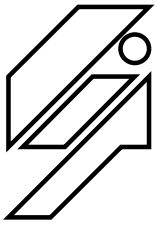
Model Theory and Computational Complexity

Grégory Lafitte, Jacques Mazoyer



Laboratoire de l'Informatique du Parallélisme

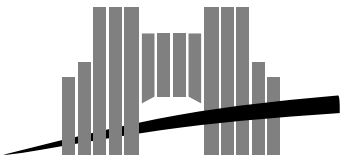
Ecole Normale Supérieure de Lyon
Unité de recherche associée au CNRS n°1398

Théorie des modèles et Complexité

Grégory Lafitte
Jacques Mazoyer

Septembre 1997

Research Report N^{O} 98-02



Ecole Normale Supérieure de Lyon
46 Allée d'Italie, 69364 Lyon Cedex 07, France
Téléphone : (+33) (0)4.72.72.80.00 Télécopieur : (+33) (0)4.72.72.80.80
Adresse électronique : lip@lip.ens−lyon.fr

Théorie des modèles et Complexité

Grégory Lafitte
Jacques Mazoyer

Septembre 1997

Abstract

Model theory has lately become a domain of interest to computer scientists. The reason is that model theory, and in particular its restriction to finite models, has led to some new results in computational complexity (e.g. NLOGSPACE = co-NLOGSPACE).
In this report, firstly we present a survey of this theory and we focus on the descriptive complexity aspects and other links between computational complexity and model theory.
Secondly, we extend some results of Grandjean and Lynch. We give a more precise logical characterization of complexity classes NTIME(n^d) for some d.
This leads us to show applications of this result and to give openings made possible by this result.

Keywords: Model theory, computational complexity, games on structures.

Résumé

Les informaticiens théoriques se sont récemment intéressés à la théorie des modèles. La raison est que la théorie des modèles, en particulier lorsque l'on se restreint à des modèles finis, a permis d'aboutir à de nouveaux résultats en complexité (e.g. NLOGSPACE = co-NLOGSPACE).
Dans ce rapport, nous présentons premièrement une étude de cette théorie dans le but de présenter les notions de complexité descriptive et d'autres liens entre la complexité et la théorie des modèles.
Deuxièmement, nous étendons des résultats de Grandjean et Lynch. Nous donnons une caractérisation logique plus précise des classes de complexité NTIME(n^d) pour un certain d.
Enfin, nous montrons des implications de notre résultat et nous donnons différentes ouvertures rendues possibles grâce à ce résultat.

Mots-clés: Théorie des modèles, complexité de calcul, jeux sur des structures.

1. Introduction

La complexité de calcul d'un problème est la quantité de ressources, telles que le temps et l'espace, requise pour une machine[1] qui résout le problème. La théorie de la complexité s'est concentrée principalement sur la complexité de calcul pour certains problèmes. Une branche plus récente de la théorie de la complexité se préoccupe de la complexité descriptive de problèmes, c'est-à-dire la complexité de la description de problèmes dans un certain formalisme logique [22]. Un des développements palpitants récents de la théorie de la complexité a été la découverte d'un lien étroit entre complexité descriptive et complexité de calcul.

Ce lien étroit a été en premier mis en avant par Fagin, qui a montré [5] que la classe de complexité NP coïncide avec la classe des propriétés des structures finies définissables dans la logique existentielle du second ordre, appelée Σ^1_1. Stockmeyer a ensuite montré que ce résultat pouvait être étendu pour donner une correspondance fine entre la hiérarchie polynomiale et la logique du second ordre [35].

Le pas suivant fut franchi par Immerman et Vardi, qui ont prouvé que la classe de complexité P coïncide avec la classe des propriétés des structures finies ordonnées définissables dans une logique *point-fixe* [19, 37]. Ce lien fort entre la complexité de calcul et la complexité descriptive fut décrit par Immerman [20] et a été depuis étudié par beaucoup de chercheurs [3, 8, 9, 10, 13, 14, 15, 17, 22, 26, 27, 28, 33, 34, 36]. Ce lien est considéré comme un des aspects les plus importants de la théorie des modèles finis (voir [6]). Pour une étude détaillée de ce lien, on invite le lecteur à se reporter au livre d'Ebbinghaus et Flum [4] et à l'article d'Immerman [22].

L'étude de ce que l'on appelle aujourd'hui la "théorie des modèles finis" a été largement motivée par la théorie de la complexité mais cette théorie a néanmoins ses racines en théorie des modèles classique. La théorie des modèles, comme elle fut appelée par Tarski en 1954, peut être considérée comme la partie de la sémantique des langages formels qui s'intéresse aux interactions entre la structure syntaxique d'un système axiomatique et les propriétés de ses modèles. Il se trouve que la logique du premier ordre est devenue le langage proéminant. La raison est qu'elle obéit à certains principes fondamentaux comme le théorème de complétude et *a fortiori* le théorème de compacité. Ce sont à la fois de vrais outils et des témoins de la faible puissance d'expression de la logique du premier ordre. C'est cette faiblesse qui permet que l'on ait des outils aussi puissants et qui justifie alors que la logique du premier ordre soit le fondement de la théorie des modèles.

1991 *Mathematics Subject Classification.* Primary: 03C13, Secondary: 68Q15.

1. bien évidemment, dans un certain modèle de calcul avec un certain type de donnée.

Par le théorème de compacité, tout système d'axiomes du premier ordre a soit des modèles finis de cardinalité bornée, soit des modèles infinis. Le premier cas étant trivial, la théorie des modèles considère habituellement *tous* les modèles d'un système axiomatique, et a donc des modèles infinis. Tout se joue dans le monde infini. Toutes les méthodes de la théorie des modèles, telles les méthodes pour la construction de modèles (ultra-produits ...), se préoccupent de structures infinies et se rapprochent fortement de la théorie des ensembles.

Il y a cependant de bonnes raisons de considérer des structures finies. Historiquement, la plus importante fut la reformulation du théorème de compacité qui dit que pour certaines classes de formules du premier ordre, il y a équivalence entre la satisfiabilité et la satisfiabilité dans le cas fini. C'est ce qui a permis de résoudre le problème de décision pour des classes préfixes de la logique du premier ordre. Il a tout de même fallu attendre une vingtaine d'années avant de se poser des questions du type de la théorie des modèles en se restreignant à des structures finies : on a alors le théorème de Trahtenbrot et la reformulation du problème du spectre par Scholz. C'est là que sont apparus les aspects probants de la calculabilité.

L'étude de la théorie des modèles finis a déjà permis de résoudre une question bien connue de complexité : Immerman [21] a montré que NL=coNL en s'inspirant d'observations concernant la définissabilité de P et d'un résultat plus faible dans [25]. Un autre vieux problème de l'informatique théorique est de trouver des bornes inférieures pour la complexité de problèmes naturels spécifiques. Bien qu'il existe beaucoup de théorèmes qui montrent l'existence de hiérarchies, ils disent rarement quoi que ce soit sur la complexité d'un problème donné. Ceci est particulièrement vrai dans la classe de complexité NP. Il y a des centaines de problèmes NP-complets connus, mais jusqu'à récemment, on ne connaissait pour aucun d'eux une borne inférieure non triviale de complexité. Grandjean a montré [11] que le problème NP-complet *Réduction d'automates incomplètement spécifiés* (noté AL7 dans la classification de Garey et Johnson [7]) n'est pas résolvable en temps déterministe linéaire. Sa méthode réside en une réduction, en temps linéaire, de tout $L \in \mathrm{NTIME}(n)$ [2] vers AL7. Ceci implique que si AL7$\in \mathrm{DTIME}(n)$, alors $\mathrm{NTIME}(n) = \mathrm{DTIME}(n)$, ce qui est bien entendu faux [31].

Une étape intermédiaire dans la réduction de Grandjean est la construction d'un énoncé dont les modèles finis représentent les entrées acceptées après le calcul d'une machine de Turing non-déterministe qui reconnaît L. La réduction est possible en temps linéaire parce que la longueur du codage de chaque modèle est linéaire en le nombre d'étapes dans le calcul correspondant. La première utilisation d'un codage linéaire des évolutions d'une machine de Turing en un modèle fini se trouve dans [29].

---

2. on utilise la notation habituelle pour les classes de complexité : $\mathrm{NTIME}(n^k)$ désigne les problèmes qui sont résolvables en temps $\mathrm{O}(n^k)$ sur une machine de Turing non-déterministe; $\mathrm{DTIME}(n)$ est la classe similaire en se restreignant à des machines de Turing déterministes; NTREAL désigne la classe des problèmes résolvables en temps exactement égal à la longueur de l'entrée et NTLIN=$\mathrm{NTIME}(n)$; enfin NLIN désigne la classe des problèmes résolvables en temps linéaire sur une machine RAM non déterministe, étudiée et introduite par Grandjean[10].

On utilisera un codage complètement similaire pour montrer notre résultat principal, qui généralise ce résultat de Grandjean et les résultats ultérieurs de Grandjean et Lynch sur ce sujet.

Le but de ce rapport est de montrer qu'il existe une caractérisation de la classe NTLIN des problèmes qui sont *résolvables* par une machine de Turing non-déterministe en temps linéaire. En fait nous caractérisons la classe NTIME(n^k) pour chaque k.

Dans leurs travaux jusqu'à maintenant, Lynch et Grandjean se sont surtout concentrés sur la comparaison de la classe NTIME(n) avec quelques classes de formules logiques $\mathcal{F}$ (donc un résultat dans un seul sens, et non pas une caractérisation). Lynch espère ainsi obtenir des résultats comme "$\Pi \notin$ NTIME(n)", pour quelques problèmes NP-complèts Π, en montrant leur non-définissabilité à l'aide des formules de $\mathcal{F}$.

Donc, en fait, pour obtenir leur objectif, "NTIME(n) $\subseteq \mathcal{F}$" suffit. On se place dans cette même perspective.

Déjà, Grandjean (en [9]) a montré que pour chaque $\mathcal{L} \in$ NTIME(n^d), il existe un énoncé fonctionnel σ de classe $\Sigma^1_1\Pi^0_1$ et de degré (arité maximale) d avec au plus d variables de premier ordre, qui définit $\mathcal{L}$.

Ensuite, en partant de ce point, Lynch (en [30]) élimine la partie fonctionnelle du théorème de Grandjean en prouvant que pour chaque $\mathcal{L} \in$ NTIME(n^d), il existe un énoncé σ de classe $\Sigma^1_1\Pi^0_2$ et de degré d (remarquons le fait que d est le degré des prédicats et non plus des fonctions) qui a comme relation *interne* PLUS en plus de l'ordre (l'*addition* est déjà dans les structures considérées), et qui définit $\mathcal{L}$.

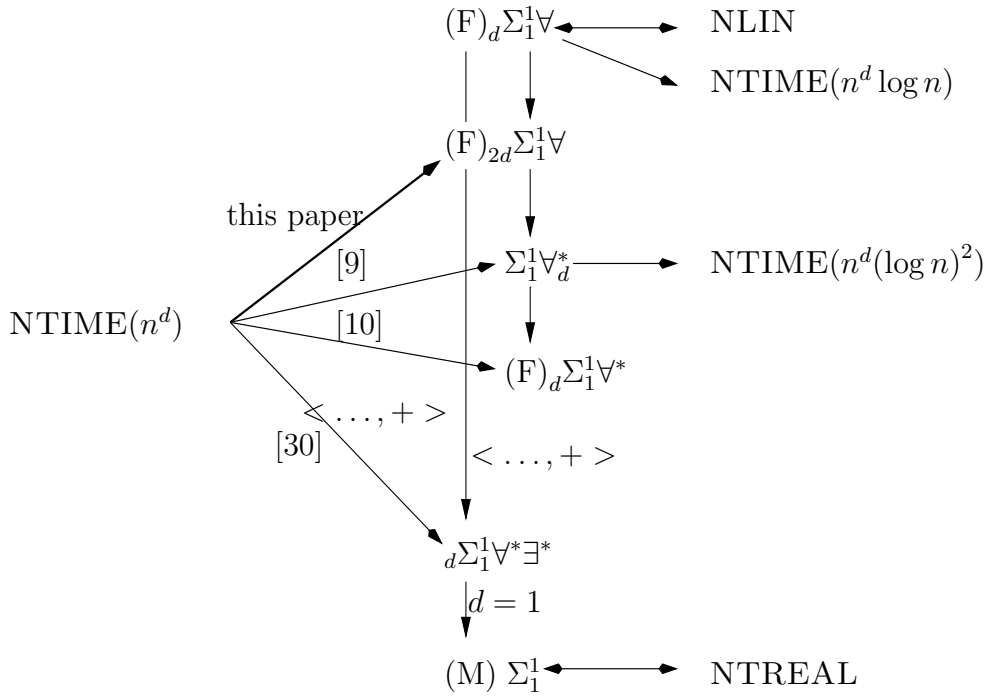
Comme le dit Lynch dans [30], "si on essaye de démontrer un résultat de non-définissabilité, il est intéressant de restreindre le plus possible la forme de l'énoncé, espérant que ceci va faciliter la preuve". En particulier, ceci motive l'intérêt de Lynch et Grandjean dans le préfixe quantificateur $\forall^*\exists^*$. Lynch écrit aussi dans le même article "Une question immédiate est : existe-t-il une forme, pour le préfixe, qui soit encore plus simple et qui suffise? La réponse ne nous est pas connue.". Ces deux remarques motivent notre intérêt concernant le préfixe quantificateur du premier ordre (seulement un $\forall$) ; aussi avons-nous comme but ultime l'obtention d'un résultat dans les deux sens (une capture exacte de la classe de complexité). Par conséquent, la restriction de la forme de notre proposition ne peut que nous faire avancer dans cette direction.

Notre résultat principal est que tout problème reconnu par une machine de Turing non-déterministe en temps O(n^k) est exprimable (la classe de structures correspondant au langage du problème est axiomatisable) par un énoncé de la logique [3] $(\mathrm{F})_{2d}\Sigma^1_1\forall$.

Le diagramme commutatif qui suit montre comment notre résultat s'insère dans le cadre des travaux déjà effectués autour de la caractérisation logique du temps polynomial non-déterministe.

---

3. pour toutes ces notations de classes (fragments) de formules de la logique du second ordre, voir la définition 1.



Les résultats (flèches minces partant de NTIME(n^d)) décrits brièvement dans ce diagramme appartiennent à Lynch et Grandjean, tandis que Grandjean (dans [12]), parmi d'autre résultats, présente l'implication entre (MF)$\Sigma_1^1\forall$ et (M)$\Sigma_1^1\forall^*\exists^*$ (qui se généralise en une implication entre (F)$_d\Sigma_1^1\forall$ et $_d\Sigma_1^1\forall^*\exists^*$), pourvu que l'*addition* soit dans la structure. La flèche en gras représente le résultat principal exposé dans ce rapport. Les flèches plus simples indiquent des implications de la théorie des modèles. Certaines de ces implications sont vraies uniquement dans certaines structures, comme indiqué sur le diagramme. On voit clairement à travers ce diagramme que notre résultat étend les résultats de Grandjean et Lynch.

Le rapport s'organise de la façon suivante : tout d'abord on introduit rapidement le lecteur à la théorie des modèles et ses méthodes, en essayant d'expliciter là où les problèmes de complexité sont transposés ; on présente ensuite l'approche *descriptive* de la complexité ; on arrive alors à notre résultat et on termine en montrant les implications de ce résultat et les différentes ouvertures possibles.

Ce rapport est le mémoire du stage de DEA effectué au LIP à l'Ecole Normale Supérieure de Lyon sous la direction de Jacques Mazoyer. Mon travail a consisté à faire une étude bibliographique profonde sur la théorie des modèles et ses dernières applications en complexité, à étudier les nombreux travaux sur la caractérisation logique de classes de complexité spécifiques, à étendre de façon significative ces caractérisations et à en tirer les applications ou ouvertures possibles. Cette étude a été menée dans l'espoir d'une part de trouver de nouveaux résultats en complexité, et d'autre part d'arriver à une caractérisation logique des classes de complexité sur les automates cellulaires. Nous nous sommes apperçus qu'il restait beaucoup de

questions à étudier au sujet de la classe des problèmes reconnus en temps linéaire par une machine de Turing. Les problèmes de complexité des automates cellulaires s'articulant autour de la comparaison de problèmes reconnus en temps linéaire et en temps réel (en la taille de l'entrée), il nous a semblé judicieux de commencer par étudier le cas linéaire pour les machines de Turing.

Je tiens à remercier Jacques Mazoyer pour son temps, nos discussions et tout particulièrement son enthousiasme. Mes remerciements vont également à mon co-directeur de thèse, Menachem Magidor, pour le vif intérêt qu'il m'a transmis pour la théorie des modèles et la théorie des ensembles. Je remercie aussi les différentes personnes du Logic Colloquium '97 pour nos discussions fructueuses, ainsi que David Coudert, André Elisseeff, Codrin Nichitiu et Rivo Randrianarivoni pour leur patience et leurs encouragements.

2. Rappels de théorie des modèles

On commence par une introduction rapide mais suffisante des éléments de la théorie des modèles dont nous avons besoin. Pour une étude plus détaillée, le lecteur est invité à se référer aux livres de Chang et Keisler [2], Hodges [18], Jech [23] et Poizat [32].

Un *langage* (ou un *vocabulaire*) est un ensemble de symboles : des symboles pour des relations, des fonctions et des constantes (des relations 0-aires).

Un *modèle* (on parle également de *structure*) pour un langage donné $\mathcal{L}$ est un couple $\mathcal{A} = (A, \mathcal{I})$, où A est l'univers de $\mathcal{A}$ et $\mathcal{I}$ la fonction d'*interprétation* qui assigne les relations, fonctions et constantes appropriées de A aux symboles de $\mathcal{L}$. Un modèle pour $\mathcal{L}$ est habituellement noté de la façon suivante :

$$\mathcal{A} = \langle A, P^{\mathcal{A}}, \ldots, F^{\mathcal{A}}, \ldots, c^{\mathcal{A}}, \ldots \rangle$$

Les exposants $\mathcal{A}$ précisent qu'il s'agit des interprétations de ces symboles dans le modèle $\mathcal{A}$; on ne s'encombrera pas de $\mathcal{A}$ lorsque ce sera clair de quel modèle il s'agit.

Par récurrence sur la longueur des termes et des formules, on définit, de façon naturelle (voir plus loin), la *valeur* d'un terme

$$t^{\mathcal{A}}[a_1, \ldots, a_n]$$

et la *satisfiabilité*

$$\mathcal{A} \models \varphi[a_1, \ldots, a_n]$$

Deux modèles $\mathcal{A} = \langle A, P, \ldots, F, \ldots, c, \ldots \rangle$ et $\mathcal{A}' = \langle A', P', \ldots, F', \ldots, c', \ldots \rangle$ sont *isomorphes* s'il existe un *isomorphisme* entre $\mathcal{A}$ et $\mathcal{A}'$, c'est-à-dire une bijection f de A sur A' telle que :

(1) $P(x_1, \ldots, x_n)$ si et seulement si $P'(f(x_1), \ldots, f(x_n))$,
(2) $f(F(x_1, \ldots, x_n)) = F'(f(x_1), \ldots, f(x_n))$,
(3) $f(c) = c'$,

pour toutes relations, fonctions et constantes de $\mathcal{A}$.

Un *sous-modèle* de $\mathcal{A}$ est un sous-ensemble $B \subseteq A$ muni des relations $P^{\mathcal{A}} \cap B^n, \ldots$, des fonctions $F^{\mathcal{A}} \mid B^m, \ldots$, et des constantes $c^{\mathcal{A}}, \ldots$; B doit être tel que toute constante $c^{\mathcal{A}}$ appartienne à B, et que B soit clos par toute fonction $F^{\mathcal{A}}$.

Un sous-modèle $\mathcal{B} \subseteq \mathcal{A}$ est un sous-modèle *élémentaire*

$$\mathcal{B} \prec \mathcal{A}$$

si pour toute formule φ, et tous $a_1, \ldots, a_n \in B$,

$$\mathcal{B} \models \varphi[a_1, \ldots, a_n] \text{ si et seulement si } \mathcal{A} \models \varphi[a_1, \ldots, a_n]$$

On montre facilement que cette notion de sous-modèle élémentaire diffère de la notion d'isomorphisme.

Le lemme clef pour la construction de sous-modèles élémentaires est: un sous-ensemble $B \subseteq A$ forme un sous-modèle élémentaire de $\mathcal{A}$ si et seulement si pour toute formule φ, et tout $a_1, \ldots, a_n \in B$,

$$\text{si } \exists a \in A \text{ tel que } \mathcal{A} \models \varphi[a, a_1, \ldots, a_n],$$

$$\text{alors } \exists a \in B \text{ tel que } \mathcal{A} \models \varphi[a, a_1, \ldots, a_n]$$

Une fonction $h : A^n \to A$ est une *fonction de Skolem* pour φ si

$$\exists a \in A, \mathcal{A} \models \varphi[a, a_1, \ldots, a_n] \Rightarrow \mathcal{A} \models \varphi[h(a_1, \ldots, a_n), a_1, \ldots, a_n]$$

pour tout $a_1, \ldots, a_n$.

Grâce à ces fonctions de Skolem (et à la skolemisation), on peut se restreindre, dans notre étude de la théorie des modèles, aux structures sans fonction. C'est ce que nous allons faire à partir de maintenant. Le prix pour se débarasser des fonctions est l'introduction, pour chaque fonction n-aire f , d'une nouvelle relation $n+1$-aire F qui est le graphe de la fonction.

Après ces brefs rappels, nous allons définir les différentes logiques (syntaxe et sémantique) utilisées en complexité descriptive et ailleurs. On a d'abord, bien entendu, ce que l'on appelle la logique du premier ordre FO qui est à la base de toutes les autres logiques. Nous rappelons brièvement sa définition ; tout d'abord, l'aspect syntaxique de FO. On fixe un vocabulaire τ. Chaque formule de la logique du premier-ordre est un mot à partir de l'alphabet $\{v_1, v_2, v_3, \ldots, \neg, \vee, \exists, =,), (\} \cup \{$ symboles de τ $\}$ (où les v_i sont les variables). Un terme de vocabulaire τ est une variable ou une constante de τ. Une formule de la logique du premier ordre de vocabulaire τ sont les mots qui s'obtiennent en appliquant un nombre fini de fois les règles suivantes :

(1) Si t et u sont des termes, alors $t = u$ est une formule;
(2) Si R est dans τ et $t_1, \ldots, t_n$ sont des termes, alors $Rt_1 \ldots t_n$ est une formule;
(3) Si φ est une formule, alors $\neg\varphi$ est une formule;
(4) Si φ et ψ sont des formules, alors $(\varphi \vee \psi)$ est une formule;
(5) Si φ est une formule et x une variable, alors $\exists x\varphi$ est une formule.

On note alors FO[τ], l'ensemble des formules de la logique du premier ordre de vocabulaire τ. Les formules obtenues par les deux premières règles sont appelées formules *atomiques*. On appelle *énoncé* une formule dans laquelle chaque occurence de chaque variable est liée (on définit de manière classique la notion d'occurence libre ou liée d'une variable).

Pour l'instant, nous avons seulement décrit la syntaxe de FO. On va maintenant donner un sens (*sémantique*) à ces symboles logiques. Soit $\mathcal{A}$ une τ-structure. Une assignation dans $\mathcal{A}$ est une fonction α de domaine l'ensemble des variables et à valeurs dans A. On étend facilement α à une fonction définie sur l'ensemble des termes. La relation $\models$ de satisfiabilité (dans $\mathcal{A}$ selon un assignation α) est alors définie comme suit :

$\mathcal{A} \models t_1 = t_2[\alpha]$	si et seulement si	$\alpha(t_1) = \alpha(t_2)$
$\mathcal{A} \models Rt_1 \dots t_n[\alpha]$	si et seulement si	$R^{\mathcal{A}}\alpha(t_1) \dots \alpha(t_n)$
$\mathcal{A} \models \neg\varphi[\alpha]$	si et seulement si	$\mathcal{A} \not\models \varphi[\alpha]$
$\mathcal{A} \models (\varphi \vee \psi)[\alpha]$	si et seulement si	$\mathcal{A} \models \varphi[\alpha]$ ou $\mathcal{A} \models \psi[\alpha]$
$\mathcal{A} \models \exists x\varphi[\alpha]$	si et seulement	s'il existe $a \in A$ tel que $\mathcal{A} \models \varphi[\alpha\,{}^{a}/_{x}]$

où $\alpha\,{}^{a}/_{x}$ est égale à α pour toutes les variables sauf pour x, pour lequel elle est égale à a. Bien évidemment, pour les énoncés, cette notion est indépendante de α.

On introduit maintenant la logique du second-ordre. La logique du *second ordre* (SO) est une extension de la logique du premier ordre dans laquelle on permet de *quantifier* sur des relations. Donc, en plus des symboles de la logique du premier ordre, l'alphabet contient des relations variables n-aires (pour chaque n) $V_1^n, V_2^n, \dots$ (en quantité dénombrable). Pour définir les formules de la logique du second ordre de vocabulaire τ, on introduit, en plus des règles pour les formules du premier ordre, les règles suivantes :

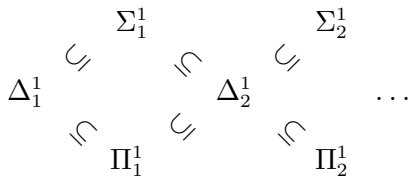
(1) Si X est n-aire et $t_1, \dots, t_n$ sont des termes, alors $Xt_1 \dots t_n$ est une formule;
(2) Si φ est une formule et X est une relation variable, alors $\exists X\varphi$ est une formule.

On définit de façon similaire (à la logique du premier ordre) les occurrences libres et liées d'une variable et la notion de satisfiabilité est étendue de façon naturelle.

On parlera également de la logique du second ordre *monadique* (MSO); c'est la logique du second ordre où les formules ne peuvent avoir que des relations variables unaires.

On montre facilement que toute formule du second-ordre est équivalente à une formule du second ordre en forme prénexe normale dans laquelle les quantificateurs du second ordre précèdent ceux du premier ordre (c'est-à-dire de la forme $Q_1X_1 \dots Q_sX_sq_1x_1 \dots q_px_p\varphi$ où $Q_i, q_i \in \{\exists, \forall\}$, où les X_i et x_i sont respectivement des variables du second et premier ordre et φ n'a pas de quantificateurs). Si la suite de ces quantificateurs est constituée de n blocs consécutifs (tels que dans chaque bloc, on n'ait qu'une sorte de quantificateurs, universels ou existentiels) et le premier bloc est existentiel, alors on dit que c'est une formule Σ_n^1; sinon, si le premier bloc est universel, alors c'est une formule Π_n^1. C'est ce qui est communément appelé la hiérarchie de Lévy. La négation d'une formule Σ_n^1 est clairement une formule Π_n^1

et vice versa. On dira qu'une formule φ est Δ^1_n si elle est logiquement équivalente[4] à une formule Σ^1_n et à une formule Π^1_n.



On montre que ces inclusions sont strictes pour des modèles arbitraires (pas restreints à des modèles finis). La question de savoir si ces inclusions sont également strictes pour des structures finies est fortement reliée à des questions de complexité comme nous le verrons plus tard. Tout cela reste vrai pour le cas monadique.

On va maintenant définir les logiques infinitaires qui sont très utiles en théorie des modèles finis. Les logiques infinitaires $\mathrm{L}_{\infty\omega}$ et $\mathrm{L}_{\omega_1\omega}$ permettent respectivement des disjonctions arbitraires et dénombrables. La classe des formules de la logique $\mathrm{L}_{\infty\omega}$ sur un vocabulaire τ est définie par la règle suivante (en plus des règles pour les formules du premier ordre) : si Ψ est un ensemble de formules, alors $\bigvee \Psi$ est une formule. Pour la logique $\mathrm{L}_{\omega_1\omega}$, on remplace cette règle par : si Ψ est un ensemble *dénombrable* de formules, alors $\bigvee \Psi$ est une formule. La sémantique de ces formules est une extension naturelle de celle de la logique du premier ordre : $\mathcal{A} \models \bigvee \Psi$ si et seulement s'il existe $\psi \in \Psi$, tel que $\mathcal{A} \models \psi$. Ces deux logiques sont clairement des extensions de la logique du premier ordre. On montre facilement que ces deux logiques ont le même pouvoir d'expression sur des structures *finies*. Puisque toute classe de structures finies est axiomatisable[5] dans $\mathrm{L}_{\infty\omega}$, cette logique est beaucoup trop puissante dans le cas de structures finies pour aboutir à des résultats. C'est ce qui motive la définition des logiques $\mathrm{L}^s_{\infty\omega}$ et FO^s ($s \leq 1$) qui contiennent seulement les formules dont les variables (libres ou liées) sont parmi $v_1, \ldots, v_s$. On définit alors la logique $\mathrm{L}^\omega_{\infty\omega} := \bigcup_{s\leq 1} \mathrm{L}^s_{\infty\omega}$.

3. Logiques utilisées dans notre étude

Des logiques[6] de point fixe sont introduites, à côté des logiques classiques du premier et second ordre. Elles sont des extensions d'une logique donnée. Nous allons considérer FO(IFP), qui est une extension du premier ordre, et qui contient la logique du premier ordre et est fermée par points fixes d'opérations inflationaires définissables.

4. φ est logiquement équivalente à ψ si $\models \varphi \leftrightarrow \psi$ (elles sont équivalentes pour toute structure et toute assignation)

5. on définit ce que l'on entend par "axiomatisable" dans la partie 5.

6. lorsque l'on parle de logiques ici, à part la vue syntaxe munie d'une sémantique, on peut considérer que ce sont des fragments de la logique du second ordre.

On se fixe un ensemble fini M. Une fonction $F : 2^M \to 2^M$ donne une suite d'ensembles

$$\emptyset, F(\emptyset), F(F(\emptyset)), \ldots$$

On note ces éléments F_i : $F_0 = \emptyset$ et $F_{n+1} = F(F_n)$. Supposons qu'il existe un $n_0 \in \mathbb{N}$ tel que $F_{n_0+1} = F_{n_0}$. Alors $F_m = F_{n_0}$ pour tout $m \geq n_0$. On note F_{n_0} avec F_∞ et on dit que le *point fixe* F_∞ de F *existe* (dans le cas où le point fixe F_∞ n'existe pas, il convient de poser $F_\infty = \emptyset$).

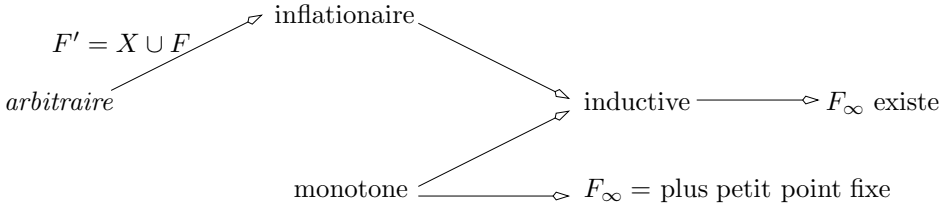
On appelle F *inductive* si $F_n \subseteq F_{n+1}$ pour tous les n, *inflationaire* si $X \subseteq F(X)$ pour tout $X \subseteq M$ et *monotone* si pour tout $X, Y \subseteq M$, $X \subseteq Y \subseteq M$ implique $F(X) \subseteq F(Y)$.

On peut montrer que :

Théorème 3.1. (1) Si F est *inductive* alors F_∞ existe et $F_\infty = F_{\|M\|}$.

(2) Si F est arbitraire et F' est donnée par $F'(X) := X \cup F(X)$, alors F' est inflationaire. Dans le cas où F est inductive, nous avons $F'_n = F_n$ pour tout $n \geq 0$ et donc $F'_\infty = F_\infty$.

Le schéma suivant résume ces relations :



Ceci nous amène à la définition de FO(IFP), la logique de point fixe inflationaire. La syntaxe de FO(IFP) est définie comme suit. Pour un vocabulaire τ, la classe de formules de FO(IFP) de vocabulaire τ contient les formules atomiques de second ordre sur τ et est fermée par les opérations syntaxiques $\neg$, $\vee$ et $\exists x$, et par notre nouvelle opération $[\mathrm{IFP}_{\overline{x},X}.]\overline{t}$ où la longueur de $\overline{x}$[7] et de $\overline{t}$ sont les mêmes et coïncident avec l'arité.

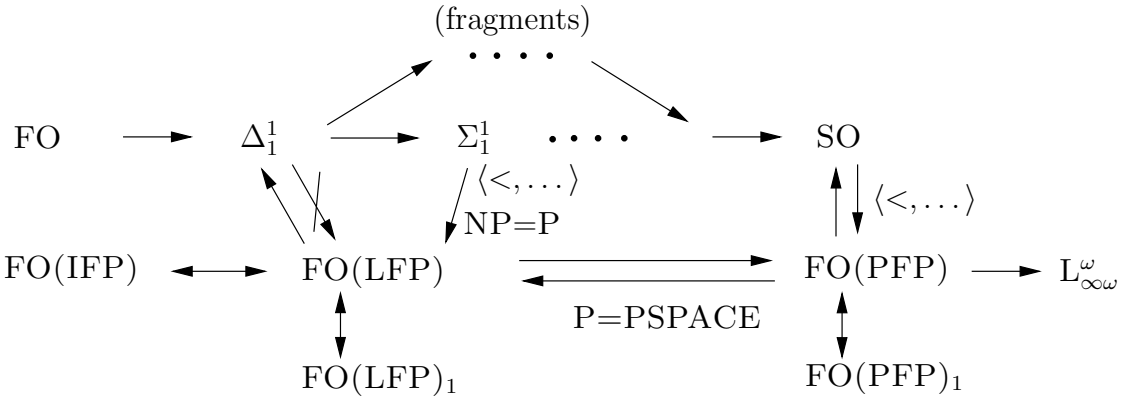
Les énoncés sont les formules sans variables libres du premier ou du second ordre. La sémantique est définie inductivement par les règles ci-dessus, la signification de $[\mathrm{IFP}_{\overline{x},X}\varphi]\overline{t}$ étant $\overline{t} \in F_\infty^{(X\overline{x}\vee\varphi)} = \{\overline{x} \mid X\overline{x} \vee \varphi(\overline{x}, X)\}$ (dont l'existence est prouvée par le théorème ci-dessus).

On définit également d'autres logiques de point fixe. On a, par exemple, la logique de point fixe partiel FO(PFP) qui se définit de la même façon que FO(IFP) mais avec des opérateurs de point de fixe arbitraires (et non plus nécessairement inductifs). Cette logique permet de caractériser la classe de complexité PSPACE des problèmes résolvables en espace polynomial. On définit également la logique de plus petit point fixe FO(LFP) qui est comme FO(IFP) sauf que les formules, que l'on utilise dans notre opération de point fixe, doivent être *positives* (chaque occurence libre d'une variable du second ordre est *précédée* d'un nombre paire de symboles

7. on note $\overline{x}$ pour $x_1, \ldots, x_s$ pour un certain s, que l'on appelle sa longueur.

de négation). On l'appelle ainsi car on montre que si la formule est positive, alors l'opérateur de point fixe est monotone. Gurevich et Shelah [16] ont montré que toute formule de FO(IFP) est équivalente à une formule de FO(LFP). Abiteboul, Vardi et Vianu [1] ont défini une autre logique de point fixe, la logique de point fixe non-déterministe, qui a deux opérateurs de point fixe et on applique d'une façon non-déterministe l'un ou l'autre. Ils ont montré qu'avec cette logique, on peut caractériser NP, ce qui permet de ramener le problème P=NP à une comparaison entre la puissance de deux sortes de points fixes.

Le diagramme suivant résume les comparaisons entre ces différentes logiques et les liens avec les problèmes classiques de complexité. Dans ce diagramme, $\mathrm{FO(LFP)}_1$ désigne FO(LFP) où l'on se restreint à utiliser un seul opérateur LFP; lorsque l'on met $\langle <, \ldots \rangle$ à coté d'une flèche, cela signifie que l'inclusion est vraie seulement pour des structures ordonnées; enfin, les égalités de classes de complexité à côté de flèches signifient également que ces inclusions sont vraies si l'on suppose que les égalités le sont.



Nous arrivons maintenant aux définitions syntaxiques de base. On note les classes de formules du second ordre en spécifiant le préfixe de la partie quantificateur du second ordre et de la partie quantificateur du premier ordre. Devant le préfixe des quantificateurs du second ordre nous mettons un (F) ou un (M) qui indique si l'on considère des logiques fonctionnelles ou avec des prédicats du second ordre. Nous assignons aussi un *préfixe de degré* à n'importe quelle partie des quantificateurs du premier ordre qui indique la plus grande arité des fonctions ou des relations quantifiées. Les autres notations sont assez simples, comme $\forall^*$, qui signifie un ou plusieurs $\forall$, et $\forall_d^*$, au plus d $\forall$. A titre d'exemple nous donnons la définition suivante :

Définition 1. $(\mathrm{F})_d\Sigma_1^1\forall$ est la classe logique de formules de la forme

$$\exists f_1 \ldots \exists f_n \forall x \varphi$$

où les f_i sont des fonctions d'arité au plus d et φ est une formule du premier ordre sans quantificateurs.

${}_d\Sigma_1^1\forall$ se définie de façon similaire avec les f_i qui sont maintenant des relations et non plus des fonctions.

4. Différences et similitudes entre l'étude des modèles finis et des modèles arbitraires

On présente tout d'abord les points en commun avec l'étude des modèles arbitraires puis les différences notoires.

4.1. Jeux d'Ehrenfeucht-Fraissé. Les jeux d'Ehrenfeucht-Fraissé forment une notion élémentaire de la théorie des modèles. Ils existent aussi bien pour les modèles arbitraires que finis. Certains auteurs [32] présentent même la théorie des modèles avec ces jeux pour fondation au lieu de parler directement de syntaxe munie d'une sémantique. Ils constituent également un outil très puissant pour montrer l'équivalence élémentaire (ou la non-équivalence) de structures et sont très souvent utilisés en théorie des modèles finis à cet égard.

Soit $\mathcal{A}$ et $\mathcal{B}$ deux structures. Soit [8] p une fonction telle que $p' \subseteq A$ et $p'' \subseteq B$.

Définition 2. p est un isomorphisme partiel de $\mathcal{A}$ dans $\mathcal{B}$ si p est injective, si pour tout $c \in \tau$, $c^{\mathcal{A}} \in p'$ et $p(c^{\mathcal{A}}) \in c^{\mathcal{B}}$ et si pour toute relation n-aire $R \in \tau$ et tout $a_1, \ldots, a_n \in p'$, $R^{\mathcal{A}} a_1 \ldots a_n$ si et seulement si $R^{\mathcal{B}} p(a_1) \ldots p(a_n)$.

Soient $\mathcal{A}$ et $\mathcal{B}$ deux τ-structures, $\overline{a} \in A^s$, $\overline{b} \in B^s$ et $m \in \mathbb{N}$. On appelle jeu d'Ehrenfeucht-Fraissé $G_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$, un jeu qui se joue à deux, le destructeur et le duplicateur. À chaque étape i du jeu, le destructeur choisit une structure $\mathcal{A}$ ou $\mathcal{B}$ et un élément dans cette structure (e_i si c'est dans $\mathcal{A}$, f_i sinon). Le duplicateur choisit alors un élément dans l'autre structure. Le duplicateur gagne cette instance du jeu si au bout de m étapes, $\overline{a}\ \overline{e} \mapsto \overline{b}\ \overline{f}$ est un isomorphisme partiel. Dans le cas contraire, c'est le destructeur qui a gagné. On dit que l'un des joueurs a une stratégie gagnante s'il peut gagner le jeu quoi que fasse l'autre joueur.

Comme nous l'avons mentionné précédemment, cette approche permet de relier l'équivalence élémentaire de deux structures et le fait que deux structures soient isomorphes. Le théorème suivant donne une première relation :

Théorème 4.1.
(1) Si $\mathcal{A} \cong \mathcal{B}$, alors le duplicateur a une stratégie gagnante pour $G_m(\mathcal{A}, \mathcal{B})$ [9].
(2) Si le duplicateur a une stratégie gagnante pour $G_{m+1}(\mathcal{A}, \mathcal{B})$ et $\|\mathcal{A}\| \leq m$, alors $\mathcal{A} \cong \mathcal{B}$.

Définition 3. Soit $\overline{x} = x_1, \ldots, x_s$.

$$\varphi^0_{\overline{a}}(\overline{x}) := \bigwedge \{\varphi(\overline{x}) \mid \varphi \text{ atomique ou la négation d'une atomique}, \mathcal{A} \models \varphi[\overline{a}]\}$$

8. on utilise les notations traditionnelles de la théorie des ensembles : p' et p'' désignent respectivement le domaine et l'image de la fonction p.

9. $G_m(\mathcal{A}, \mathcal{B})$ est le jeu $G_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$ lorsque $s = 0$.

et pour tout $m > 0$,

$$\varphi^m_{\overline{a}}(\overline{x}) := \bigwedge_{a\in A} \exists x_{s+1}\varphi^{m-1}_{\overline{a}a}(\overline{x}, x_{s+1}) \wedge \forall x_{s+1} \bigvee_{a\in A} \varphi^{m-1}_{\overline{a}a}(\overline{x}, x_{s+1}).$$

Ces formules nous permettent de faire le lien avec l'équivalence élémentaire de structures. On rappelle que $\mathcal{A}\equiv_m\mathcal{B}$ signifie que $\mathcal{A}$ et $\mathcal{B}$ satisfont les mêmes formules de rang de quantification inférieure ou égal à m.

Théorème 4.2 (Théorème d'Ehrenfeucht). Les assertions suivantes sont équivalentes [10] :

(1) le duplicateur a une stratégie gagnante pour $G_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$;
(2) $\mathcal{B} \models \varphi^m_{\overline{a}}[\overline{b}]$;
(3) si $\mathrm{qr}(\varphi) \leq m$, alors $\mathcal{A} \models \varphi[\overline{a}]$ si et seulement si $\mathcal{B} \models \varphi[\overline{b}]$.

Pour s=0, la dernière assertion est en réalité $\mathcal{A}\equiv_m\mathcal{B}$. De ce théorème, on en déduit une nouvelle relation avec l'isomorphisme de structures.

Corollaire 4.3. Soit $\mathcal{A}$ une τ-structure telle que $\|\mathcal{A}\| \leq m$, alors $\mathcal{B} \models \varphi^{m+1}_{\mathcal{A}}$ si et seulement si $\mathcal{A}\cong\mathcal{B}$.

On définit alors le *Va et vient* de Fraissé qui nous donne la relation finale entre isomorphisme de structures et jeux d'Ehrenfeucht.

Définition 4. $\mathcal{A}\cong_m\mathcal{B}$ s'il existe $(I_j)_{j\leq m}$ tel que :

- I_j est un ensemble non vide d'isomorphismes partiels de $\mathcal{A}$ vers $\mathcal{B}$;
- (*Va*) pour tout $j < m$, $p \in I_{j+1}$, et $a \in A$, il existe $q \in I_j$ tel que $q \supseteq p$ et $a \in q'$;
- (*Vient*) pour tout $j < m$, $p \in I_{j+1}$, et $b \in B$, il existe $q \in I_j$ tel que $q \supseteq p$ et $b \in q$".

Théorème 4.4. Les assertions suivantes sont équivalentes :

(1) Le duplicateur a une stratégie gagnante pour $G_m(\mathcal{A}, \mathcal{B})$;
(2) $\mathcal{A}\cong_m\mathcal{B}$;
(3) $\mathcal{A}\equiv_m\mathcal{B}$;
(4) $\mathcal{B} \models \varphi^m_{\mathcal{A}}$.

De par ces équivalences, on parlera désormais de jeu d'Ehrenfeucht-Fraissé.

Exemple 4.1. On montre grâce à ces notions que la classe des structures dont le domaine est de cardinalité paire n'est pas axiomatisable dans la logique du premier ordre. Soient τ le vocabulaire vide et $\mathcal{A}$ et $\mathcal{B}$ des structures sur τ (en fait, des ensembles non vides). On suppose que $\|A\| \geq m$ et $\|B\| \geq m$. Alors $\mathcal{A}\cong_m\mathcal{B}$ ($(I_j)_{j\leq m} : \mathcal{A}\cong_m\mathcal{B}$ avec $I_j := \{p \in \mathrm{Part}(\mathcal{A}, \mathcal{B}) \mid \|\mathrm{do}(p)\| \leq m - j\}$ [11]. On a donc que la classe des τ-structures de cardinalité paire n'est pas axiomatisable dans la

10. le rang de quantificateur $\mathrm{qr}(\varphi)$ d'une formule φ est le nombre maximum de quantificateurs emboités dans φ : $\mathrm{qr}(\varphi) := 0$ si φ est atomique, $\mathrm{qr}(\neg\varphi) := \mathrm{qr}(\varphi)$, $\mathrm{qr}(\varphi\wedge\psi) := \max\{\mathrm{qr}(\varphi), \mathrm{qr}(\psi)\}$ et $\mathrm{qr}(\exists x\varphi) := \mathrm{qr}(\varphi) + 1$

11. $\mathrm{Part}(\mathcal{A}, \mathcal{B})$ est l'ensemble des isomorphismes partiels de $\mathcal{A}$ vers $\mathcal{B}$.

logique du premier ordre : pour chaque m, soit $\mathcal{A}_m$ la structure de cardinalité m. Alors, $\mathcal{A}_m$ est une structure de cardinalité paire si et seulement $\mathcal{A}_{m+1}$ n'en est pas une; mais $\mathcal{A}_m \cong_m \mathcal{A}_{m+1}$. Soit alors φ un énoncé du premier ordre. On prend $m := \mathrm{qr}(\varphi)$. Puisque $\mathcal{A}_{m+1}$ n'est pas de cardinalité paire et $\mathcal{A}_m \cong_m \mathcal{A}_{m+1}$, la classe des τ-structures de cardinalité paire n'est pas égale à la classe des modèles satisfaisant φ. On montre de façon similaire que la classe des ordres finis de cardinalité paire n'est pas axiomatisable dans la logique du premier ordre (on peut également pour tout vocabulaire τ).

4.2. Propriétés de la logique du premier ordre. Il y a des changements d'un point de vue "théorie des modèles", lorsque l'on étudie la logique du premier ordre non plus dans le cas de modèles arbitraires mais lorsque l'on se cantonne à des modèles finis. On sait, par exemple, que le théorème de compacité est faux dans le cas où on ne se restreint qu'à des modèles finis. D'autres propriétés telles les théorèmes de Beth et de Craig ne sont plus valables dans le cas finis. Ces changements nous permettent de mieux comprendre exactement où se situe le problème pour certaines questions de complexité.

La logique du premier ordre a été utilisée comme fondation pour analyser la notion de preuve mathématique. Un résultat de cette analyse est le théorème de complétude de Gödel. La question est de savoir si pour toute conséquence $\Phi \models \varphi$, il existe une preuve de φ à partir Φ. Pour répondre à cette question, Gödel utilise une notion de *preuve formelle* qui est basée sur un système fini de règles formelles. Une preuve formelle de φ à partir de Φ consiste en une suite d'applications de ces règles arrivant à φ à partir des formules de Φ. Gödel a montré :

Théorème 4.5 (Théorème de Complétude). φ est une conséquence de Φ si et seulement si φ est démontrable à partir de Φ.

Une conséquence du théorème précédent est :

Théorème 4.6 (Théorème de Compacité). (1) Si φ est une conséquence de Φ, alors φ est une conséquence d'un sous-ensemble fini de Φ.
(2) Si tout sous-ensemble fini de Φ est satisfiable, alors Φ est satisfiable.

La preuve du théorème de complétude amène à démontrer le célèbre théorème de la théorie des modèles :

Théorème 4.7 (Théorème de Löwenheim-Skolem). Si Φ a un modèle, alors Φ a un modèle au plus dénombrable.

Ceci est la version montante du théorème de Löwenheim-Skolem. La version descendante dit que Φ a alors un modèle de cardinalité aussi grande que l'on veut.

Le théorème de compacité devient faux lorsqu'on se restreint à des structures finies. Il suffit de considérer l'ensemble des formules φ_n exprimant que l'univers du modèle est de cardinalité $\geq n$. Chaque sous-ensemble de cet ensemble a un modèle fini mais l'ensemble tout entier n'a pas de modèle fini.

4.3. Propriété de Beth. La propriété de Beth montre clairement où se situe la différence majeure entre l'étude des modèles arbitraires (éventuellement infinis) et l'étude des modèles finis. On montre qu'elle n'est plus valable pour la logique du premier ordre restreinte à des structures finies, ce qui signifie que lorsque l'on va vouloir préciser qu'un problème est exprimable dans une logique, il faudra avoir recours à une définissabilité explicite et non implicite (c'est ce que l'on fait avec l'*axiomatisabilité*). Elle exprime, de manière plus particulière pour les modèles finis, l'idée sous-jacente dans les jeux d'Ehrenfeucht-Fraissé : la comparaison entre isomorphisme et équivalence élémentaire.

Soit $\mathcal{L}$ une logique, R une relation n-aire n'appartenant pas à notre vocabulaire τ.

Définition 5. (1) Une $\mathcal{L}[\tau \cup \{R\}]$-formule φ définit *implicitement* R si toute τ-structure $\mathcal{A}$ a au plus une extension $(\mathcal{A}, R^{\mathcal{A}})$, qui soit une $(\tau \cup \{R\})$-structure, qui satisfasse φ;

(2) R est définit explicitement relativement à φ s'il existe une $\mathcal{L}[\tau]$-formule $\psi(\overline{x})$ telle que $\varphi \models \forall \overline{x}(R\overline{x} \leftrightarrow \psi(\overline{x}))$.

La théorème de Beth sur FO est que si R est définissable implicitement par φ alors R est également définissable explicitement relativement à φ.

Théorème 4.8. La propriété de Beth n'est pas vérifiée pour FO dans le cas fini.

Preuve. On sait que la classe des ordres finis de cardinalité paire n'est pas axiomatisable dans FO. Or la conjonction des axiomes pour un ordre et la formule $\neg R\text{min} \wedge \forall x \forall y(Sxy \rightarrow (Rx \leftrightarrow \neg Ry))$ définissent implicitement la relation unaire R, qui est l'ensemble des paires dans notre univers. Si on suppose que la propriété de Beth est vraie, alors $\psi(\text{max})$ (c'est le même ψ que dans la définition), avec les formules pour les axiomes d'un ordre, axiomatise cette classe des ordres finis de cardinalité paire. D'où la contradiction. □

La preuve utilise donc le fait que cette classe n'est pas définissable en logique du premier ordre fini, ce qui se démontre en utilisant les jeux d'Ehrenfeucht-Fraissé. On revient à nouveau au problème de la différence entre isomorphisme et équivalence élémentaire de structures.

4.4. Propriété de Craig. Une logique $\mathcal{L}$ a la propriété d'*interpolation* (ou la propriété de *Craig*) si pour tous vocabulaires σ et τ, pour toutes formules φ et ψ (de vocabulaires respectifs σ et τ) telles que $\varphi \models \psi$, il existe une $\mathcal{L}[\sigma \cap \tau]$-formule interpolante ξ telle que $\varphi \models \xi$ et $\xi \models \psi$.

Le théorème de Craig dit que la logique du premier ordre FO a cette propriété. On montre, en utilisant encore des jeux d'Ehrenfeucht-Fraissé, que la logique du premier ordre restreinte à des structures finies n'a pas cette propriété.

On montre cependant que ces deux propriétés (Beth et Craig) restent vraies dans le cas fini pour la logique $\mathcal{L}_{\omega_1\omega}$ (elles le sont déjà dans le cas de structures arbitraires pour cette logique).

4.5. Le pouvoir d'expression de FO. Soit une τ-structure $\mathcal{A}$. On dit que a et b (éléments de A) sont voisins si $a \neq b$ et s'il existe $R \in \tau$ et $\overline{c} \in A$ tels que $R^{\mathcal{A}}\overline{c}$ et a et b soient des composantes de $\overline{c}$. À partir de cette définition, on munit la structure $\mathcal{A}$ d'une distance $d_{\mathcal{A}}$ telle que $d_{\mathcal{A}}(a,b)$, pour $a, b \in A$, soit la longueur de la plus petite suite $(u^{a,b})^i$ telle que $u_0^{a,b} = a$, $u_{d_{\mathcal{A}}(a,b)}^{a,b} = b$ et pour tout $0 \leq i < d_{\mathcal{A}}(a,b)$, $u_i^{a,b}$ et $u_{i+1}^{a,b}$ soient voisins dans $\mathcal{A}$.

Pour $a \in A$ et $r \in \mathbb{N}$, $S_{\mathcal{A}}(r,a)$ (ou $S(r,a)$ si on sait de quelle structure on parle) désigne la r-sphère de a : $S(r,a) := \{b \in A \mid d_{\mathcal{A}}(a,b) \leq r\}$. $\mathcal{S}(r,a)$ désigne la sous-structure de $\mathcal{A}$ avec univers $S(r,a)$.

On définit alors le type r-sphère d'un point a dans $\mathcal{A}$ par le type d'isomorphisme de $(\mathcal{S}(r,a),a)$. Le théorème suivant nous donne un apperçu du pouvoir d'expression de la logique du premier ordre.

Théorème 4.9 (Théorème de Hanf). Soit $\mathcal{A}$ et $\mathcal{B}$ deux τ-structures et $m \in \mathbb{N}$. Si pour un certain $e \in \mathbb{N}$, les 3^m-sphères dans $\mathcal{A}$ et $\mathcal{B}$ ont moins de e éléments et si pour tout $n \leq 3^m$ et type n-sphère ı, soit $\mathcal{A}$ et $\mathcal{B}$ ont le même nombre d'éléments de type n-sphère ı, soit ils ont plus de $m \cdot e$ éléments de type n-sphère ı, alors $\mathcal{A} \equiv_m \mathcal{B}$.

On dit qu'un sous-ensemble M d'un τ-structure $\mathcal{A}$ est l-éparpillé si pour tout $a, b \in M$, $d_{\mathcal{A}}(a,b) > l$. Soient $r, n \geq 1$ et une τ-formule $\varphi(x)$, on peut ecrire un énoncé du premier ordre exprimant qu'il existe un sous-ensemble M $2r$-éparpillé, dont la cardinalité est au moins n, tel que $\mathcal{S}(r,a) \models \varphi[a]$ pour tout $a \in M$. Le théorème suivant établit que tout énoncé du premier ordre est logiquement équivalent à une combinaison booléenne de tels énoncés, que l'on appelle énoncés locaux. C'est une reformulation de l'idée déjà présente dans le théorème de Hanf, selon laquelle les énoncés de premier ordre peuvent seulement saisir des propriétés locales des structures.

Théorème 4.10 (Théorème de Gaifman). Tout énoncé de la logique du premier ordre est logiquement équivalent à un énoncé local.

Ces théorèmes montrent que la logique du premier ordre ne s'attarde que sur des phénomènes locaux. Cela renforce l'idée que la logique du premier ordre est un bon candidat pour être à la base d'une théorie (complexité descriptive), dont le but est de saisir (savoir exprimer) les problèmes de l'informatique, où tout est principalement local.

5. Complexité descriptive

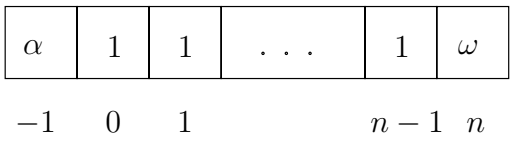
Nous voulons caractériser la complexité en temps (ou en espace) d'un problème par la complexité de sa "définition logique". Essentiellement, ceci est un aspect de théorie des modèles plutôt que de logique (générale) : nous disons qu'un langage $\mathcal{L}$ (pour un problème, le langage des entrées pour lesquelles la réponse à la question du problème est OUI) est défini par un énoncé σ dans une certaine logique si $\mathcal{L}$ correspond *d'une certaine manière* aux modèles qui valident σ. Nous disposons de plusieurs manières de réaliser cette correspondance entre modèles (structures et langages).

Une des manières les plus naturelles est l'encodage de mots dans des structures. Soit τ le vocabulaire $\{<, X\}$, où $<$ est binaire et X unaire. Pour un mot donné $u \in \{0,1\}^+$, nous considérons des structures de la forme $(A, <, X)$, où la cardinalité de A est égale à la longueur de u, où $<$ est un ordre sur A et où X correspond aux positions dans u d'un 1. Nous les appelons *modèles de mots* pour u.

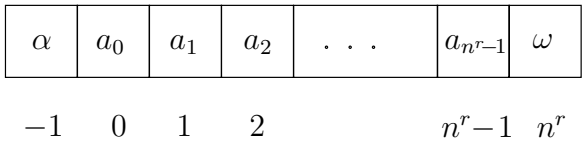
Une autre manière consiste à définir *des machines de Turing avec des structures comme entrées*. Autrement dit, nous construisons une correspondance des structures vers les entrées des machines de Turing à plusieurs rubans, et ce dans ce seul sens.

Nous rappelons comment les structures peuvent être considérées comme des entrées des machines de Turing. Soit $\mathcal{A} \in \mathcal{O}(\tau)$ [12] une structure ordonnée avec $||A|| = n$.

Supposons $\tau = \tau_0 \cup \tau_1$, $\tau_1 = \{R_1, \ldots, R_k, c_1, \ldots, c_l)$ et $\{<\} \subseteq \tau_0 \subseteq \{<, S, \min, \max\}$. Une *machine de Turing pour des structures de type* τ a $1 + k + l$ *rubans d'entrée* (on parle également de *bandes*) et m *rubans de travail* pour un certain $m \geq 1$. A une structure ordonnée $\mathcal{A}$ de type τ nous associons l'entrée suivante sur les $1+k+l$ rubans d'entrée : le ruban 0 contient une suite de 1 de longueur $n := ||A||$.



Pour $1 \leq i \leq k$, le ruban d'entrée i contient l'information sur R_i encodé comme suit : supposons que R soit r-aire. Pour $j < n^r$, soit $|j|_r$ le $j$ème r-tuple dans l'ordre lexicographique de $\{0, \ldots, n-1\}^r$. Alors le ruban d'entrée i contient



où $a_j = 1$ si et seulement si $R^{\mathcal{A}}|j|_r$. Pour $1 \leq i \leq l$, le ruban d'entrée $(k+i)$ contient la représentation binaire de $j := c_i^{\mathcal{A}}$ sans les zéros du début.

Sur chacun de nos rubans d'entrée, le symbole α précède l'entrée et le symbole ω indique la fin du mot d'entrée.

Nous pouvons maintenant donner quelques autres définitions pour les classes habituelles de complexité. La classe P est la classe des classes de structures finies $\mathcal{K}$ telles qu'il existe une machine de Turing déterministe polynomiale qui reconnaît seulement les structures dans $\mathcal{K}$.

On peut facilement montrer que les deux définitions de ces classes de compléxité (P et $\mathcal{K}$) sont équivalentes aux transitions près. De surcroît, on peut montrer le

12. on note ainsi la classe des structures sur τ contenant un ordre total: τ contient $<$ (l'ordre) et éventuellement la fonction successeur S sur l'ordre et les constantes min (ou 0) et max, interprétées comme le plus petit et le plus grand élément de l'ordre.

théorème suivant, qui témoigne de la raisonabilité de notre étude :

Lemme 5.1. Soient $\mathcal{C}_1$ (resp. $\mathcal{C}_2$) et $\mathcal{C}'_1$ (resp. $\mathcal{C}'_2$) deux mêmes classes de complexité, mais avec des définitions différentes pour les entrées. Alors

$$\mathcal{C}_1 \subseteq \mathcal{C}_2 \quad \text{si et seulement si} \quad \mathcal{C}'_1 \subseteq \mathcal{C}'_2$$

Nous arrivons maintenant à quelques définitions formelles :

Définition 6. Une logique $\mathcal{L}$ *capture* une classe de complexité C si pour tout τ avec $<\in \tau$ et $K \subseteq \mathcal{O}(\tau)$, nous avons

$$K \in \mathcal{C} \;\text{ si et seulement si }\; K \text{ est axiomatisable dans } \mathcal{L}.$$

On notera la *capture* par le symbole $\rightleftharpoons$ (ou si c'est seulement dans un sens par $\leftharpoondown$ ou $\rightharpoonup$).

Nous rappelons que chaque classe $\mathcal{K}$ de structures finies est évidemment définissable par un ensemble Φ d'énoncés du premier ordre ($\mathcal{K} = \mathrm{Mod}(\Phi)$), mais ici nous sommes intéressés par la possibilité de les définir à l'aide d'une seule énoncé, ce que nous appelons *axiomatisabilité*(le fait d'être axiomatisable).

6. Caractérisation logique de P et NP

Nous rappelons maintenant les résultats classiques, comme le fait que FO(IFP) et Σ^1_1 capturent respectivement P et NP. Ceci est fait en décrivant comment passer d'une configuration à une autre, en partant de la configuration initiale et en arrivant à une configuration finale en temps polynomial.

Nous allons utiliser exactement les mêmes notations, et aussi une partie des définitions de [4] (Ch. 6). Nous n'allons pas les rappeler ici.

Nous allons décrire brièvement la preuve de la capture de P, car la preuve de notre résultat principal en dépend crucialement.

Lemme 6.1. Soit $K \subseteq \mathcal{O}(\tau)$ une classe de structures ordonnées sur τ. Si K est dans P, alors K est axiomatisable dans FO(IFP).

Preuve. Supposons que $K \in \mathcal{C} = \mathrm{P}$, et soit M une machine de Turing témoignant que $K \in \mathcal{C}$. Nous allons décrire le comportement de M par une formule $\varphi_M \in \mathcal{L} = \mathrm{FO(IFP)}$ d'une telle manière que pour toute structure ordonnée $\mathcal{A} \in K$,

$$\mathcal{A} \models \varphi_M \quad \text{ si et seulement si } \quad M \text{ accepte } \mathcal{A}$$

Nous fixons un vocabulaire $\tau = \tau_0 \cup \tau_1$ où τ_0 est $\{<, S, \min, \max\}$ et $\tau_1 = \{R_1, \ldots, R_k\}$ est relationnel avec R_i r_i-aire. Donc notre machine de Turing M a $1 + k$ rubans d'entrée. Nous supposons que M a m rubans de travail. Puisque la taille de l'espace parcouru par la machine de Turing est inférieure au temps mis pour arriver à l'etat final, on peut considérer que les rubans sont de taille égale au temps maximum mis pour arriver à l'etat final.

Nous sommes maintenant en mesure de décrire les configurations. Pour des raisons de facilité, nous étendons la définition d'un successeur d'une configurations en posant que toute configuration d'*acceptation* est son propre successeur.

Remarque. Notons que nous pouvons supposer que $n > k + m$ et donc n code les états de M dans l'univers de $\mathcal{A}$.

Nous avons supposé que $K \in \mathrm{P}$, donc M va arriver dans l'état final accepteur dans un temps $O(n^d)$ pour un certain d.

Nous codons une configuration C avec l'ensemble suivant de $(d+2)$-tuples (qui est une relation $(d+2)$-aire dans notre modèle) [13] :

$$\begin{aligned}
\mathrm{C} \;:=\; & \{(0,0)\} \times \{\tilde{0}\} \times \textsc{State}_{\mathrm{C}} \\
\cup\; & \bigcup_{0 \le j \le k+m} \{(1,j)\} \times \{\tilde{0}\} \times \textsc{Frontiers}^j_{\mathrm{C}} \\
\cup\; & \bigcup_{0 \le j \le k} \{(2,j)\} \times \{\tilde{0}\} \times \textsc{InputHead}^j_{\mathrm{C}} \\
\cup\; & \bigcup_{k < j \le k+m} \{(3,j)\} \times \{\tilde{0}\} \times \textsc{WorkHead}^j_{\mathrm{C}} \\
\cup\; & \bigcup_{k < j \le k+m} \{(4,j)\} \times \textsc{Punched}^j_{\mathrm{C}}
\end{aligned}$$

où $\textsc{State}_{\mathrm{C}}$ est $\{s\}$ où s est l'état de C; $\textsc{Frontiers}^j_{\mathrm{C}}$ est $\{0\}$ (resp. $\{n-1\}$) si la tête du j-ème ruban est en face de α (resp. ω) et $\emptyset$ sinon; $\textsc{InputHead}^j_{\mathrm{C}}$ est $\{|i|_{r_j}\}$ si la tête du j-ème ruban lit la i-ème case qui ne contient pas ω, $\emptyset$ sinon; $\textsc{WorkHead}^j_{\mathrm{C}}$ est $\{|i|_d\}$ si la tête du j-ème ruban lit la $i$ème case, $\emptyset$ sinon; et enfin $\textsc{Punched}^j_{\mathrm{C}}$ est l'ensemble de tous les $|i|_d$ tels que la i-ème case du j-ème ruban de travail contient le symbole 1.

Le premier couple dans C sert de sélection, les $\tilde{0}$ servent à compléter la relation en une relation $d+2$-aire et le reste est l'information correspondant à la sélection.

Maintenant que nous avons un codage approprié (la configuration est déterminée de manière unique) pour des configurations bornées par n^d, nous pouvons décrire le comportement de M et le fait qu'il existe une configuration d'acceptation dans la logique FO(IFP). Nous avons besoin de dire dans FO(IFP) qu'au temps 0 la configuration est celle de départ, et du temps t au temps $t+1 < n^d$, la machine avance de la configuration courante à son successeur (machine déterministe). M acceptant son entrée équivaut à «la (n^d-1)-ème configuration de M est définie et a comme état l'état s_f (l'état final)». Ceci peut être fait à l'aide d'un processus inflationaire dans lequel, après chaque étape, la nouvelle configuration obtenue a une «marque du temps» : à l'étape i, nous allons avoir l'union de tous les ensembles $\{|k|_d\} \times \mathrm{C}_k$ pour $k < i$, où C_k est la configuration de M au temps i. A la fin du processus inflationaire nous allons obtenir l'union de tous ces ensembles pour

13. on note $\tilde{0}$ pour «autant de 0 que nécessaire».

$k < n^d$, et nous pourrons alors vérifier que cette configuration, dont le temps est marqué à $n^d - 1$, a comme état l'état final s_f.

Utilisant notre codage, le processus inflationaire est formalisé par la proposition[14] :

$$\mathcal{A} \models \underbrace{[\mathrm{IFP}_{\overline{vx},Z}\ \overbrace{(\overline{v} = \widetilde{\min} \wedge \varphi_{\mathrm{start}}(\overline{x}))}^{\overline{x}\in \text{ conf de départ pour } \overline{v}=\tilde{0}} \vee \overbrace{\exists \overline{u}(S^d_{\mathrm{lex}}\,\overline{uv} \wedge \varphi_{\mathrm{succ}}(\overline{x}, Z\overline{u}\_))}^{\overline{x}\in \text{ conf successeur / temps}-1}]\widetilde{\max}\min\min\widetilde{\min}\, s_f}_{\text{c'est fini après } n^d \text{ étapes}}$$

où $\mathcal{A}$ est la structure donnée en entrée à M ; $\overline{u}$, $\overline{v}$ sont des marques temporelles ; $\varphi_{\mathrm{start}}(\overline{x})$ une formule du premier ordre décrivant la configuration de départ ($\mathcal{A} \models \varphi_{\mathrm{start}}(\overline{x})$ si et seulement si $\overline{x} \in \mathrm{C}$ où C est la configuration de départ de M démarrée avec $\mathcal{A}$), et $\varphi_{\mathrm{succ}}(\overline{x}, Y)$ une formule du second ordre sans quantificateurs du second ordre, décrivant que $\overline{x}$ appartient à la configuration successeur de Y (omettant les marques temporelles).

φ_{start} et φ_{succ} sont les formules suivantes ($\overline{x}$ est une abréviation pour $xyx_1 \dots x_d$):

$$\begin{aligned}
\varphi_{\mathrm{start}}(\overline{x}) \quad &:= \quad \overline{x} = \widetilde{0} \\
&\quad\ \vee (x = 2 \wedge 0 \leq y \leq k \wedge x_1 \dots x_d = \widetilde{0}) \\
&\quad\ \vee (x = 3 \wedge k+1 \leq k+m \wedge x_1 \dots x_d = \widetilde{0}) \\
\varphi_{\mathrm{succ}}(\overline{x}, X) &:= (X00\widetilde{0}s_f \wedge X\overline{x}) \vee \bigvee_{\mathrm{instr} \in \mathrm{instr}(M)} \varphi_{\mathrm{instr}}(\overline{x}, X)
\end{aligned}$$

où $\mathrm{instr}(M)$ sont les *instructions* de M, i.e. les transitions d'un état et une lettre vers un nouvel état, une nouvelle lettre et un déplacement.

Les instructions sont de la forme

$$sb_0 \dots b_k \dots c_1 \dots c_m \to s'c'_1 \dots c'_m h_0 \dots h_{k+m}$$

ce qui veut dire «quand la machine est dans l'état s et elle lit $b_0 \dots b_k$ sur le ruban d'entrée et $c_1 \dots c_m$ sur les rubans de travail, alors elle peut aller dans l'état s', écrire $c'_1 \dots c'_m$ sur les rubans de travail et déplacer la tête du i-ème ruban de h_i cases». Nous appelons $s\overline{b}\overline{c}$ la base de l'instruction.

$\varphi_{\mathrm{instr}}(\overline{x}, X)$ est la formule $\varphi^{s,\overline{b},\overline{c}}_{\mathrm{base}}(X) \wedge \varphi^{s',\overline{c'},\overline{h}}_{\mathrm{result}}(\overline{x}, X)$ où l'instruction instr est $s\overline{b}\overline{c} \to s'\overline{c'h}$. Bien sûr, $\varphi^{s,\overline{b},\overline{c}}_{\mathrm{base}}(X)$ assure que dans la configuration X la machine est dans l'état s et a respectivement $\overline{b}$ et $\overline{c}$ sur ses rubans d'entrée et de travail ; aussi $\varphi^{s',\overline{c'},\overline{h}}_{\mathrm{result}}(\overline{x}, X)$ assure que $\overline{x}$ appartient à la configuration résultant de l'*exécution* de l'instruction instr commençant dans la configuration X.

14. nous utilisons la notation $\widetilde{a}$ pour «autant de a que nécessaire».

Nous finissons en définissant $\varphi^{s,\overline{b},\overline{c}}_{\text{base}}(X)$ utilisant les abbréviations $\textsc{Frontier}yz$ pour $X1y\tilde{0}z$, $\textsc{IHead}y\overline{z}$ pour $X2y\tilde{0}\overline{z}$, $\textsc{WHead}y\overline{z}$ pour $X3y\tilde{0}\overline{z}$ et $\textsc{Punched}y\overline{z}$ pour $X4y\overline{z}$:

$X00\tilde{0}s$
"s est l'état"
$\wedge \bigwedge_{b_j=\alpha} \textsc{Frontier}j\min \wedge \bigwedge_{c_j=\alpha} \textsc{Frontier}(k+j)\min$
"têtes sur α"
$\wedge \bigwedge_{b_j=\omega} \textsc{Frontier}j\max$
"têtes sur ω"
$\wedge \bigwedge_{b_j=1} \exists x_1 \ldots \exists x_{r_j} (\textsc{IHead}j\tilde{0}x_1 \ldots x_{r_j} \wedge R_j x_1 \ldots x_{r_j})$
"têtes des rubans d'entrée en face d'un 1"
$\wedge \bigwedge_{b_j=0} \exists x_1 \ldots \exists x_{r_j} (\textsc{IHead}j\tilde{0}x_1 \ldots x_{r_j} \wedge \neg R_j x_1 \ldots x_{r_j})$
"têtes des rubans d'entrée en face d'un 0"
$\wedge \bigwedge_{c_j=1} \exists x_1 \ldots \exists x_d (\textsc{WHead}(k+j)x_1 \ldots x_d \wedge \textsc{Punched}(k+j)x_1 \ldots x_d)$
"têtes des rubans de travail en face d'un 1"
$\wedge \bigwedge_{c_j=0} \exists x_1 \ldots \exists x_d (\textsc{WHead}(k+j)x_1 \ldots x_d \wedge \neg\textsc{Punched}(k+j)x_1 \ldots x_d)$
"têtes des rubans de travail en face d'un 0"

En utilisant les mêmes abréviations, on définit $\varphi^{s',\overline{c'},\overline{h}}_{\text{result}}(\overline{x}, X)$ par

$$\Big(\bigwedge_{\substack{h_j=1 \\ k+1\le j\le k+m}} \neg\textsc{WHead}j\widetilde{\max}\Big) \wedge \psi$$

"les têtes des rubans de travail qui avancent vers la droite
ne sont pas en face de la case $n^d - 1$"

et ψ par :

$(x = y = 0 \wedge x_1 \ldots x_{d-1} = \tilde{0} \wedge x_d = s')$
"s' est le nouvel état"
$\vee \bigvee_{k+1\le j\le k+m} (\neg\textsc{WHead}j x_1 \ldots x_d \wedge \textsc{Punched}j x_1 \ldots x_d \wedge x = 4 \wedge y = j)$
"le contenu des cases, des rubans de travail, non parcourus n'est pas modifié"
$\vee \bigvee_{c'_j=1} (\textsc{WHead}j x_1 \ldots x_d \wedge x = 4 \wedge y = k+j)$
"nouveau contenu =1 sur les cases, des rubans de travail, parcourus"
$\vee \bigvee_{h_j=1} (\textsc{Frontier}j0 \wedge (x = 2 \vee x = 3) \wedge y = j \wedge x_1 \ldots x_d = \tilde{0})$
"têtes sur α et allant à droite tombent à la position 0"

$\vee \quad \bigvee_{h_j=0} (\textsc{Frontier}j0 \wedge x = 1 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$

"têtes sur α qui ne bougent pas"

$\vee \quad \bigvee_{h_j=-1} (\textsc{Frontier}j\mathrm{max} \wedge (x = 2 \vee x = 3) \wedge y = j \wedge x_1 \dots x_{d-r_j} = \tilde{0} \wedge x_{d-r_j+1} \dots x_d = \widetilde{\mathrm{max}})$

"têtes sur ω et allant à gauche tombent à la position $n^{r_j} - 1$"

$\vee \quad \bigvee_{h_j=0} (\textsc{Frontier}j\mathrm{max} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_{d-1} = \tilde{0} \wedge x_d = \mathrm{max})$

"têtes sur ω qui ne bougent pas"

$\vee \quad \bigvee_{\substack{h_j=-1 \\ 0 \le j \le k}} (\textsc{IHead}j\tilde{0} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$

"têtes de rubans d'entrée sur α après déplacement"

$\vee \quad \bigvee_{\substack{h_j=-1 \\ k+1 \le j \le k+m}} (\textsc{WHead}j\tilde{0} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_d = \tilde{0})$

"têtes de rubans de travail sur α après déplacement"

$\vee \quad \bigvee_{\substack{h_j=1 \\ 0 \le j \le k}} (\textsc{IHead}j\tilde{0}\, \widetilde{\mathrm{max}}^{r_j} \wedge x = 1 \wedge y = j \wedge x_1 \dots x_{d-1} = \tilde{0} \wedge x_d = \mathrm{max})$

"têtes de rubans d'entrée sur ω après déplacement"

$\vee \quad \bigvee_{0 \le j \le k} \exists u_1 \dots \exists u_d (\text{“}x_1 \dots x_d = u_1 \dots u_d + h_j\text{”} \wedge \textsc{IHead}ju_1 \dots u_d \wedge x = 2 \wedge y = j)$

"têtes de rubans d'entrée qui sont ni sur α ni sur ω"

$\vee \quad \bigvee_{k+1 \le j \le k+m} \exists u_1 \dots \exists u_d (\text{“}x_1 \dots x_d = u_1 \dots u_d + h_j\text{”} \wedge \textsc{WHead}ju_1 \dots u_d \wedge x = 3 \wedge y = j)$

"têtes de rubans de travail qui sont ni sur α ni sur ω"

Ceci complète la preuve de notre lemme. □

Pour montrer que FO(IFP) capture exactement P, on doit montrer l'implication réciproque : si $\mathcal{A} \models \varphi$ où φ est énoncé de FO(IFP), alors il existe une machine de Turing M telle que M accepte $\mathcal{A}$. Nous n'allons pas réécrire cette partie de la preuve parce qu'elle n'est pas utile pour la preuve de notre résultat principal.

Remarque. Ce que nous venons de faire ici est la capture de la complexité temporelle P, ou, pour être plus exact, de P′, en utilisant les notations employées dans le lemme 5.1. Autrement dit, ici nous avons utilisé le codage plutôt artificiel des structures par des entrées de machines de Turing, mais nous pouvions aussi travailler avec des structures plus naturelles (pour des mots d'entrée comme pour les machines de Turing) $\langle A, S, <, P_1, \dots, P_k, \min, \max \rangle$ où $P_i(a)$ signifie «la a-ème case du i-ième ruban contient le symbole 1». Tout ce que nous aurions du changer est de différencier l'encodage de InputHead de celui de WorkHead, et de remplacer le $\exists x_1 \dots \exists x_{r_j}$ dans $\varphi^{s,\overline{b},\overline{c}}_{\mathrm{base}}(X)$ par un $\exists x$. Par conséquent, nous aurions toujours la capture de P, mais cette fois exactement P, et non plus seulement *aux transitions près*. Nous allons travailler avec ces structures dans la section suivante, afin de prouver notre résultat principal.

7. Théorème principal

Nous arrivons donc à notre résultat principal :

Théorème 7.1. $\mathrm{NTIME}(n^d) \rightharpoonup (\mathrm{F})_{2d}\Sigma_1^1\forall$.

Noux montrons d'abord le lemme suivant :

Lemme 7.2. Soit $K \subseteq \mathcal{O}(\tau)$ une classe de structures ordonnées. Si K est dans $\mathrm{NTIME}(n^d)$, alors K est axiomatisable en $(\mathrm{F})_d\Sigma_1^1\forall$.

Preuve. On ne répètera pas ici les idées élémentaires décrites dans la preuve de la caractérisation logique de P. En ce qui nous concerne, l'idée phare est de prendre la caractérisation de P, et *de la rendre* non-déterministe en utilisant le fait que l'on connaît précisément la complexité en temps de notre machine M.

Pour une structure particulière $\mathcal{A} \in K$, comme $K \in \mathrm{NTIME}(n^d)$, $\mathcal{A}$ est reconnue par une machine de Turing M en temps Cn^d, où C est un entier constant et n la longueur de l'entrée (ou plus précisément la cardinalité de l'univers de $\mathcal{A}$). A partir d'ici nous n'allons pas travailler avec $\mathcal{A}$, mais avec son univers étendu à $\lceil \sqrt[d]{C} \rceil n$. Appelons c, la constante $\lceil \sqrt[d]{C} \rceil$.

Ceci va nous permettre de dire (dans notre logique) que nous atteignons une «bonne» configuration avant un certain «temps linéaire» (ou temps $O(n^d)$ avec des quantificateurs sur un d-tuple de variables). Nous avons ensuite besoin de revenir à $\mathcal{A}$ avec pour domaine (univers) n. L'idée est de remplacer chaque fonction $f : cn \to cn$ (qui apparaît dans la partie existentielle du second ordre de notre formule finale) par $2c$ fonctions $f_i^0 : n \to n\ (i < c)$ et $f_i^1 : n \to c\ (i < c)$ avec $f_i^0(x) = f(in+x)$ modulo n et $f_i^1(x) = \lfloor f(in + x)/n \rfloor$. Cette transformation (et comment on peut la faire sur une structure aussi simple qu'une structure ordonnée) est expliquée en détail par Grandjean dans [12].

Comme nous l'avons mentionné à la fin de la section précédente, nous considérons des structures encodant d'une manière naturelle les rubans d'entrée de la machine de Turing M afin de capturer les classes *réelles* de complexité, et non pas des classes parallèles.

L'idée principale est de réduire le non-déterminisme au déterminisme, afin de pouvoir utiliser la logique FO(IFP), vraie uniquement (bien sûr !) dans notre restriction de l'étude de $\mathrm{NTIME}(n^d)$ pour un d précis. Le non-déterminisme est réduit au déterminisme si, à chaque pas, lorsqu'*un choix doit être fait*, nous connaissons exactement lequel va être celui qui permettra à la machine de Turing d'arriver plus tard dans un état final d'acceptation. D'habitude nous travaillons sur des structures dont le domaine est n. Mais comme on a une structure avec domaine cn, il est suffisant d'avoir des fonctions S,$C_1, \ldots, C_m$ et $H_0, \ldots, H_{k+m}$ d'arité d, qui donnent, pour une certaine marque temporelle $(t_1, \ldots, t_d)$ respectivement l'état, les changements des rubans de travail et le déplacement des têtes sur chaque ruban afin de pouvoir exprimer le non-déterminisme.

Afin de pouvoir comparer respectivement le contenu des rubans de travail et le déplacement des têtes avec $C_i(\overline{u})$ and $H_i(\overline{u})$, nous devons étendre l'encodage de nos configurations de la manière suivante :

$$\begin{array}{rcl} \mathrm{C} & := & \mathrm{C}_{\mathrm{old}} \\ & \cup & \bigcup_{k\le j\le k+m}\{(5,j)\}\times\{\tilde{0}\}\times \textsc{HeadPunched}^j_{\mathrm{C}} \\ & \cup & \bigcup_{0\le j\le k+m}\{(6,j)\}\times\{\tilde{0}\}\times \textsc{Disp}^j_{\mathrm{C}} \end{array}$$

où $\textsc{HeadPunched}^j_{\mathrm{C}}$ et $\textsc{Disp}^j_{\mathrm{C}}$ sont respectivement le symbole contenu dans la case lue par la tête du j-ème ruban et le dernier déplacement fait par la tête du j-ème ruban (-1 est encodé par 2).

Voici de manière précise les modifications faites à la caractérisation de P par FO(IFP), en tenant compte de cette idée :

$$\begin{array}{lll} \mathcal{A}\in K & \text{ssi} & M \text{ accepte } \mathcal{A} \\ & \text{ssi} & \text{la } (Cn^d)-1\text{-ème configuration de } M, \text{ démarrée avec } \mathcal{A}, \text{ est définie} \\ & & \quad \text{et contient l'état } s_f \\ & \text{ssi} & \mathcal{A}\models \exists S\exists C_1\ldots\exists C_m\exists H_0\ldots\exists H_{k+m} \\ & & \quad [\mathrm{IFP}_{v\overline{x},Z}\varphi(\overline{v},\overline{x},X,S,\ldots,H_{k+m})]\max\min\min\widetilde{\min}\, s_+ \end{array}$$

où $\varphi_{\mathrm{succ}}(\overline{u},\overline{x},X)$ est maintenant :

$$\begin{array}{lll} (\varphi_{\mathrm{acc}}(X)\wedge X\overline{x}) & \vee((& \bigvee_{\mathrm{instr}\in\mathrm{instr}(M)} \varphi_{\mathrm{instr}}(\overline{x},X)) \\ & & \wedge((x=0\wedge y=0\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=S(\overline{u})) \\ & & \quad \vee(\bigvee_{k\le j\le k+m} x=5\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=C_j(\overline{u})) \\ & & \quad \vee(\bigvee_{0\le j\le k+m} x=6\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=H_j(\overline{u})) \\ &)) & \end{array}$$

Nous devons adapter légèrement les autres formules :

$$\begin{array}{rcl} \varphi_{\mathrm{start}}(\overline{x}) & := & \overline{x}=\widetilde{0} \\ & & \vee((x=2\vee x=6)\wedge 0\le y\le k\wedge x_1\ldots x_d=\widetilde{0}) \\ & & \vee((x=3\vee x=4\vee x=5)\wedge k+1\le k+m\wedge x_1\ldots x_d=\widetilde{0}) \end{array}$$

et rajouter les disjonctions suivantes à ψ pour rendre compte des modifications de notre encodage des configurations :

$$\begin{array}{ll} \vee & \bigvee_{\substack{h_j=1\\ 0\le j\le k+m}} (x=6\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=1) \\ \vee & \bigvee_{\substack{h_j=0\\ 0\le j\le k+m}} (x=6\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=0) \\ \vee & \bigvee_{\substack{h_j=-1\\ 0\le j\le k+m}} (x=6\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=2) \\ \vee & \bigvee_{\substack{c_j=a\\ k+1\le j\le k+m}} (x=5\wedge y=j\wedge x_1\ldots x_{d-1}=\tilde{0}\wedge x_d=a) \end{array}$$

Nous observons que φ_{instr} est une formule de classe Σ_1^0 et de profondeur d de quantificateur, et donc φ de profondeur $2d$ de quantificateur.

Utilisant le lemme suivant, on montre qu'il existe $\psi \in (\mathrm{F})_{2d}\Sigma_1^1\forall$ tel que

$$\mathcal{A} \models \varphi \quad \text{ si et seulement si } \quad \mathcal{A} \models \psi$$

Par conséquent, K est la classe de modèles ordonnés d'énoncé $(\mathrm{F})_{2d}\Sigma_1^1\forall$. □

Lemme 7.3. Pour toute formule φ de classe ${}_d\Sigma_1^0$ $(\exists_d^*)$, il existe $\psi \in (\mathrm{F})_d\Sigma_1^1\forall$ telle que pour toutes les structures ordonnées $\mathcal{A}$

$$\mathcal{A} \models [\mathrm{IFP}_{\overline{x},X}\varphi]\overline{t} \quad \text{ si et seulement si } \quad \mathcal{A} \models \psi(\overline{t})$$

Preuve. Par le lemme 7.1.1.(b) de [4], $F_\infty = F_{\max}$ (nous sommes dans une structure ordonnée donc max existe) où $F(X) = \{\overline{x} \mid X\overline{x} \vee \varphi(\overline{x}, X)\}$ et donc est inductive. Pour X unaire, soit

$$\psi(t) := \exists X_0 \ldots \exists X_{\max} \forall x[(X_0 x \to \neg x = x) \wedge \bigwedge_{0 \le i \le \max} (X_{i+1}x \leftrightarrow \varphi(x, X_i))] \wedge X_{\max} t$$

φ est ${}_d\Sigma_1^0$, donc par skolemisation, nous obtenons la formule de classe $(\mathrm{F})_d\Sigma_1^1\forall$ *équivalente* à φ.

Si X est n-aire ($n > 1$), nous appliquons le même principe à un opérateur *Simultané Inflationaire de Point Fixe* équivalent à notre énoncé IFP (nous pouvons toujours nous en sortir avec seulement un $\forall x$, utilisant *le même* chaque fois que nous avons besoin de dire «ceci est toujours vrai»). Nous utilisons les mêmes notations que celles employées pour désigner un S-IFP comme dans [4](p.172) ; la seule différence est que, pour nous, $[\text{S-IFP}_{x_0,X_0,\ldots,x_{d-1},X_{d-1}}\varphi_0, \ldots, \varphi_{d-1}]\overline{t}$ signifie que pour tout i, $t_i \in F^i_{(\infty)}$ (notons que ici nous avons des x_i et non pas des $\overline{x_i}$). Le d, que nous utilisons ici, est la longueur de l'encodage utilisé dans la partie 6.3 de [4]. Les φ_i sont définies de manière évidente à partir des φ comme étant la «projection» sur la i-ème composante (de $\overline{x}$). Nous remarquons immédiatement que $[\text{S-IFP}_{x_0,X_0,\ldots,x_{d-1},X_{d-1}}\varphi_0, \ldots, \varphi_{d-1}]\overline{t}$ et $[\mathrm{IFP}_{\overline{x},X}\varphi]\overline{t}$ sont équivalents sur toute structure ordonnée. Par conséquent nous nous sommes restreints à un seul $\forall x$ dans notre formule finale, décrivant l'opérateur de point fixe de la même manière que pour l'X unaire mais, cette fois (parce qu'il est Simultané de Point Fixe), avec d fois plus de $\exists X_i$. □

Preuve du théorème principal. Par le lemme 7.2 nous obtenons notre résultat.

On remarque que dans l'autre sens, de la logique vers la classe de complexité, on arrive dans $\mathrm{NTIME}(n^d \log n^2)$. Soit $\varphi := \exists f_1 \ldots \exists f_n \forall x \psi$, où ψ n'a pas de quantificateurs. Etant sur une machine non-déterministe, nous pouvons deviner les f_i en temps $n^d \log n^2$ et vérifier $\forall x \psi$ sur eux en temps linéaire. Les détails sont laissés au lecteur intéressé. □

8. Applications et ouvertures

Nous sommes intéressés de pouvoir montrer la séparation de classes de complexité (i.e. la non-équivalence élémentaire) et la non-appartenance d'un problème à une classe de complexité (i.e. le fait qu'une classe n'est pas axiomatisable par une logique). Les jeux d'Ehrenfeucht-Fraissé sont un puissant outil pour cela. On montre dans ce qui suit des jeux d'Ehrenfeucht-Fraissé utiles maintenant que l'on a notre résultat.

8.1. Jeux d'Ehrenfeucht-Fraissé sur d'autres logiques. Comme pour la logique du premier ordre, on peut caractériser l'équivalence élémentaire, restreinte à des formules de rang de quantificateur borné par une constante, par des jeux d'Ehrenfeucht-Fraissé pour d'autres logiques.

On étend, par exemple, facilement cela à la logique du second ordre monadique MSO. On note $\mathcal{A} \equiv_m^{\mathrm{MSO}} \mathcal{B}$ si $\mathcal{A}$ et $\mathcal{B}$ satisfont les mêmes énoncés du second ordre monadique de rang de quantificateur $\leq m$; le rang de quantificateur pour un énoncé du second ordre étant le nombre maximal de quantificateurs, du premier et second ordre, imbriqués. Les règles du jeu sont les mêmes que pour le jeu d'Ehrenfeucht-Fraissé pour la logique du premier ordre, mais maintenant à chaque étape du jeu, le destructeur décide s'il joue un *point* ou un *ensemble*. Les points correspondent aux étapes du jeu pour FO. Lorsque le destructeur joue un ensemble, il choisit un sous-ensemble dans la structure qu'il a choisit (il choisit à chaque étape une structure comme dans le cas du premier ordre) et ensuite c'est au tour du duplicateur de choisir un sous-ensemble dans l'autre structure. Après m étapes, les éléments $a_1, \ldots, a_p$ (resp. $b_1, \ldots, b_p$) et sous-ensembles $A_1, \ldots, A_q$ (resp. $B_1, \ldots, \ldots B_q$) de A (resp. B) ont été choisis ($m = p + q$). Le duplicateur gagne le jeu si $\overline{a} \mapsto \overline{b}$ est un isomorphisme partiel de $(\mathcal{A}, A_1, \ldots, A_q)$ dans $(\mathcal{B}, B_1, \ldots B_q)$. On note ce jeu $\mathrm{MSO} - G_m(\mathcal{A}, \mathcal{B})$.

On montre de la même façon que dans le cas de la logique du premier ordre (c'est-à-dire en introduisant des formules similaires aux $\varphi_{\overline{a}}^m$) le théorème suivant :

Théorème 8.1. $\mathcal{A} \equiv_m^{\mathrm{MSO}} \mathcal{B}$ si et seulement si le duplicateur a une stratégie gagnante pour $\mathrm{MSO} - G_m(\mathcal{A}, \mathcal{B})$.

Nous allons maintenant présenter un jeu à la Ehrenfeucht-Fraissé pour la caractérisation de l'équivalence élémentaire de structures pour les logiques FO^s et $\mathrm{L}^s_{\infty\omega}$. On note $\mathcal{A} \equiv^s \mathcal{B}$ et $\mathcal{A} \equiv^{\mathrm{L}^s_{\infty\omega}} \mathcal{B}$ pour exprimer que les structures $\mathcal{A}$ et $\mathcal{B}$ sont élémentairement équivalentes dans ces logiques respectives. Pour $\overline{a} = a_1 \ldots a_s \in (A \cup \{\Box\})^s$, on appelle le support de $\overline{a}$, l'ensemble des indices i tels que a_i appartienne à A. On définit alors une autre notion d'isomorphisme partiel pour notre jeu :

Définition 7. Soient $\overline{a} \in (A \cup \{\Box\})^s$ et $\overline{b} \in (B \cup \{\Box\})^s$. $\overline{a} \mapsto \overline{b}$ est appelé un *s-isomorphisme partiel* de $\mathcal{A}$ dans $\mathcal{B}$ si $\overline{a}$ et $\overline{b}$ ont même support et $\underline{\overline{a}} \mapsto \underline{\overline{b}}$ est un isomorphisme partiel de $\mathcal{A}$ dans $\mathcal{B}$ où $\underline{\overline{a}}$ et $\underline{\overline{b}}$ sont les mêmes suites que $\overline{a}$ et $\overline{b}$ privées des $\Box$.

Dans le jeu $G^s_m(\mathcal{A}, \overline{a}, \mathcal{B}, \overline{b})$, il y a s cailloux $\triangle_1, \ldots, \triangle_s$ pour $\mathcal{A}$, et s cailloux $\triangledown_1, \ldots, \triangledown_s$ pour $\mathcal{B}$. Au début du jeu, le caillou $\triangle_i$ est mis sur a_i si $a_i \in A$ et en dehors du plateau (du jeu) si $a_i = \square$. De façon similaire, le caillou $\triangledown_i$ est mis sur $b_i \in B$ ou en dehors du plateau. Le jeu se fait en m étapes. À chaque étape, le destructeur choisit une structure, $\mathcal{A}$ ou $\mathcal{B}$, et un caillou ($\triangle_i$ ou $\triangledown_i$) pour cette structure, qu'il soit sur ou en dehors du plateau. Il place alors son caillou sur un élément de la structure choisie et le duplicateur place le caillou correspondant ($\triangledown_i$ ou $\triangle_i$) sur un élément de l'autre structure. On remarquera qu'il peut y avoir plusieurs cailloux sur un élément.

Le duplicateur gagne le jeu si pour $j \leq m$, $\overline{e} \mapsto \overline{f}$ est un s-isomorphisme partiel où $\overline{e}$ sont les éléments sur lesquels sont posés les cailloux $\overline{\triangle}$ ($e_i = \square$ lorsque $\triangle_i$ est en dehors du plateau). $\overline{f}$ correspond de façon similaire aux éléments pointés par les cailloux $\overline{\triangledown}$. $G^s_m(\mathcal{A}, \mathcal{B})$ est le jeu avec $\overline{a} = \overline{b} = \square \cdots \square$.

On montre alors le théorème suivant :

Théorème 8.2. (1) $\mathcal{A} \equiv^s_m \mathcal{B}$ si et seulement si le duplicateur a une stratégie gagnante pour $G^s_m(\mathcal{A}, \mathcal{B})$;
(2) $\mathcal{A} \equiv^{\mathrm{L}^s_{\infty\omega}} \mathcal{B}$ si et seulement si le duplicateur a une stratégie gagnante pour $G^s_\infty(\mathcal{A}, \mathcal{B})$ (le jeu avec une infinité d'étapes).

On remarque que toutes nos logiques avec un seul quantificateur universel (de premier ordre) sont incluses *élémentairement* dans $\Sigma^1_1\mathrm{FO}^1$. Par une combinaison des jeux d'Ehrenfeucht-Fraissé pour les logiques SO et FO^1, on a donc un nouveau moyen puissant de montrer qu'un problème nécessite un temps exponentiel en montrant que ce problème n'est ni dans NP ni dans co-NP : il suffit de montrer que la classe de structures correspondant au problème n'est pas axiomatisable dans $\Sigma^1_1\mathrm{FO}^1$. Cela se fait en montrant que pour chaque m, il existe des structures $\mathcal{A}$ et $\mathcal{B}$ telles que $\mathcal{A} \in K$, $\mathcal{B} \notin K$ et $\mathcal{A} \equiv_m \mathcal{B}$ (dans $\Sigma^1_1\mathrm{FO}^1$). Les jeux d'Ehrenfeucht-Fraissé servent à montrer ce dernier point grâce au théorème 4.2. En montrant que le problème n'est pas axiomatisable dans $\Sigma^1_1\mathrm{FO}^1$, on montre qu'il est axiomatisable dans aucune des logiques par lesquels on a caractérisé $\mathrm{NTIME}(n^k)$ et donc qu'il n'est pas dans NP. On montre de même qu'il n'est pas dans co-NP.

8.2. Les fragments de SO avec des relations de valence bornée. On introduit une nouvelle classe de formules du second ordre existentiel que l'on note $\Sigma^{1\,<k}_1$. Cette classe désigne les formules telles que les relations considérées dans ces formules sont de valence inférieure à k. Ce que l'on entend par valence d'une relation R est le maximum du nombre de voisins dans R que possède un élément (ce sont les mêmes voisins que pour le théorème de Hanf).

Nous allons montrer que si un problème est définissable par une formule de $\Sigma^{1\,\leq k}_1$, alors il est reconnaissable par une machine de Turing déterministe en temps $\leq n^k$. Pour cela, on va d'abord montrer la caractérisation logique exacte des langages réguliers par la logique du second ordre monadique existentiel.

Théorème 8.3. $\mathrm{DTREAL} \quad \rightleftharpoons \quad (\mathrm{M})\Sigma^1_1$

Preuve. On montre facilement que tout langage régulier est définissable par une formule Σ_1^1 de la logique du second ordre monadique : on le montre par induction structurelle sur l'expression régulière correspondant au langage.

Dans l'autre sens, on suppose que notre langage $L \subseteq \Sigma^+$ est définissable dans la logique du second ordre monadique : la classe des modèles finis qui satisfont $\varphi \in$ MSO est l'ensemble des modèles de mots $\mathcal{M}_u$ correspondant aux mots u de L. Soit m le rang de quantificateur de φ et soit $\sim$ la relation d'équivalence sur Σ^+ définie par

$$u \sim v \qquad \text{si et seulement si} \qquad \mathcal{M}_u \equiv_m^{\text{MSO}} \mathcal{M}_v$$

Puisqu'à équivalence logique près, il n'y a qu'un nombre fini d'énoncés de rang de quantificateur $\leq m$, $\sim$ est d'indice finie (a un nombre fini de classes d'équivalence).

Soient $u, v, w \in \Sigma^+$ tels que $u \sim v$. On montre facilement en utilisant notre jeu d'Ehrenfeucht-Fraissé pour la logique du second ordre monadique que $\equiv_m^{\text{MSO}}$ est compatible avec $\lhd$ (somme ordonnée sur les structures). On a donc $\mathcal{M}_u \lhd \mathcal{M}_w \equiv_m^{\text{MSO}} \mathcal{M}_v \lhd \mathcal{M}_w$. Et puisque les structures $\mathcal{M}_u \lhd \mathcal{M}_w$ et $\mathcal{M}_v \lhd \mathcal{M}_w$ sont respectivement isomorphes à $\mathcal{M}_{uw}$ et $\mathcal{M}_{vw}$, on a que $\mathcal{M}_{uw} \equiv_m^{\text{MSO}} \mathcal{M}_{vw}$. $\sim$ est donc invariante.

Puisque par définition, si $\mathcal{M}_u \models \varphi$ et $u \sim v$ alors $\mathcal{M}_v \models \varphi$, L est donc l'union des classes d'équivalences des mots u tels que $\mathcal{M}_u \models \varphi$. Or cette classe d'équivalence, on vient de le montrer, est d'indice fini et invariante. L est donc reconnaissable par un automate. □

On notera que puisque les automates finis déterministes et non-déterministes définissent la même classe de language, DTREAL et NTREAL sont également identiques.

Théorème 8.4. $\Sigma_1^{1\leq k} \rightharpoonup \text{DTIME}(n^k)$

Preuve. Soit $\varphi \in \Sigma_1^{1\leq k}$, pour un certain k que l'on se fixe, telle que $\varphi := \exists R_1 \exists R_s \psi$. La classe des structures de mots satisfaisant φ est bien entendu dans NP. On construit alors une machine de Turing non-déterministe reconnaissant ce langage (classe). Le ruban d'entrée prend le mot que l'on veut reconnaître et la machine non-déterministe devine les relations R_i sur ses rubans de travail. L'idée est de *développer* chaque R_i, de valence $\leq k$, en n^{k-1} relations de valence 0 : chaque *test* par notre machine de Turing est transformé en n^{k-1} *tests* pour lesquels il n'y a plus qu'à deviner des relations de valence 0 (relations 1-aires).

Cela se fait très facilement : pour toute relation R, de valence ≥ 1 (donc de degré $d_R \geq 2$), on décide de *laisser libre* une composante de R et de fixer chaque autres composantes. Le nombre de relations que l'on obtient est égal à $K \cdot n^{k-1}$ (òu K est le nombre de façon de placer les $k-1$ valences dans $d_R - 1$ composantes) car nos relations sont de valences $\leq k$.

Pour chacune de ces n^{k-1} formules de valence 0, on vérifie que notre mot d'entrée correspond bien, en temps n, en utilisant le théorème précédent. Notre langage est donc bien dans $\text{DTIME}(n^k)$. □

On peut montrer également que l'on peut séparer $\Sigma_1^{1<k}$ et $\Pi_1^{1<k}$ en utilisant le théorème de Hanf, ce qui nous conforte dans notre idée que ce sont des logiques naturelles et qu'elles peuvent correspondre à DTIME(n^k). Pour cela, il resterait à montrer que l'on arrive à décrire le fait que la machine de Turing s'arrête avant un temps polynomial en l'entrée dans cette logique. Le problème provient du fait que classiquement, lorsque l'on prouve pour les machines de Turing un résultat dans ce sens, on code le temps (nombre d'étapes déjà effectuées); et il serait difficile de coder le temps dans une relation qui soit de valence $\leq k$. Il faudrait donc utiliser un autre type de codage.

8.3. Lois 0-1**.** La plupart des méthodes que nous avons utilisées jusqu'ici sont des techniques qui ont été développées pour des structures arbitraires. Nous allons maintenant présenter un concept qui est propre aux structures finies et essayer de voir où cela peut être utile pour notre problème.

Définition 8. Pour une classe K de structures (finies) sur un vocabulaire τ, on appelle $\mu(K)$ la *probabilité asymptotique* de K, c'est-à-dire la limite (si elle existe) quand $n \to \infty$ de la fraction des structures dont le domaine est $\{1, \ldots, n\}$, qui sont dans K (nombre de structures de domaine $\{1, \ldots, n\}$ dans K divisé par le nombre de structures de domaine $\{1, \ldots, n\}$). On appelle $\widetilde{\mu}(K)$ la *probabilité asymptotique isomorphe* de K, c'est-à-dire la limite (si elle existe) quand $n \to \infty$ de la fraction de types d'isomorphisme de structures de cardinalité n, qui sont dans K.

On dit alors qu'un énoncé est vrai pour presque toutes les structures finies si la probabilité asymptotique de la classe des modèles finis qui la vérifient est définie et égale à 1. Pour une classe Ψ d'énoncés d'une logique, si $\mu(\psi) = 1$ ou $\mu(\psi) = 0$ pour tout $\psi \in \Psi$, on dit que Ψ satisfait la *loi* 0-1 (pour la probabilité asymptotique isomorphe, on parlera de *loi isomorphe*).

On définit une famille d'énoncés que l'on appelle des axiomes d'extension qui sont de grande utilité dans les démonstrations sur les lois 0-1.

Définition 9. Un $r+1$-*axiome d'extension* est un énoncé

$$\forall v_1 \ldots \forall v_r \Big(\bigwedge_{1 \leq i < j \leq r} v_i \neq v_j \to \exists v_{r+1} \Big(\bigwedge_{1 \leq i \leq r} v_i \neq v_{r+1} \wedge \bigwedge_{\varphi \in \Phi} \varphi \wedge \bigwedge_{\varphi \in \Delta_{r+1} - \Phi} \neg \varphi \Big) \Big)$$

où Φ est un sous-ensemble de Δ_{r+1} qui est l'ensemble des formules de la forme $R\overline{x}$ ($R \in \tau$) telles que leurs variables libres sont parmi $v_1, \ldots, v_{r+1}$ et contiennent au moins v_{r+1}. On note T_{rand}, l'ensemble de tous les axiomes d'extension.

On montre que tout axiome d'extension est vrai dans presque toutes les structures finies et que T_{rand} a un modèle dénombrable unique $\mathcal{R}$ (à isomorphisme près).

On introduit alors les probabilités asymptotiques conditionnelles.

Définition 10. Pour K et H, deux classes de structures (finies) sur un vocabulaire τ, on appelle $\mu(K \mid H)$ la *probabilité asymptotique* de K en sachant H, c'est-à-dire la limite quand $n \to \infty$ de la fraction des structures **dans** $\boldsymbol{H}$ et dont le domaine est $\{1, \ldots, n\}$, qui sont dans K.

On donne une définition similaire pour la probabilité asymptotique isomorphe de K en sachant H. On dit qu'un énoncé du premier ordre φ est *paramétrique* s'il est la conjonction d'énoncés de la forme $\forall x_1 \dots \forall x_s((\bigwedge_{1\leq i\leq s-1} \neg x_i = x_{i+1}) \to \psi)$, où $s \geq 1$ et ψ est une combinaison booléenne de formules de la forme $Ry_1 \dots y_t$ avec $R \in \tau$ et $\{y_1, \dots, y_t\} = \{x_1, \dots, x_s\}$. On dit alors qu'une classe K de structures est *paramétrique* si K est égal à l'ensemble des modèles finis d'un énoncé paramétrique. Cet énoncé est qualifié de *non trivial* s'il a un modèle de cardinalité supérieure au maximum des arités des symboles de relation dans τ. On montre facilement qu'un énoncé paramétrique non trivial a des modèles arbitrairement grands.

On dit alors qu'un $r+1$-axiome d'extension (avec $\Phi \subseteq \Delta_{r+1}$) est *compatible* avec un énoncé paramétrique non trivial φ_0 si

$$\{\varphi_0\} \cup \{\exists v_1 \dots \exists v_{r+1}(\bigwedge_{1\leq i<j\leq r+1} \neg v_i = v_j \wedge \bigwedge_{\varphi\in\Phi} \varphi \wedge \bigwedge_{\varphi\in\Delta_{r+1}-\Phi} \neg\varphi)\}$$

est satisfiable. On appelle alors $T_{\text{rand}}(\varphi_0)$, l'union de $\{\varphi_0\}$ et l'ensemble des axiomes d'extension compatibles avec φ_0. On montre comme précédemment que $T_{\text{rand}}(\varphi_0)$ a un modèle dénombrable unique $\mathcal{R}(\varphi_0)$ (à un isomorphisme près). On montre également que pour tout axiome d'extension ψ compatible avec φ_0, $\mu(\psi \mid \varphi_0) = 1$. Enfin, on dit que H *satisfait la loi* 0-1 *pour* Ψ si pour tout $\psi \in \Psi$, $\mu(\psi \mid H) = 1$ ou $\mu(\psi \mid H) = 0$.

On montre le théorème suivant (voir [4]) :

Théorème 8.5. Pour un vocabulaire relationnel,

(1) FO et $\mathrm{L}^{\omega}_{\infty\omega}$ satisfont la loi 0-1;
(2) si H est une classe paramétrique non triviale, alors H satisfait la loi 0-1 pour $\mathrm{L}^{\omega}_{\infty\omega}$ et donc aussi pour FO;
(3) si H est une classe paramétrique non triviale, alors H satisfait la loi isomorphe 0-1 pour $\mathrm{L}^{\omega}_{\infty\omega}$ et donc aussi pour FO.

Pour faire la différence entre les probabilités asymptotiques et les probabilités asymptotiques isomorphes, on introduit la classe RIG de structures *rigides* (une structure est rigide si l'identité sur son domaine est le seul automorphisme $\mathcal{A}$). On montre alors le théorème suivant :

Théorème 8.6. Si H est une classe paramétrique non triviale telle que pour un $m \geq 2$, il existe une relation R r-aire et une surjection $f : \{1, \dots, r\} \to \{1, \dots, m\}$ telles que $\varphi_0 \wedge \exists x_1 \dots \exists x_m(Rx_{i(1)} \dots x_{i(r)} \wedge \bigwedge_{1\leq k<l\leq m} \neg x_k = x_l)$ et $\varphi_0 \wedge \exists x_1 \dots \exists x_m(\neg Rx_{i(1)} \dots x_{i(r)} \wedge \bigwedge_{1\leq k<l\leq m} \neg x_k = x_l)$ soient satisfiables, alors presque toutes les structures dans H sont rigides.

On montre alors grâce au théorème précédent, que presque tous les graphes sont rigides. Cela nous permet de donner un résultat de non-définissabilité grâce au théorème suivant :

Théorème 8.7. Soit φ_0 un énoncé paramétrique non trivial et φ un énoncé $\Sigma^1_1(\exists^*\forall^*)$. Si $\mathcal{R}(\varphi_0) \models \varphi$, alors il existe un énoncé du premier ordre ψ tel que $\widetilde{\mu}(\psi \mid \varphi_0) = 1$ et $\models_{\text{fin}} \psi \to \varphi$.

En appliquant le théorème précédent, si on prend pour φ_0 un énoncé axiomatisant la classe des graphes (qui soit, bien entendu, paramétrique et non trivial) et si on suppose que le fait d'être non rigide est exprimable par un énoncé $\Sigma_1^1(\exists^*\forall^*)$, alors presque tous les graphes seraient non rigides. D'où la contradiction. On a donc que la classe des graphes non rigides n'est pas axiomatisable par un énoncé $\Sigma_1^1(\exists^*\forall^*)$.

Remarque. On remarque que ce résultat ne nous dit pas que la non rigidité n'est pas dans (F)$\Sigma_1^1\forall$, car on se place ici dans la classe des graphes. Il faudrait se placer dans la classe des ordres.

On montre cependant que la classe des ordres n'est pas paramétrique. On situe, une fois de plus, la difficulté pour montrer une borne inférieure en complexité.

Théorème 8.8. La classe des ordres n'est pas paramétrique.

On montre ce théorème en constatant que les probabilités $\mu(\varphi|\mathrm{ORD})$ et $\widetilde{\mu}(\varphi|\mathrm{ORD})$ n'existent pas pour un énonce φ de $\mathrm{L}^2_{\infty\omega}$ exprimant que l'ordre a un nombre pair d'éléments (on rappelle que la classe des ordres (finis) de cardinalité paire n'est pas axiomatisable dans la logique du premier ordre).

Un espoir de montrer des résultats de non définissabilité pour notre logique réside dans le fait que Kolaitis et Vardi ont montré dans [24] que $\Sigma_1^1\exists^*\forall\exists^*$ a une loi 0-1.

9. Conclusion

Comme on peut le voir, notre résultat principal généralise les résultats de Lynch et Grandjean et permet de caractériser les sous-classes de complexité classiques par une logique naturelle et néanmoins restreinte. Le fait que l'on ait une loi 0-1 sur cette logique renforce l'idée que cette logique est "naturelle" et correspond bien aux sous-classes de complexité étudiées; on rappelle que pour les logiques naturelles mais peu restreintes, utilisées par Lynch et Grandjean pour leurs caractérisations, telles que la logique du second ordre monadique, il n'y a pas de loi 0-1.

D'autre part, en regardant la description par la théorie des modèles que nous avons, il semble difficile de trouver une logique plus expressive qui pourrait encore définir les problèmes dans NTIME(n) à part l'extension de notre logique au second ordre monadique et non plus avec des fonctions unaires. On pourrait également tenter de simplifier la logique d'arrivée en se restreignant à des fonctions de degré d et non plus $2d$.

Ceci motive la question suivante : Est-il vraiment nécessaire d'utiliser des fonctions et non pas des prédicats dans notre caractérisation de NTIME(n^d) sachant que nous désirons *capturer* (exactement) la classe de complexité ? La capture non-exacte peut toutefois être utilisée afin de prouver qu'un certain problème n'est pas dans la classe de complexité parce qu'il ne peut pas être exprimé dans la logique qui la contient.

Cette caractérisation nous permet de donner de nouvelles techniques de la théorie des modèles pour prouver des bornes inférieures sur la complexité de problèmes pour lesquels on ne sait pas s'ils sont calculables en temps polynomial ou exponentiel. De plus, elle permet de donner une autre formulation de la fameuse question P=NP

puisque c'est équivalent à ce que la classe des problèmes reconnus en temps linaire soit incluse dans P.

Une autre direction pourrait être de comprendre à quelles classes de complexité correspondent $\Sigma_1^{1\leq k}$; ceci pourrait certainement nous donner une caractérisation de classes similaires à DTIME(n^d). Il est intéressant de constater qu'avec les classes de formules de $\Sigma_1^{1\leq k}$ pour tout k, on a Σ_1^1, ce qui correspond à NP. Il serait donc intéressant de comprendre pourquoi on atteint P à partir de ces logiques et non pas NP. De manière générale, il reste beaucoup à faire pour comprendre ce que signifie en logique les différentes notions de complexité telles que les reductions, les oracles, les classes de comptage ...

Références

1. S. Abiteboul, M.Y. Vardi, and V. Vianu, *Fixpoint logics, relational machines, and computational complexity*, Proceedings of the 7th IEEE Symposium on Logic in Computer Science, 1992, pp. 156–168.
2. C.C. Chang and H.J. Keisler, *Model theory*, Studies in Logic and the Foundations of Mathematics, vol. 73, North-Holland, Amsterdam, 1973.
3. K.J. Compton, *An algebra and a logic for* NC^1, Proceedings of the 3rd IEEE Symposium on Logic in Computer Science, 1988, pp. 12–21.
4. H.-D. Ebbinghaus and J. Flum, *Finite model theory*, Springer-Verlag, Berlin, 1995.
5. R. Fagin, *Generalized first-order spectra and polynomial-time recognizable sets*, Complexity of Computation, SIAM-AMS Proceedings, vol. 7, 1974, pp. 43–73.
6. ———, *Finite-model theory – a personal perspective*, Theoretical Computer Science **116** (1993), 3–31.
7. M. Garey and D.S. Johnson, *Computers and intractability: a guide to the theory of* NP*-completeness*, Freeman, 1979.
8. A. Goerdt, *Characterizing complexity classes by higher-type primitive-recursive definitions*, Proceedings of the 4th IEEE Symposium on Logic in Computer Science, 1989, pp. 364–374.
9. E. Grandjean, *The spectra of first-order sentence and computational complexity*, SIAM Journal on Computing **13** (1984), 356–373.
10. ———, *Universal quantifiers and time complexity of random access machines*, Mathematical Systems Theory **18** (1985), 171–187.
11. ———, *A nontrivial lower bound for an* NP *problem on automata*, SIAM Journal on Computing **19** (1990), 438–451.
12. E. Grandjean and F. Olive, *Monadic logical definability of nondeterministic linear time*, Preprint.
13. Y. Gurevich, *Algebras of feasible functions*, Proceedings of the 24th IEEE Symposium on Foundations of Computer Science, 1983, pp. 210–214.
14. ———, *Toward logic tailored for computational complexity*, Computation and Proof Theory (M.M. Richter et al., ed.), Lecture Notes in Mathematics, vol. 1104, Springer-Verlag, 1984, pp. 175–216.
15. ———, *Logic and the challenge of computer science*, Current trends in theoretical computer science (E. Börger, ed.), Computer Science Press, 1988, pp. 1–57.
16. Y. Gurevich and S. Shelah, *Fixed-point extensions of first-order logic*, Annals of Pure and Applied Logic **32** (1986), 265–280.

17. D. Harel and D. Peleg, *Static logics, dynamic logics, and complexity classes*, Information and Control (1984), 86–102.
18. W. Hodges, *A shorter model theory*, Cambridge University Press, 1997.
19. N. Immerman, *Relational queries computable in polynomial time*, Information and Control **68** (1986), 86–104.
20. ———, *Languages that capture complexity classes*, SIAM Journal on Computing **16** (1987), no. 4, 760–778.
21. ———, *Nondeterministic space is closed under complement*, SIAM Journal on Computing (1988), 935–938.
22. ———, *Descriptive and computational complexity*, Computational Complexity Theory (Providence, RI) (J. Hartmanis, ed.), Proceedings of Symposia in Applied Mathematics, vol. 38, American Mathematical Society, 1989, pp. 75–91.
23. T. Jech, *Set theory*, Academic Press, New York, 1978.
24. P.G. Kolaitis and M.Y. Vardi, 0-1 *laws and decision problems for fragments of second-order logic*, Information and Computation **87** (1990), 302–338.
25. K.J. Lange, B. Jenner, and B. Kirsig, *The logarithmic hierarchy collapses:* $A\Sigma_2^L = A\Pi_2^L$, Proceedings of the 14th International Conference on Automata, Languages, and Programming, 1987.
26. D. Leivant, *Descriptive characterization of computational complexity*, Journal of Computer and System Sciences **39** (1989), 51–83.
27. A.B. Livchak, *The relational model for systems of automatic testing*, Automatic Documentation and Mathematical Linguistics **4** (1982), 17–19.
28. ———, *The relational model for process control*, Automatic Documentation and Mathematical Linguistics **4** (1983), 27–29.
29. J.F. Lynch, *Complexity classes and theories of finite models*, Mathematical Systems Theory **15** (1982), 127–144.
30. ———, *The quantifier structure of sentences that characterize nondeterministic time complexity*, Computational Complexity **2** (1992), 40–66.
31. W.J. Paul, N. Pippenger, E. Szemeredi, and W.T. Trotter, *On determinism versus nondeterminism and related problems*, Proceedings of the 24th IEEE Symposium on Foundations of Computer Science, 1983, pp. 429–438.
32. B. Poizat, *Cours de théorie des modèles*, OFFILIB, 1985.
33. V.Yu. Sazonov, *A logical approach to the problem of* P=NP, Proceedings of the 9th Conference on Mathematical Foundations of Computer Science, Lecture Notes in Computer Science, vol. 88, Springer-Verlag, 1980, pp. 561–575.
34. ———, *Polynomial computability and recursivity in finite domains*, Elektronische Informationverarbeitung und Kybernetik **16** (1980), 319–323.
35. L.J. Stockmeyer, *The polynomial time hierarchy*, Theoretical Computer Science **3** (1977), 1–22.
36. J. Tiuryn and P. Urzyczyn, *Some relationships between logic of programs and complexity theory*, Theoretical Computer Science **60** (1988), 83–108.
37. M.Y. Vardi, *The complexity of relational query languages*, Proceedings of the 14th Annual ACM Symposium on Theory of Computing, 1982, pp. 137–146.

Laboratoire de l'Informatique du Parallélisme, Ecole Normale Supérieure de Lyon, 46, allée d'Italie, 69364 Lyon Cedex 07, France
E-mail address: Gregory.Lafitte@ens-lyon.fr