouve dans l'état $\delta(q_i, a)$ ».

La formule ϕ_{accepte} impose que l'automate termine son exécution dans un état final. Elle peut se lire : « pour toute position, si cette position est la dernière, alors l'automate se trouve dans un des états finaux ».

Conclusion. Pour tout mot u, $u \in L$ si et seulement s'il existe une exécution de $\mathcal{A}$ sur u terminant dans un état final, donc si et seulement s'il existe des variables du second ordre $X_0, ..., X_{m-1}$ vérifiant ϕ, donc si et seulement si $u \in L(\Phi)$.

Caractère régulier des langages définissables. Soit Φ un énoncé de $\mathcal{L}_\Sigma$. On va construire un automate reconnaissant $L(\Phi)$. Posons $k = \mathrm{qr}(\Phi)$. Notons $\tau_0, ..., \tau_n$ les MSO-types de rang k de $\mathcal{L}_\Sigma$, et $\Psi_0, ..., \Psi_n$ les énoncés de MSO$[k]$ les caractérisant. On pose F l'ensemble des types cohérents avec Φ, de sorte que Φ soit équivalente à $\bigvee_{\tau_i \in F} \Psi_i$. On suppose que l'on a numéroté les types de sorte que $\tau_0 = \text{mso-tp}_k(\mathfrak{M}_\epsilon)$.

Construction de l'automate. On construit à présent l'automate $\mathcal{A} = (\{\tau_0, ..., \tau_n\}, E, \tau_0, F)$ où $\{\tau_0, ..., \tau_n\}$ est l'ensemble des états, τ_0 l'état initial, F les états finaux, et E l'ensemble des transitions défini par la relation suivante :

$$(\tau_i, a, \tau_j) \in E \iff \exists s \in \Sigma^*, \left(\underset{k}{\text{mso-tp}}(\mathfrak{M}_s) = \tau_i \wedge \underset{k}{\text{mso-tp}}(\mathfrak{M}_{s \cdot a}) = \tau_j \right)$$

Attention : comme précédemment, les symboles logiques sont uniquement utilisés comme abréviations ici.

Montrons que l'automate $\mathcal{A}$ est déterministe complet.

Caractère déterministe. Soit τ_i un type, $a \in \Sigma$. Supposons qu'il existe $s_1, s_2 \in \Sigma^*$ telles que $\text{mso-tp}_k(\mathfrak{M}_{s_1}) = \text{mso-tp}_k(\mathfrak{M}_{s_2}) = \tau_i$, $\text{mso-tp}_k(\mathfrak{M}_{s_1 \cdot a}) = \tau_{j_1}$ et $\text{mso-tp}_k(\mathfrak{M}_{s_2 \cdot a}) = \tau_{j_2}$. On a alors $\mathfrak{M}_{s_1} \equiv_k \mathfrak{M}_{s_2}$. En appliquant le lemme 26, on en déduit que $\mathfrak{M}_{s_1 \cdot a} \equiv_k \mathfrak{M}_{s_2 \cdot a}$, d'où $\tau_{j_1} = \tau_{j_2}$.

Caractère complet. Pour démontrer le caractère complet, on va expliciter l'état auquel aboutit la lecture d'un mot. Démontrons par récurrence sur la longueur du mot que la lecture d'un mot u nous conduit à l'état $\text{mso-tp}_k(\mathfrak{M}_u)$.

La lecture de ϵ nous conduit à l'état $\tau_0 = \text{mso-tp}_k(\mathfrak{M}_\epsilon)$.

Soit n un entier naturel. Supposons que la lecture d'un mot u de longueur n nous conduise à l'état $\text{mso-tp}_k(\mathfrak{M}_u)$. Soit v un mot de longueur $n+1$. Il s'écrit $u \cdot x$ avec u un mot de longueur n et $x \in \Sigma$. La lecture de u, par hypothèse de récurrence, nous conduit à l'état $\tau_i = \text{mso-tp}_k(\mathfrak{M}_u)$. En choisissant $s = u$, on a bien $\text{mso-tp}_k(\mathfrak{M}_s) = \tau_i$ par définition, et $\text{mso-tp}_k(\mathfrak{M}_{s \cdot x}) = \text{mso-tp}_k(\mathfrak{M}_v)$. Ainsi, la lecture de v nous conduit bien à l'état $\text{mso-tp}_k(\mathfrak{M}_v)$

Conclusion. D'après le résultat que l'on vient de démontrer, il est clair que $\mathcal{A}$ accepte un mot s si et seulement si $\text{mso-tp}_k(\mathfrak{M}_s)$ est dans F, c'est-à-dire (par définition) si et seulement si $\text{mso-tp}_k(\mathfrak{M}_s)$ est cohérent avec Φ. Mais c'est le cas, par définition de la cohérence, si et seulement si $\mathfrak{M}_s \models \Phi$, donc si et seulement si $s \in L(\Phi)$. Ainsi, $\mathcal{A}$ accepte exactement $L(\Phi)$, donc $L(\Phi)$ est régulier. □

Références

[1] O. Carton. *Langages formels, calculabilité et complexité.* Vuibert, 2008.

[2] L. Libkin. *Elements of Finite Model Theory.* Springer, 2004.

[3] F. Loeser. Un premier cours de logique. `http://www.dma.ens.fr/~loeser/cours_2_12_2009.pdf`, 2009.