Équivalence entre logique monadique du second ordre et automates

Antoine Amarilli

ÉNS

Sommaire

Langages

On a un ensemble de **variables** $x_1, ..., x_n, ...$

Langages

Un langage $\mathcal{L}$ comprend :

- Des **constantes** $c_1, ..., c_n, ...$
- Des **relations** $P_1, ..., P_n, ...$
- La relation "$=$" (égalité).
- Le symbole "T" (tautologie).

Termes, formules atomiques, formules

- Les **termes** sont les variables et les constantes.
- Les **formules atomiques** sont les expressions de la forme $P(t_1, ..., t_k)$ pour $t_1, ..., t_k$ des termes et P une relation d'arité k.
- Les **formules** sont les formules atomiques et les constructions de la forme $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\exists x\, \phi_1$ et $\forall x\, \phi_1$.

Variables libres

- Toutes les variables d'un terme ou d'une formule atomique sont libres.
- Les variables libres d'une formule sont celles qui ne sont pas quantifiées.

Un **énoncé** est une formule sans variables libres.

Structures

Structures

Pour un langage $\mathcal{L}$, $\mathcal{L}$-**structure** comprend :

- Un ensemble A appelé **univers**.
- Une application associant à toute constante du langage un élément de l'univers (l'interprétation de la constante dans la structure).
- Une application associant à toute relation d'arité k une partie de A^k (son interprétation).
- On demande que l'égalité soit interprétée par l'égalité usuelle.

Sous-structures

- On prend un sous-ensemble de l'univers d'une structure de départ où on peut interpréter les constantes.
- On interprète les relations par leur restriction à l'univers de la sous-structure.

Satisfaction d'une formule

Pour $\phi(\underline{x})$ une formule de variables libres $\underline{x} = (x_1, ..., x_n)$, pour $\underline{a} = (a_1, ..., a_n)$ des éléments de l'univers d'une structure $\mathfrak{A}$, on dit que $\underline{a}$ **satisfait** ϕ **dans** $\mathfrak{A}$ (noté $\mathfrak{A} \models \phi(\underline{a})$) si les conditions suivantes sont vérifiées :

- Pour ϕ de la forme $P(t_1, ..., t_k)$, on veut que $(t_1^{\mathfrak{A}}(\underline{a}), ..., t_k^{\mathfrak{A}}(\underline{a})) \in P^{\mathfrak{A}}$.
- Pour ϕ une combinaison booléennes de formules, on exige que les connecteurs booléens soient respectés.
- Pour ϕ de la forme $\exists x\, \phi_1(x, \underline{x})$, on veut qu'il existe un $x \in A$ tel que $\mathfrak{A} \models \phi_1(x, \underline{a})$.
- Pour le quantificateur universel, c'est analogue.

Définissabilité

Association d'un langage à un alphabet

- On prend Σ un alphabet.
- On lui associe le langage $\mathcal{L}_\Sigma$ comprenant le symbole binaire $<$ et des symboles de relation unaire P_a pour tout $a \in \Sigma$.

Association d'une structure à un mot

- On prend $u \in \Sigma^*$ de longueur n.
- On lui associe une $\mathcal{L}_\Sigma$-structure $\mathfrak{M}_u$ d'univers $\{1, ..., n\}$.
- On y interprète $<$ comme l'ordre sur les entiers naturels.
- On y interprète P_a comme l'ensemble des positions où la lettre a apparaît dans u.

Exemple

- On prend $\Sigma = \{a, b, c\}$.
- On prend $u = aba$.
- L'univers de $\mathfrak{M}_u$ est $\{1, 2, 3\}$.
- On a $<^{\mathfrak{M}_u} = \{(1, 2), (1, 3), (2, 3)\}$ (peu surprenant).
- On a aussi (et surtout) $P_a^{\mathfrak{M}_u} = \{1, 3\}$, $P_b^{\mathfrak{M}_u} = \{2\}$ et $P_c^{\mathfrak{M}_u} = \emptyset$.

Association d'un langage formel à un énoncé

- On prend un énoncé Φ sur $\mathcal{L}_\Sigma$.
- On lui associe alors le langage formel $L(\Phi) = \{u \in \Sigma^* \mid \mathfrak{M}_u \models \Phi\}$.
- $L(\Phi)$ est le langage défini par Φ, il est donc **définissable**.

Exemple

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

Exemple

- On prend (encore) $\Sigma = \{a, b, c\}$.
- On va poser :

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

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

Définissabilité des langages sans étoile

Théorème

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

- On ne démontrera ici que le sens le plus facile : les langages sans étoile sont définissables dans la logique du premier ordre.
- On va procéder par induction sur la définition du langage sans étoile.

Rappel : définition des langages sans étoile

Les langages sans étoile forment la plus petite famille de langages comprenant le langage vide et les singletons $\{a\}$ pour $a \in \Sigma$, et close par union, complémentation et produit (mais pas par étoile). Attention, le langage $a\Sigma^* = \{a\} \cdot \overline{\emptyset}$ est sans étoile, malgré les apparences.

Cas de base

- Le langage vide est définissable par $\neg T$.
- Le langage $\{a\}$ est définissable par $(\forall x \forall y\,(x = y)) \wedge (\forall x\, P_a(x))$.

Récurrence (cas faciles)

- Le complémentaire $\overline{L}$ d'un langage L définissable par Φ est définissable par $\neg\Phi$.
- L'union $L_1 + L_2$ de deux langages définissable par Φ_1 et Φ_2 est définissable par $\Phi_1 \vee \Phi_2$.

Récurrence (cas du produit)

- On suppose que L_1 et L_2 sont définissables par des énoncés Φ_1 et Φ_2, et on veut montrer que $L = L_1 \cdot L_2$ est définissable.
- On crée $\phi_1(x)$ en remplaçant dans Φ_1 toutes les quantifications $\exists y\, \phi$ ou $\forall y\, \phi$ par $\exists y\, ((y \leq x) \wedge \phi)$ ou $\forall y\, ((y \leq x) \wedge \phi)$.
- On a alors $\mathfrak{M}_u \models \phi_1(p)$ si et seulement si $\mathfrak{M}_u^{\leq p} \models \Phi_1(p)$, où $\mathfrak{M}_u^{\leq p}$ désigne le sous-structure de $\mathfrak{M}_u$ sur le domaine $\{1, ..., p\}$.
- On construit de même $\phi_2(x)$, mais avec $y > x$, et on a $\mathfrak{M}_u \models \phi_2(p)$ si et seulement si $\mathfrak{M}_u^{>p} \models \Phi_2(p)$.
- Or un mot u appartient à L si et seulement 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 $L(\Phi) = L$.

- On peut également montrer que les langages sans étoile sont précisément ce que permet de définir la logique du premier ordre.
- Elle ne suffit donc pas à définir tous les langages rationnels, puisque l'on sait 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.

Sommaire

Définitions

Idée

La logique monadique 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.

Langages

C'est comme en logique du premier ordre. Mais on suppose que pour tout $k > 0$, on dispose d'un ensemble de variables du second ordre $X_1, ..., X_n, ...$ représentant des relations unaires.

Remarque

En logique du second ordre "standard", on pourrait avoir des variables du second ordre de n'importe quelle arité (pas seulement 1).

Formules

- Les **termes** sont les mêmes qu'en logique du premier ordre.
- Les **formules atomiques** sont les mêmes, plus les expressions de la forme $X_i(t)$ où t est un terme et X_i une variable du second ordre.
- Les **formules** sont les formules atomiques et les constructions de la forme $\phi_1 \wedge \phi_2$, $\phi_1 \vee \phi_2$, $\neg\phi_1$, $\exists x\, \phi_1$, $\forall x\, \phi_1$, $\exists X\, \phi_1$, et $\forall X\, \phi_1$

Variables libres

C'est comme en logique du premier ordre, mais on peut avoir des variables libres du second ordre (définies suivant les mêmes règles).

Structures

Les notions de structure et d'interprétation sont comme en logique du premier ordre.

Satisfaction d'une formule

Pour $\phi(\underline{x}, \underline{X})$ une formule du second ordre de variables libres $\underline{x} = (x_1, ..., x_n)$ et $\underline{X} = (X_1, ..., X_m)$, pour $\underline{b} = (b_1, ..., b_n) \in A^n$, pour $\underline{B} = (B_1, ..., B_m)$ avec $B_i \subset A$ pour tout i (où $\mathfrak{A}$ est une $\mathcal{L}$-structure), on dit que $(\underline{a}, \underline{B})$ **satisfait** ϕ **dans** $\mathfrak{A}$ (noté $\mathfrak{A} \models \phi(\underline{a}, \underline{B})$) dans les cas suivants :

- Si $\phi(\underline{x}, X)$ est de la forme $X(t)$ où t est un terme et X une variable du second ordre, on a $\mathfrak{A} \models \phi(\underline{a}, B)$ si et seulement si $t^{\mathfrak{A}} \in B$.
- Si $\phi(\underline{x}, \underline{X})$ est de la forme $\exists Y \, \psi(\underline{x}, Y, \underline{X})$ où Y est une variable du second ordre on a $\mathfrak{A} \models \phi(\underline{a}, \underline{B})$ si et seulement s'il existe un $C \subset A$ tel que $\mathfrak{A} \models \psi(\underline{a}, C, \underline{B})$.
- Dans le cas du quantificateur universel, c'est analogue.
- Dans les autres cas, c'est comme en logique du premier ordre.

Énoncé du théorème

Théorème (Büchi)

Un langage est définissable dans la logique monadique du second ordre si et seulement s'il est régulier.

Remarque

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.

- On va d'abord montrer le sens facile : les langages réguliers sont définissables.
- Pour le sens réciproque, plus difficile, on aura besoin de définir les types.

Définissabilité des langages réguliers

Notations

- On note Σ l'alphabet.
- On pose L un langage régulier.
- On va construire un énoncé Φ de $\mathcal{L}_\Sigma$ qui définit L.
- On pose $\mathcal{A} = (Q, \delta, q_0, F)$ un automate déterministe reconnaissant L ($Q = q_0, ..., q_{m-1}$ l'ensemble des états, δ la fonction de transition, q_0 l'état initial, F les états finaux).

Abréviations

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

L'idée

Le principe

- On va 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.
- Un mot est reconnu par l'automate si et seulement s'il existe des valeurs de $X_0, ...X_{m-1}$ représentant une exécution valide de l'automate sur son entrée.

Les contraintes à imposer

- Les X_i forment une partition des positions (ie. dans chaque position, on est à un et un seul état).
- L'exécution commence en q_0.
- Le passage d'un état à un autre respecte la fonction de transition
- L'exécution se termine à un état final

Construction détaillée (vue d'ensemble)

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

$$\phi \equiv \phi_{\text{partition}} \wedge \phi_{\text{initial}} \wedge \phi_{\text{transition}} \wedge \phi_{\text{accepte}}$$

$$\phi_{\text{partition}} \equiv \forall x \left(\left(\bigvee_{i=0}^{m-1} X_i(x) \right) \wedge \neg \left(\bigvee_{0 \leq i < j \leq m-1} X_i(x) \wedge X_j(x) \right) \right)$$

$$\phi_{\text{initial}} \equiv \forall x \left((\forall y\, (x \leq y)) \Rightarrow \bigwedge_{a \in \Sigma} \left(P_a(x) \Rightarrow X_{\hat{\delta}(q_0, a)}(x) \right) \right)$$

$$\phi_{\text{transition}} \equiv \forall x \forall y \left((x \prec y) \Rightarrow \bigwedge_{i=0}^{m-1} \bigwedge_{a \in \Sigma} \left(X_i(x) \wedge P_a(y) \Rightarrow X_{\delta(q_i, a)}(y) \right) \right)$$

$$\phi_{\text{accepte}} \equiv \forall x \left((\forall y\, (y \leq x)) \Rightarrow \bigvee_{q_i \in F} X_i(x) \right)$$

Explications

Formule générale Φ

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

La formule Φ impose qu'il existe une exécution de l'automate respectant les conditions imposées par ϕ.

Formule ϕ

$$\phi \equiv \phi_{\text{partition}} \wedge \phi_{\text{initial}} \wedge \phi_{\text{transition}} \wedge \phi_{\text{accepte}}$$

La formule ϕ impose les quatre conditions $\phi_{\text{partition}}$, ϕ_{initial}, $\phi_{\text{transition}}$ et ϕ_{accepte}.

Formule $\phi_{\text{partition}}$

$$\phi_{\text{partition}} \equiv \forall x \left(\left(\bigvee_{i=0}^{m-1} X_i(x) \right) \wedge \neg \left(\bigvee_{0 \leq i < j \leq m-1} X_i(x) \wedge X_j(x) \right) \right)$$

« pour toute position, l'automate après la lecture de la lettre à cette position se trouve en un état et un seul »

Formule ϕ_{initial}

$$\phi_{\text{initial}} \equiv \forall x \left((\forall y \, (x \leq y)) \Rightarrow \bigwedge_{a \in \Sigma} \left(P_a(x) \Rightarrow X_{\hat{\delta}(q_0, a)}(x) \right) \right)$$

« après avoir lu la première lettre, l'automate se trouve dans l'état correspondant à la lecture de cette lettre depuis l'état initial »

Formule $\phi_{\text{transition}}$

$$\phi_{\text{transition}} \equiv \forall x \forall y \left((x \prec y) \Rightarrow \bigwedge_{i=0}^{m-1} \bigwedge_{a \in \Sigma} \left(X_i(x) \wedge P_a(y) \Rightarrow X_{\delta(q_i,a)}(y) \right) \right)$$

« 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 $\delta(q_i, a)$ »

Formule ϕ_{accepte}

$$\phi_{\text{accepte}} \equiv \forall x \left((\forall y \, (y \leq x)) \Rightarrow \bigvee_{q_i \in F} X_i(x) \right)$$

« pour toute position, si cette position est la dernière, alors l'automate se trouve dans un des états finaux »

Sommaire

Définitions préliminaires

Profondeur maximale de quantificateurs

On définit inductivement la **profondeur maximale de quantificateurs** d'une formule 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.
- $\mathrm{qr}(\exists x\, \phi) = \mathrm{qr}(\forall x\, \phi) = \mathrm{qr}(\exists X\, \phi) = \mathrm{qr}(\forall X\, \phi) = \mathrm{qr}(\phi) + 1$ pour une formule ϕ et une variable x.

On note $\mathrm{MSO}[k]$ l'ensemble des formules de la logique monadique du second ordre dont la profondeur maximale de quantificateurs est inférieure ou égale à k.

Types

- On fixe un langage $\mathcal{L}$.
- On pose $\mathfrak{A}$ 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 $\{\phi \in \mathrm{MSO}[k] \mid \mathfrak{A} \models \phi\}$.
- 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 $\phi \in \mathrm{MSO}[k]$, $\mathfrak{A}_1 \models \phi$ si et seulement si $\mathfrak{A}_2 \models \phi$.

Nombre fini de types

Théorème

Pour un langage $\mathcal{L}$ fini fixé, pour tout k, pour tous m et m', $\mathrm{MSO}[k]$ ne contient qu'un nombre fini de formules ayant $\underline{x} = x_1, ..., x_m$ et $\underline{X} = X_1, ..., X_{m'}$ pour variables libres, à équivalence logique près.

On va procéder par récurrence sur k.

Cas de base

- On fixe m et m'.
- Les éléments de $\mathrm{MSO}[0]$ sont les combinaisons booléennes de formules atomiques.
- Le nombre de variables, de constantes et de relations possibles est fini ($m + m'$ variables libres).
- Il y en a un nombre fini de combinaisons booléennes non équivalentes.

Récurrence

- On suppose que $\mathrm{MSO}[k]$ est fini pour tous m, m'. On fixe m et m'.
- On a les formules de $\mathrm{MSO}[k]$, en nombre fini.
- On a les formules du type $Qx_{m+1}\,\phi(\underline{x}, x_{m+1}, \underline{X})$ pour $\phi \in \mathrm{MSO}[k]$ à $m+1$ variables libres, en nombre fini aussi. Même chose pour le second ordre.
- On a les combinaisons booléennes de tout cela, en nombre fini à équivalence près, comme précédemment.

Corollaire

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

Démonstration

Il suffit de prendre des représentants ϕ_i des énoncés de $\mathrm{MSO}[k]$.

Corollaires supplémentaires

Corollaire

On peut caractériser un type τ de rang k par une formule ψ_τ qui indique lesquels des ϕ_i sont satisfaits et lesquels ne le sont pas. Formellement, si $K_\tau = \{\phi_i \in \mathrm{MSO}[k] \mid \phi_i \in \tau\}$:

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

Corollaire

Réciproquement, toute formule ϕ de $\mathrm{MSO}[k]$ est équivalente à une disjonction de formules de la forme ψ_τ (celles où elle apparaît).

Type cohérents

L'ensemble des **types cohérents avec** $\phi \in \mathrm{MSO}[k]$ est l'ensemble des types τ tels que $\phi \in \tau$, ie. les types τ_i tels que $\phi = \bigvee_i \psi_{\tau_i}$.

Caractère régulier des langages définissables

Notations

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

Construction de l'automate

On pose $\mathcal{A} = (\{\tau_0, ..., \tau_n\}, E, \tau_0, F)$ avec $\{\tau_0, ..., \tau_n\}$ l'ensemble des états, τ_0 l'état initial, F les états finaux, et E l'ensemble des transitions. On prend :

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

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

Déterminisme

- Si on a $\mathrm{tp}_k(\mathfrak{M}_{s_1}) = \mathrm{tp}_k(\mathfrak{M}_{s_2}) = \tau_i$, on a $\mathfrak{M}_{s_1} \equiv_k \mathfrak{M}_{s_2}$.
- Un lemme (admis) nous permet alors de montrer que $\mathrm{tp}_k(\mathfrak{M}_{s_1 \cdot a}) = \mathrm{tp}_k(\mathfrak{M}_{s_2 \cdot a}) = \tau_{j_2}$.

Complétude

- Démontrons par récurrence sur la longueur du mot que la lecture d'un mot u nous conduit à l'état $\mathrm{tp}_k(\mathfrak{M}_u)$.
- La lecture de ϵ nous conduit à l'état $\tau_0 = \mathrm{tp}_k(\mathfrak{M}_\epsilon)$.
- Si la lecture d'un mot u de longueur n nous conduise à l'état $\mathrm{tp}_k(\mathfrak{M}_u)$, il est immédiat d'après la définition que la lecture de $v = ux$ nous conduit à l'état $\mathrm{tp}_k(\mathfrak{M}_v)$ (il existe une transition).

Conclusion

- $\mathcal{A}$ accepte un mot s si et seulement si $\mathrm{tp}_k(\mathfrak{M}_s)$ est dans F
- Donc, l'automate accepte s si et seulement si $\mathrm{tp}_k(\mathfrak{M}_s)$ est cohérent avec Φ.
- Mais, par définition de la cohérence, c'est vrai 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.