De la complexité des problèmes de contraintes

Florent R. Madelaine

Université d'Auvergne * LIMOS †

Florent.Madelaine@u-Clermont1.fr

Résumé

La *conjecture de la dichotomie* postule qu'un problème de satisfaction de contraintes est soit facile (dans P), soit difficile (NP-complet). Cette conjecture, motivée par Feder et Vardi il y a 20 ans, reste ouverte malgré les efforts importants d'une communauté internationale regroupant des chercheurs issus d'horizons très variés. Nous présentons ici un bref aperçu des résultats obtenus et des techniques utilisées pour étudier cette question centrale en informatique théorique afin d'illustrer cette interdisciplinarité qui mêle algèbre, combinatoire, complexité et logique.

Abstract

The *dichotomy conjecture* asserts that a constraint satisfaction problem is either tractable (in P) or hard (NP-complete). This conjecture, motivated by Feder and Vardi 20 years ago, remains open, despite important efforts from a growing international community that regroups researchers from diverse horizons. We present here a brief overview in French of results and techniques used in attempts to settle this conjecture, in order to illustrate this multidisciplinarity which combines Algebra, Combinatorics, Complexity and Logic and makes this a central question in Theoretical Computer Science.

1 Introduction

Il est bien connu que les problèmes de contraintes sont très généraux et permettent de modéliser naturellement de nombreux problèmes combinatoires et industriels difficiles. La généralité qu'offre ce cadre de travail et l'existence de méthodes génériques comme la propagation de contraintes est la motivation première de la communauté IA qui s'attache à développer des solveurs efficients tant au niveau de la modélisation que du calcul d'une solution. Ce qui est peut-être moins connu, et qui indique la robustesse des problèmes de contraintes, est leur ubiquité : ils sont étudiés sous d'autres noms en *bases de données* (inclusion de requêtes conjonctives), en *combinatoire et théorie des graphes* (problème d'existence d'homomorphisme, coloriage de graphes) et en *logique* (évaluation de formules primitives positives). La complexité de ce(s) problème(s), en particulier les cas pour lesquels il existe un algorithme polynomial [1] semblent s'expliquer soit de manière combinatoire, soit ce qui est peut-être plus surprenant par des *propriétés algébriques*.

La motivation ici n'est pas de présenter les détails techniques mais de citer certain résultats théoriques importants, d'en expliquer l'intuition et de les remettre dans leur contexte en espérant que le lecteur se tournera vers des *survey* récents en anglais (le livre [14] en regroupe plusieurs). Nous commencerons en §2 par des exemples et par deux théorèmes de dichotomie historiquement importants, celui de Schaefer qui concerne le cas Booléen et celui de Hell et Nešetřil qui concerne les graphes non-orientés. Ces théorèmes ont conduit Feder et Vardi à avancer la conjecture de la dichotomie.

Au delà de l'intérêt évident d'une caractérisation des cas polynomiaux, cette conjecture est motivée par des résultats de *complexité structurelle* puisque la classe des problèmes de contraintes serait « la plus large » pour laquelle on observerait ce phénomène de dichotomie. On abordera cet aspect en §3.

Deux approches fondamentalement différentes permettent de restreindre le problème pour obtenir des classes polynomiales. Dans le premier cas, la notion de décomposition arborescente bornée du graphe des contraintes, bien connue depuis les travaux fondateurs de Freuder [21] est centrale et on verra en §4 qu'il existe un algorithme polynomial même lorsqu'il n'existe pas directement de telle décomposition. Dans le second

*Clermont Université, Université d'Auvergne, LIMOS, BP 10448, F-63000 Clermont-Ferrand.

†CNRS, UMR 6158, Laboratoire d'Informatique, de Modélisation et d'Optimisation des Systèmes, F-63173 Aubière.

1. pour le problème de décision (calculer, compter ou énumérer les solutions sont aussi des questions importantes mais on ne les abordera pas ici par manque de place).

cas, on restreint le langage de contraintes, et on verra en §5 que l'algèbre joue un rôle prépondérant et permet de caractériser par exemple le fait que pouvoir établir un certain niveau de cohérence suffit pour pouvoir déterminer si une solution existe. La *conjecture de la dichotomie* concerne en fait ce second cas et reste ouverte contrairement au premier cas qui est essentiellement complètement classifié. Finalement, nous nous tournerons en §6 vers des variantes, comme les problèmes de contraintes quantifiées, pour lesquels la méthode algébrique peut être adaptée avec succès.

2 Préliminaires

Une instance du *problème de satisfaction de contraintes* est traditionnellement définie comme un triplet $(\mathsf{Var}, \mathsf{Dom}, \mathcal{C})$, où Var est un ensemble de variables, Dom est un ensemble fini de valeurs et $\mathcal{C}$ est un ensemble de contraintes. Chaque contrainte est de la forme $(v_{i_1}, \ldots, v_{i_r}, R)$, où r est l'arité de la contrainte et $R \subseteq \mathsf{Dom}^r$ une relation spécifiant tous les r-uplets de valeurs admissibles simultanément par les variables $v_{i_1}, \ldots, v_{i_r}$. Une *solution* de cette instance est une application qui associe une valeur à chaque variable de sorte que chaque contrainte soit satisfaite, à savoir qu'à $v_{i_1}, \ldots, v_{i_r}$ correspond un r-uplet admissible.

Exemple. Le problème de satisfiabilité propositionelle (Sat) peut facilement se coder comme un problème de contraintes. On parle alors de *Generalized Satisfiability* puisque les entrées des deux problèmes sont légèrement différentes. En effet, on aura un domaine booléen $\mathsf{Dom} = \{0, 1\}$, et à la clause $x \vee y \vee \bar{z}$ de Sat on fait correspondre la contrainte ternaire portant sur les variables x, y, z et de relation $\{0, 1\}^3 \setminus \{(0, 0, 1)\}$.

L'exemple précédent montre que le problème de contraintes est NP-complet. La généralité de ce cadre de travail n'est donc pas surprenante puisque si par *modéliser par des contraintes* on entend *réduction polynomiale au problème de satisfaction de contraintes*, tout problème Ω de NP est modélisable en ce sens.

Remarque. Mais est-ce-vraiment la bonne notion de modélisation ? Probablement pas puisque si le problème de départ Ω était facile on voudrait le réduire à une restriction Ω' du problème de contraintes qui soit aussi facile. Nous aborderons plus en détail cette aspect en §3.

On reformule souvent le problème de contraintes en tant que *problème d'homomorphisme* puisqu'on peut naturellement séparer une instance en deux structures similaires – la première $\mathcal{A}$ de domaine Var correspond essentiellement au graphe des contraintes, la seconde $\mathcal{B}$ de domaine Dom regroupe les relations listant les r-uplets admissibles – et qu'une solution correspond exactement à un homomorphisme . Un *homomorphisme* de graphe est une application de sommets à sommets qui doit nécessairement envoyer une arête sur une arête. Le cas des graphes orientés est similaire sauf qu'on préservera aussi l'orientation d'un arc. Nous allons expliciter cette reformulation sur un exemple simple ci-dessous. Un exemple plus complexe avec plusieurs contraintes et donc des structures avec plusieurs relations est donné pour le cas de 2-Sat page suivante.

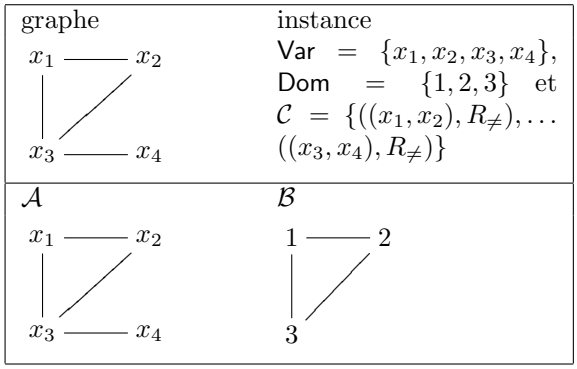

FIGURE 1 – Reformulation de la 3-colorabilité comme problème d'homomorphisme

Exemple. On peut facilement coder le problème de 3-colorabilité d'un graphe $G := (V, E)$ comme problème de contraintes. Les sommets sont les variables, soit $\mathsf{Var} := V$, le domaine correspond aux trois couleurs $\mathsf{Dom} := \{1, 2, 3\}$ et puisque les sommets incidents à une arête doivent prendre une couleur différente, on poste la contrainte $(x_i, x_j, R_{\neq})$ pour chaque arête (x_i, x_j) dans E, où $R_{\neq}$ est la relation binaire $\{1, 2, 3\}^2 \setminus \{(1, 1), (2, 2), (3, 3)\}$ listant les paires de couleurs admissibles.

Sous forme de problème d'homomorphisme, dans notre cas simplifié $\mathcal{A}$ et $\mathcal{B}$ seront des graphes. Pour un graphe G, on aura $\mathcal{A} := G$ et $\mathcal{B}$ qui sera un triangle K_3 (*cf.* figure 1). Ce problème est NP-complet alors que la 2-colorabilité est polynomiale.

En combinatoire, de nombreux problèmes difficiles deviennent faciles lorsque l'instance est un arbre. C'est le cas pour n'importe quel problème exprimable en logique monadique du second ordre, d'après le célèbre méta-théorème de Courcelle [12]. On verra en § 3 que notre problème s'exprime dans cette logique.

Théorème 1. *Si on se place dans le cas des graphes et qu'on restreint $\mathcal{A}$ comme étant un arbre, le problème d'homorphisme est polynomial.*

Remarque. On peut remplacer *graphe* par *structure* et *arbre* par *structure arborescente bornée*. Le résultat de Courcelle est très général et reste trop théorique

avec des constantes impratiquables (des tours d'exponentielle liées à la construction d'automates d'arbres). Par contre, pour le cas restreint qui nous intéresse, il existe un algorithme pratique dans l'esprit des travaux de Freuder [21] : on résoud localement en progressant du bas vers le haut dans l'arbre. Il s'agit d'un algorithme de type *bucket elimination* [17].

Le problème du H-*coloring* généralise le problème de coloriage de graphe. Il s'agit de savoir étant donné un graphe G si il existe un homomorphisme de G dans H (le k-coloriage correspond au choix d'une k-clique pour H) [2]. Si le graphe H est biparti, alors ce graphe H est 2-colorable. C'est-à-dire qu'il existe un homomorphisme de H vers K_2 (le graphe qui consiste en une seule arête). Si le graphe H contient au moins une arête, alors il existe un homomorphisme qui envoie l'arête de K_2 sur cette arête de H. Ainsi H et K_2 sont dit *homomorphiquement équivalent* puisque tout graphe G qui admet un homomorphisme vers l'un en admet un vers l'autre (car deux homomorphismes se composent naturellement pour donner un homomorphisme). Notez de plus que K_2 est le plus petit sous-graphe de H qui lui est homomorphiquement équivalent. On dira dans ce cas que K_2 est le *core* [3] de H. Du point de vue du problème de décision, le H-coloring et le 2-coloriage sont exactement le même problème, lorsque H est biparti et contient au moins une arête. Le problème du H-coloring est donc polynomial si H est biparti. Une analyse combinatoire subtile permet de montrer qu'il s'agit là des seuls cas faciles. À partir des cas difficiles connus comme les cliques de taille 3 ou plus, la preuve procède par étapes successives afin d'augmenter cette classe progressivement. La preuve se termine puisqu'on impose tellement de conditions sur les graphes restants qu'on fini par montrer une contradiction. Il s'agit d'une preuve qui relève de la théorie des graphe extrémaux.

Théorème 2 (Hell et Nešetřil 1990 [24])**.** *Le H-coloring est polynomial si H est biparti et NP-complet sinon.*

En général on considère des structures relationnelles quelconques et non pas juste des graphes. On rappelle qu'une *structure relationnelle* $\mathcal{B}$ consiste en un ensemble B (toujours fini dans cette note) et une liste finie de relations sur B. La notion d'homomorphisme entre deux structures similaires généralise naturellement le cas des graphes : il s'agit d'une fonction qui envoie chaque tuple de chaque relation de la première structure sur un tuple de la relation correspondante dans la seconde structure.

2. Ce H correspond à notre $\mathcal{B}$ partout ailleurs, nous gardons le H ici pour préserver la nomenclature utilisée en combinatoire.

3. En anglais, le terme est *core*.

Exemples. On peu coder 2-Sat comme suit. Soit $R^{\mathcal{B}}_{ab} = \{0,1\}^2 \setminus \{(a,b)\}$ et $\mathcal{B} = (\{0,1\}; R^{\mathcal{B}}_{00}, R^{\mathcal{B}}_{01}, R^{\mathcal{B}}_{11})$. Une instance $F = (\neg x \vee \neg z) \wedge (x \vee y) \wedge (y \vee \neg z) \wedge (u \vee x) \wedge (x \vee \neg u)$ correspond à une structure $\mathcal{A}$ de domaine $\{x, y, z, u\}$ et de relations $R^{\mathcal{A}}_{00} = \{(x,y),(u,x)\}$, $R^{\mathcal{A}}_{01} = \{(y,z),(x,u),\}$ et $R^{\mathcal{A}}_{11} = \{(x,z)\}$. Notez la correspondance naturelle entre homomorphisme de $\mathcal{A}$ à $\mathcal{B}$ et assignation valide de F. Ce problème est polynomial (NL-complet pour être précis).

De même, on peut coder *Horn 3-Sat* comme suit. On dispose d'une relation ternaire pour les clauses de Horn non triviales et de deux relations unaires pour les clauses unitaires. On pose $\mathcal{B} = (\{0,1\}; R; \{0\}, \{1\}\}$ où $R = \{(x,y,z) \mid y \wedge z \rightarrow x\}$. Ce problème est P-complet. Dans ce cas, il est bien connu que les relations de $\mathcal{B}$ sont préservées par l'opération booléenne $\wedge$ et qu'il en est de même pour l'ensemble des solutions d'un problème de type Horn en général. Ceci est la clé de l'approche algébrique que nous expliciterons en §5. On dit que $\wedge$ préserve la relation $R = \{0,1\}^3 \setminus \{(0,1,1)\}$ puisque pour tout choix de 2 triplets de R, par exemple $(1,1,0)$ et $(1,0,1)$ en appliquant $\wedge$ à chaque coordonnée on obtient le triplet $(1,0,0)$ qui appartient lui aussi à R.

Pour 2-Sat, l'opération booléenne ternaire h qui retourne toujours la valeur majoritaire (par exemple $h(0,1,1) = 1$) joue un rôle similaire.

Remarque. Notez que $\mathcal{A}$ peut être une structure quelconque. Ainsi, les exemples ci-dessus restent polynomiaux même si $\mathcal{A}$ est une structure non arborescente analogue à une grille par exemple.

Cette vision des problèmes de contraintes via l'homomorphisme est assez populaire car elle permet de séparer de manière naturelle *deux approches fondamentalement différentes* qui permettent d'obtenir des restrictions polynomiales. Les restrictions sur $\mathcal{A}$ sont combinatoires (lien avec les décompositions arborescentes déjà évoquées) alors que les restrictions sur $\mathcal{B}$ sont régies par les propriétés algébriques de fonctions associées à $\mathcal{B}$, comme pour les deux exemples ci-dessus [4].

Un résultat historiquement très important, qui entre dans ce second cas correspond à la classification de *Generalized Satisfiability*. Nous en esquisserons une preuve en §5.

Théorème 3 (Schaefer 1978 [13])**.** *Si le domaine de $\mathcal{B}$ a deux éléments alors le problème d'homomorphisme est soit polynomial, soit NP-complet. Si les relations de $\mathcal{B}$ sont* 0-valide*,* 1-valide*,* 2-Sat*,* Horn*,* dual-Horn*, ou* affine *alors le problème devient polynomial, sinon il est NP-complet.*

4. Notons que même si c'est plutôt contre-intuitif, le théorème de Hell et Nešetřil s'explique lui aussi algébriquement [4].

À l'instar du problème du H-coloring, pour lequel H est fixé, on fixe fréquemment la structure $\mathcal{B}$ et seule la structure $\mathcal{A}$ est une entrée du problème. On dénote ce problème par $CSP(\mathcal{B})$ et on parle dans ce cas de problème de contrainte *non-uniforme*. On notera par la suite par CSP la classe formée par ces problèmes non-uniformes.

Remarque. Cette vision est quelque peu restrictive et ne permet pas de capturer les *contraintes globales*. Cependant, il est fréquent que les algorithmes « uniformisent ». Nous reviendrons sur cet aspect plus loin.

3 Conjecture de la dichotomie

Un corollaire du théorème de Ladner [27] est que si P est différent de NP, alors il existe des problèmes de complexité intermédiaire *qui ne sont ni dans P, ni NP-complet*. En d'autres termes, on ne devrait pas pouvoir prouver de théorème de dichotomie pour NP.

Dans un article qui a eu beaucoup d'influence [20], Feder et Vardi établissent que trois classes de complexité, qu'on ne définira pas formellement ici, $\mathcal{L}_1$, $\mathcal{L}_2$ et $\mathcal{L}_3$ sont *calculatoirement équivalentes à NP*, c'est-à-dire que pour tout problème Ω de NP, il existe un représentant Ω' dans la classe $\mathcal{L}_i$ telle que Ω se réduit à Ω' en temps polynomial et inversement Ω' se réduit à Ω. Ainsi, structurellement ces trois classes se comportent exactement comme NP et en particulier, par le théorème de Ladner, elles ne peuvent pas avoir de dichotomie (on suppose dorénavant que P est différent de NP).

Ces trois classes de complexité se définissent de manière logique et le fragment immédiatement en dessous de ces trois classes correspond à des problèmes exprimables dans la logique MMSNP, qui est un fragment de la logique monadique du second ordre chère à Courcelle [12]. Cette logique permet d'exprimer des problème combinatoires de la forme suivante : il existe un coloriage de l'entrée qui interdit localement un nombre fini d'obstructions coloriées. En particulier on peut exprimer les problèmes $CSP(\mathcal{B})$, pour tout $\mathcal{B}$.

Exemple. Pour la 3 colorabilité, les obstructions seront simplement les arêtes rouge-rouge, vert-vert et bleue-bleue. La formule de MMSNP sera de la forme suivante : $\exists$ une partition R, V, B des sommets telle que $\forall$ sommets x, y $\neg\big(E(x,y) \wedge R(x) \wedge R(y)\big) \wedge \neg\big(E(x,y) \wedge V(x) \wedge V(y)\big) \wedge \neg\big(E(x,y) \wedge B(x) \wedge B(y)\big)$.

La logique MMSNP ne correspond pas exactement à CSP, car elle est trop générale, mais elle a de nombreux points communs avec CSP.

Théorème 4 (Feder et Vardi 1993 [20], Kun [5])**.**

1. *Tout CSP s'exprime par une formule de MMSNP, mais il existe des formules de MMSNP qui n'expriment pas un problème de CSP.*
2. *Par contre, MMSNP est calculatoirement équivalente à CSP.*

Remarques. La logique MMSNP permet d'exprimer un problème comme « pas de triangle » qui n'est pas un problème de contraintes non-uniforme. Il s'agit en fait d'un problème de contrainte non-uniforme *de domaine infini* au sens de Bodirsky [1]. On peut déterminer de manière effective lorsqu'un problème de MMSNP est un CSP à domaine fini ou bien à domaine infini [32]. Pour MMSNP, l'inclusion de 2 problèmes est décidable mais plus complexe que pour CSP : pour CSP, l'inclusion est un CSP uniforme (NP-complet) alors que pour MMSNP on est au moins au second niveau de la hiérarchie polynomiale [29].

Les résultats de dichotomie historiques de Schaefer, et Hell et Nešetřil et des considérations techniques sur la nature foncièrement différente d'une classe $\mathcal{L}_i$ et de MMSNP ont conduit Feder et Vardi à avancer la conjecture suivante en 1993.

Conjecture de la dichotomie. *Tout problème de contrainte non-uniforme est soit polynomial soit NP-complet.*

Notons que cette conjecture implique que MMSNP a une dichotomie puisque MMSNP est calculatoirement équivalent à CSP. L'inverse est trivialement vrai puisque CSP est une restriction de MMSNP. Puisque, les classes de complexité $\mathcal{L}_i$ « immédiatement au dessus » de MMSNP contiennent des problèmes de complexité intermédiaire et que MMSNP est calculatoirement équivalent à CSP, on peut donc voir CSP comme la plus grande classe qui devrait avoir une dichotomie.

De manière générale, on dira qu'une classe de complexité $\mathcal{C}$ est *dichotomie-complète* si $\mathcal{C}$ est calculatoirement équivalent à CSP, puisque démontrer que $\mathcal{C}$ a une dichotomie impliquerait la conjecture de la dichotomie.

Exemples. Le problème d'homomorphisme pour les graphes orientés est dichotomie-complet [20]. Le problème d'homomorphisme pour les algèbres ayant deux fonctions unaires est dichotomie-complet [19].

Lorsqu'on s'intéresse au cas spécial où la formule MMSNP est du premier ordre, c'est-à-dire que les obstructions ne sont pas coloriées comme dans le cas de l'exemple « pas de triangle », le cas où cette formule

5. L'une des réductions reposait sur un lemme dit de Erdös de nature aléatoire, mais ce lemme a été déterminisé par Kun.

exprime un CSP fini est étudié sous le nom de *paire duale* en combinatoire. Par exemple, si la formule n'a qu'une obstruction qui est un chemin orienté de longueur n, cela revient à avoir un homomorphisme dans un tournoi transitif à n sommets. L'étude des paires duales et de la dualité est une question importante en combinatoire, voir [7] pour un survey.

On peut aussi se poser la question si cette différence d'expressivité entre MMSNP et CSP s'estompe si on restreint la classe des structures considérées. Ceci est le cas lorsque les structures sont de degré borné, sont planaires, ou plus généralement définissables par mineurs interdits [6] : dans ces cas, une formule de MMSNP, qui définit en général un CSP de domaine infini, devient forcément un CSP de domaine fini [28].

4 Graphe des contraintes arborescent

Nous nous tournons dans cette section vers les restrictions dites structurelles, qui concernent uniquement la structure $\mathcal{A}$, la structure $\mathcal{B}$ étant quelconque. Il s'agit de restrictions qui se comportent particulièrement bien et qui impliquent l'existence d'un algorithme polynomial même lorsque $\mathcal{B}$ n'est plus fixe et participe aussi à l'entrée. On dénote ce problème dit uniforme par $CSP(\mathscr{C}, \_)$, pour le cas où l'entrée $\mathcal{A}$ appartient à une classe $\mathscr{C}$ de structure et $\mathcal{B}$ est quelconque.

Question. *Quelles sont les classes $\mathscr{C}$ pour lesquelles $CSP(\mathscr{C}, \_)$ est polynomial ?*

Nous avons évoqué qu'un problème de contrainte est facile à résoudre si l'entrée est un arbre. Si l'entrée n'est pas un arbre mais peut être décrite par une décomposition arborescente de largeur au plus k, où k est une constante, alors le problème peut être résolu en temps polynomial n^k. Par contre cet algorithme nécessite d'avoir la décomposition. Décider si une structure $\mathcal{A}$ a une telle largeur arborescente est NP-complet si k est une donnée du problème. Par contre si la largeur k est fixée, il existe un algorithme linéaire en la taille de $\mathcal{A}$ qui calcule, si c'est possible, une telle décomposition.

On travaille sur des structures et non des graphes. On se ramène à un graphe en considérant « le graphe des contraintes ». Ce graphe est connu sous le nom de graphe de Gaifman (de $\mathcal{A}$) en logique. Ce graphe a le même domaine que $\mathcal{A}$ et on a une arête entre deux sommets si ils participent à un même tuple d'une relation quelconque de $\mathcal{A}$. Ainsi, les sommets de $\mathcal{A}$ participant à un tuple formeront une clique dans ce graphe. La complexité de l'algorithme sera alors $n^{O(k)}$ où k est la largeur arborescente du graphe de Gaifman, le $O(k)$ cachant la complexité du codage d'une structure relationnelle en un graphe. On peut donc prendre pour $\mathscr{C}$ la classe $\mathscr{T}_k$ des structures $\mathcal{A}$ dont le graphe de Gaifman a largeur d'arborescence au plus k.

On peut se demander si cette restriction structurelle est la plus générale possible. En fait, on peut autoriser $\mathcal{A}$ a être non pas une structure de $\mathscr{T}_k$ mais *une structure homomorphiquement équivalente à une structure $\mathcal{A}'$ appartenant à $\mathscr{T}_k$*. On notera cette nouvelle classe par $\mathscr{H}\mathscr{T}_k$.

Théorème 5 (Dalmau, Kolaitis, Vardi [16])**.** *Pour tout k fixé, le problème de contrainte uniforme $CSP(\mathscr{H}\mathscr{T}_k, \_)$ est polynomial.*

Exemple. On peut par exemple considérer l'ensemble $\mathscr{B}$ des graphes bipartis. Lorsque un graphe biparti a une arête, son cœur est K_2 et sinon un seul sommet c'est-à-dire K_1. Ces deux graphes sont des arbres, donc $\mathscr{B}$ est inclus dans $\mathscr{H}\mathscr{T}_1$. Notez que $\mathscr{B}$ contient les grilles qui sont le prototype même du graphe n'ayant pas une largeur arborescente bornée. $CSP(\mathscr{B}, \_)$ est polynomial pour une raison triviale : pour une entrée $(\mathcal{A}, \mathcal{B})$, il suffit de tester si $\mathcal{B}$ contient une arête ou si $\mathcal{A}$ n'en contient pas.

Pourquoi le problème $CSP(\mathscr{H}\mathscr{T}_k, \_)$ est-il polynomial ? Puisque $\mathcal{A}$ et $\mathcal{A}'$ sont homomorphiquement équivalents, ces deux structures seront ou bien simultanément acceptées ou bien simultanément rejetées du fait de leur équivalence (on rappelle que la composition de deux homomorphismes est un homomorphisme). Ainsi, il suffit étant donné $\mathcal{A}$ de calculer $\mathcal{A}'$, de calculer une décomposition de $\mathcal{A}'$ et d'utiliser l'algorithme polynomial reposant sur cette décomposition. Le problème avec cet argument c'est que calculer $\mathcal{A}'$ est une question difficile. En effet, étant donné $\mathcal{A}$, déterminer si un tel $\mathcal{A}'$ existe est NP-complet, même lorsque k est fixé. L'algorithme polynomial pour $CSP(\mathscr{H}\mathscr{T}_k, \_)$ ne nécessitera pas le calcul d'une décomposition : il repose sur un jeu associé à l'expressivité dans une logique, qui est un fragment de la logique du premier ordre n'ayant que $k+1$ variables.

Dans le cas des arbres, on a $k = 1$. La logique a seulement 2 variables et dans le jeu associé, chaque joueur a deux pions. Le jeux oppose le *saboteur* au *copieur* qui jouent sur deux structures $\mathcal{A}$ et $\mathcal{B}$: le premier essaye d'exhiber des différences entre $\mathcal{A}$ et $\mathcal{B}$ alors que le second essaye de montrer leur similarité. Le saboteur joue sur $\mathcal{A}$ et le copieur sur $\mathcal{B}$.

Au début, le saboteur pose deux pions p_1 et p_2 où il le veut sur deux sommets de la structure $\mathcal{A}$ (les som-

6. Un mineur d'un graphe G est un graphe H obtenu par contraction d'arête(s) et suppression de sommet(s) de G. Le célèbre théorème de Kuratowski caractérise les graphes planaires comme étant les graphes n'ayant ni K_5 ni $K_{3,3}$ comme mineur. La classe des graphes planaires est un exemple de classe définissable par mineurs interdits.

mets pouvant coïncider). Le copieur répond en posant deux pions q_1 et q_2 sur deux sommets de $\mathcal{B}$. Ensuite, à chaque tour suivant, le saboteur ramasse un ou plusieurs pions et les replace sur $\mathcal{A}$ où il le souhaite ; et, le copieur procède de même. Par exemple, si le saboteur déplace son pion p_2, le copieur doit jouer q_2.

Le saboteur gagne la partie si après un tour donné, la fonction qui envoie le sommet de $\mathcal{A}$ sur lequel est placé p_1 sur le sommet de $\mathcal{B}$ associé au premier pion du copieur q_1 et de même pour la paire de sommets correspondant à p_2 et q_2, n'est pas un homomorphisme partiel de $\mathcal{A}$ dans $\mathcal{B}$. Dans le cas des graphes, cela correspond au cas où les pions du saboteur sont adjacents sur $\mathcal{A}$ et ceux du copieur ne le sont pas sur $\mathcal{B}$. Le copieur a une stratégie gagnante si il peut jouer indéfiniment sans perdre.

Si il existe un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$, alors le copieur peut utiliser cet homomorphisme pour répondre à n'importe quel coup du saboteur et a donc une stratégie gagnante.

Si $\mathcal{A}$ est un chemin et que le copieur a une stratégie gagnante, on peut construire un homomorphisme h de $\mathcal{A}$ dans $\mathcal{B}$. On utilise les deux pions du saboteur pour « marcher » le long du chemin $\mathcal{A}$ en déplaçant alternativement le premier pion puis le second pion. On regarde la réponse du copieur. L'homomorphisme partiel existant à chaque étape construit progressivement l'homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$. L'argument est similaire si $\mathcal{A}$ est un arbre.

On a donc démontré que si $\mathcal{A}$ est un arbre, alors il y a un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$ si et seulement si le copieur a une stratégie gagnante. Il se trouve qu'on peut déterminer en temps polynomial si c'est le cas. On a donc pour l'instant une preuve alternative du théorème 1 utilisant la logique et les jeux. Si $\mathcal{A}$ n'est pas un arbre mais que $\mathcal{A}'$ est un arbre homomorphiquement équivalent à $\mathcal{A}$, on peut utiliser ces homomorphismes pour montrer que le copieur a une stratégie gagnante si et seulement si il y a un homomorphisme de $\mathcal{A}$ dans $\mathcal{B}$.

Le théorème précédent caractérise précisément les restrictions structurelles qui permettent d'obtenir un algorithme polynomial, puisque Grohe montre l'implication inverse.

Théorème 6 (Dalmau *et. al.*[16], Grohe [22])**.** *Sous une hypothèse de complexité paramétrée*[7]*, pour toute classe de structures $\mathscr{C}$ qui est récursivement énumérable, $CSP(\mathscr{C}, \_)$ est polynomial si et seulement si $\mathscr{C}$ est inclus dans $\mathscr{H}\mathscr{T}_k$, pour k fixé.*

À la lumière de ces résultats, on peut penser que la question des restrictions structurelles est complètement résolue. Ce n'est pas tout à fait le cas puisque le cadre de travail proposé a deux inconvénients. D'une part, il ne permet que de travailler dans le cas où l'arité d'une contrainte est bornée, ce qui est plutôt réducteur si on voit l'importance qu'ont prises les contraintes globales. D'autre part, la polynomialité de $CSP(\mathscr{H}\mathscr{T}_k, \_)$ est en quelque sorte inaccessible puisque le méta-problème qui consiste à décider si $\mathcal{A}$ appartient à $\mathscr{H}\mathscr{T}_k$ est NP-complet même si k est fixé.

Ce domaine d'étude reste donc particulièrement actif. Il y a de nombreux travaux portant sur la décomposition d'hypergraphes plutôt que de graphes afin de pouvoir appréhender le cas des contraintes globales. Ces travaux ont beaucoup de liens avec les bases de données puisque le problème de satisfaction de contraintes d'instance $(\mathcal{A}, \mathcal{B})$ correspond au problème d'évaluation d'une requête conjonctive $\varphi_{\mathcal{B}}$ sur une base de données $\mathcal{A}$. Pour plus de détails, voir par exemple [36].

La notion la plus générale dans le cas des hypergraphes est lié à la notion de *fractional hyper-tree width* proposée par Grohe et Marx [23]. Une classification complète par rapport à la complexité paramétrée a été donné par Marx [35].

5 Approche algébrique

Dans cette section, on considère que la structure $\mathcal{B}$ de domaine Dom et de relations Γ est fixée. On appelle Γ le *langage des contraintes*. On s'intéresse au pouvoir d'expressivité du langage de contrainte Γ. En particulier, on aimerait pouvoir comparer l'expressivité de différents langages de contraintes. Puisqu'on va être amené à changer Γ mais pas Dom, on notera $CSP(\Gamma)$ pour $CSP(\mathcal{B})$ dans la suite.

Exemple. Supposons que Γ contiennent deux relations binaires R_1 et R_2. Les deux contraintes $((x,z), R_1)$ et $((z,y), R_2)$ induisent une contrainte implicite $((x,y), R_3)$, où $R_3 = R_1 \circ R_2$. La relation R_3 n'est pas forcément dans Γ mais elle est *exprimable* par Γ. Il y a une réduction polynomiale de $CSP(\Gamma \cup \{R_3\})$ dans $CSP(\Gamma)$ puisqu'on peut remplacer chaque contrainte R_3 par un gadget composé de deux contraintes R_1 et R_2 et d'une variable additionnelle. Il existe une réduction triviale dans l'autre direction. Les deux problèmes $CSP(\Gamma)$ et $CSP(\Gamma \cup \{R_3\})$ sont donc polynomialement équivalents.

Il est important de noter que R_3 est l'interprétation de la formule $\exists z R_1(x,z) \wedge R_2(z,y)$.

On note par $\langle\Gamma\rangle$ l'ensemble des relations *exprimables* à partir de celles de Γ. Formellement, il s'agit de toutes les relations qu'on peut obtenir en interprétant des formules *primitives positives* sur Γ (il s'agit de formules contenant $\exists, \wedge, =$). Notez que $\langle\Gamma\rangle$ contient une infinité

7. $FPT \neq W[1]$: une conjecture analogue à $P \neq NP$ en complexité paramétrée [18].

de relations. Jusqu'ici nous ne considérions que des langages de contraintes finis. On dira qu'un langage infini est polynomial [8] ssi tous ses fragments finis le sont. Le théorème suivant nous permet de restreindre notre étude de la complexité aux langage de contraintes tels que $\Gamma = \langle\Gamma\rangle$.

Théorème 7 (Jeavons 1998 [25]). *Si Γ_1 et Γ_2 sont des langages de contraintes tels que $\langle\Gamma_1\rangle \subseteq \langle\Gamma_2\rangle$ alors $CSP(\Gamma_1)$ se réduit polynomialement [9] à $CSP(\Gamma_2)$.*

On peut considérer ces ensembles clos par $\langle\,.\,\rangle$, les *clones relationnels*, par rapport à l'inclusion et chercher parmi eux les langages Γ maximaux (respectivement minimaux) qui sont polynomiaux (respectivement NP-complets). Ceci revient à étudier la frontière entre cas faciles et difficiles dans un treillis connu sous le nom de *treillis des clones relationnels* en algèbre universelle. Dans le cas booléen, lorsque Dom a deux éléments, ce treillis est connu sous le nom de *treillis de Post* et les preuves modernes du théorème de Schaefer s'appuient très directement sur ce treillis. En fait le treillis de Post ne concerne pas des relations mais des fonctions. On passe de l'un à l'autre par la notion de *préservation* déjà entrevue et qu'on ne définira pas formellement ici.

Exemples. Si Γ est l'ensemble des relations correspondant à des clauses de Horn, alors toutes les relations de Γ sont préservées par l'opération booléenne $\wedge$. On dira dans ce cas que $\wedge$ est un *polymorphisme* de Γ.

Si Γ est l'ensemble des relations 0-valides, c'est-à-dire contenant un r-uplet avec seulement des 0, alors la fonction constante 0 est un polymorphisme de Γ.

On note par $\mathsf{Pol}(\Gamma)$ l'ensemble de tous les polymorphismes de Γ. Inversement, si F est un ensemble de fonctions [10], on note par $\mathsf{Inv}(F)$ l'ensemble des relations préservées (**inv**ariantes) par toutes les fonctions de F. Les opérateurs Pol et Inv établissent une correspondance de Galois entre deux treillis, l'un étant celui des clones relationnels, l'autre étant celui des *clones (fonctionnels)*. Puisque, plus il y a de relations à préserver, moins il y a de fonctions qui vont les préserver, et inversement, les deux treillis ont exactement la même structure, mais le haut de l'un correspond au bas de l'autre [11] (cf. Figure 2). En particulier, le langage de toutes les relations correspond au clone des fonctions triviales que sont les projections. Le clone

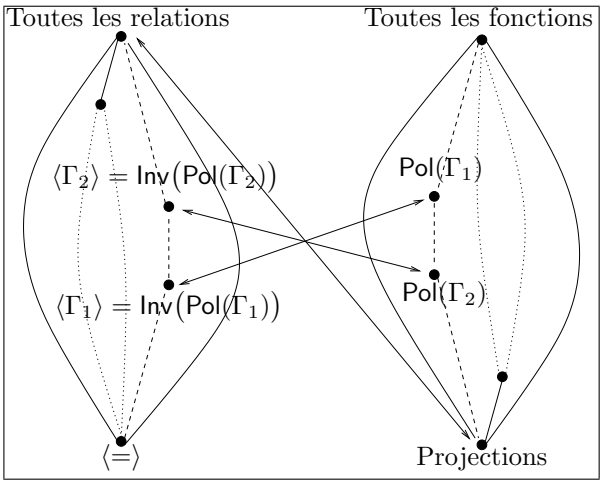

FIGURE 2 – Correspondance de Galois

de toutes les fonctions ne laisse *grosso modo* invariant que la relation $=$, et correspond au clone relationnel $\langle=\rangle$.

Intuitivement les polymorphismes contrôlent l'expressivité et donc la complexité. Comme les deux treillis sont structurellement les mêmes on va travailler sur le treillis des clones.

Les éléments du treillis de Post (cf. Figure 3) correspondent à des clones booléens, c'est-à-dire des ensemble de fonctions booléennes. Sur cette figure, on a tout en haut, toutes les fonctions booléennes (BF), tout en bas les projections (I_2). La classification de Schaefer (voir théorème 3) peut être vue comme un corollaire du théorème de Jeavons, de la description du treillis de Post et d'une étude de la complexité de 7 cas. Sur le treillis, ces 7 cas sont $I_0, I_1, V_2, E_2, L_2, D_2$ (qui correspondent aux 6 langages de contraintes polynomiaux et maximaux) et N_2 (qui est NP-complet et minimal).

Remarque. Ce qui est remarquable mais n'apparaît pas dans notre version aseptisée du théorème de Schaefer, c'est que chaque langage de contrainte maximalement polynomial est caractérisé en terme de préservation par une fonction booléenne. Ainsi, on peut disposer d'un algorithme général qui saura *identifier* les cas pour lesquels il existe un algorithme polynomial, et basculer s'il y a lieu en mode polynomial.

Lorsqu'on se penche sur les cas où le domaine a au moins trois valeurs, on peut de nouveau se ramener à l'étude du treillis des clones, mais contrairement à celui de Post, ces treillis sont grands (ils ne sont plus dénombrables), très complexes et largement inconnus par les algébristes. Jeavons et ses co-auteurs ont beaucoup travaillé à la généralisation des résultats de Schaefer. Ils démontrent par exemple précisément quand *établir la k-cohérence forte* permet de décider si

8. Pour être précis, il s'agit de *local tractability*. En pratique, des algorithmes vont « uniformiser » : par exemple, l'algorithme pour résoudre Horn-3-Sat est le même pour Horn-Sat. Ce second cas est dit *globally tractable*.

9. En fait, cette réduction est même dans logspace et préservera donc une analyse de complexité plus fine avec des classes comme L ou NL.

10. de Dom^r dans Dom, r étant une arité quelconque.

11. Pol et Inv sont des anti-isomorphismes de treillis.

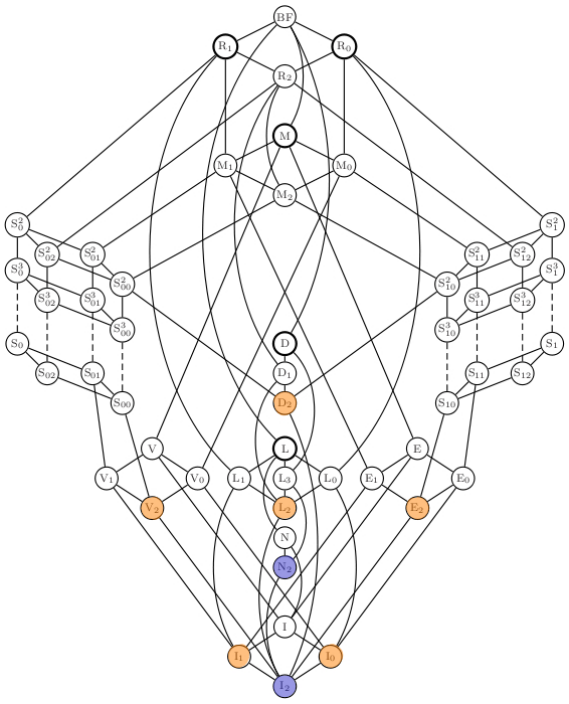

FIGURE 3 – Treillis de Post (1941).

une instance est satisfaisable [26] : à savoir, lorsque le langage des contraintes est préservé par une opération *near-unanimity* d'arité $k+1$.

Exemple. Le cas du langage 2-Sat déjà évoqué est un example pour $k=2$: il y a préservation par la fonction majorité ternaire. La 2-cohérence (*path consistency*) est un algorithme complet pour le problème de décision dans ce cas restreint.

Feder et Vardi étudient ce même cas en utilisant la notion de programme Datalog, qui est intrinsèquement exécutable en temps polynomial [20]. Datalog est un cadre plus riche puisqu'il permet aussi d'expliquer le cas d'Horn-Sat de nature différente. Un programme Datalog calcule récursivement des relations auxiliaires. Intuitivement, c'est une forme de propagation de contraintes. On dérive localement de nouvelles contraintes à l'aide des règles du programme (l'optique étant de dériver une contradiction).

Exemple. Le programme suivant décide si une instance de Horn-3-Sat est insatisfaisable.

$$\begin{aligned} \lambda(x) &\leftarrow 1(x) \\ \lambda(x) &\leftarrow \lambda(y), \lambda(z), R(x,y,z) \\ \gamma &\leftarrow \lambda(x), 0(x) \end{aligned}$$

Le prédicat constant γ est le *but du programme* : si ce dernier « est activé » alors l'instance est rejetée.

Malgré ces avancés importantes, les résultats sont longtemps restés parcellaires et l'analyse sur le treillis ne permettait pas d'obtenir de classification complète. Il a fallu attendre Bulatov pour une classification complète du cas où il y a seulement *trois valeurs* [5]. Ce résultat peut sembler très incrémental mais conceptuellement il contient des idées très importantes qui permettent de s'abstraire du treillis et de travailler directement sur des variétés [12]. En effet, un langage de contrainte correspond à une algèbre et localement cette algèbre ne peut prendre que 5 types (par exemple elle se comporte comme un treillis ou encore comme une algèbre booléenne) : l'analyse de ces types permet d'établir la complexité associée au langage de contrainte. Ce renouveau de l'approche algébrique a permis à Bulatov d'établir une classification complète, *pour n'importe quel domaine fini*, de la complexité des langages de contraintes dits *conservatifs* [6]. Il s'agit du cas où on peut associer à chaque variable un domaine, une hypothèse qui semble être prévalente en intelligence artificielle.

Ce second résultat très important confirme ce que Feder et Vardi avaient suggéré dans leur papier fondateur [20], à savoir la pauvreté relative de « l'arsenal algorithmique polynomial ». Il existe essentiellement deux algorithmes, pour résoudre un problème non-uniforme : l'un généralise la cohérence et correspond à l'existence d'un programme Datalog, et l'autre découle du fait qu'on peut dans certain cas représenter de manière compacte l'ensemble des solutions [13].

Il existe une reformulation algébrique de la conjecture de la dichotomie et de nombreux raffinements pour d'autres classes de complexité. Depuis quelques années, l'approche algébrique des problèmes de contraintes a contribué à redynamiser l'algèbre universelle et à apporter de nouvelles questions dans cette discipline. Après une longue série de travaux, on sait par exemple maintenant que l'assertion suivante implique la conjecture de la dichotomie.

Conjecture 8. *Si* $\mathsf{Pol}(\Gamma)$ *contient un* terme de Siggers *alors* Γ *est polynomial.*

Un terme de Siggers est une opération d'arité 4 satisfaisant les identités $f(x,x,x,x) = x$ et $f(y,x,y,z) = f(x,y,z,x)$.

Pour plus de détails sur la dichotomie de la conjecture et la nouvelle approche algébrique, voir [9].

12. au sens de Birkhoff.
13. un exemple typique est la résolution d'équation linéaires.

Fragment	Dual	Classification ?
$\{\exists, \vee\}$ $\{\exists, \vee, =\}$	$\{\forall, \wedge\}$ $\{\forall, \wedge, \neq\}$	logspace
$\{\exists, \wedge, \vee\}$ $\{\exists, \wedge, \vee, =\}$	$\{\forall, \wedge, \vee\}$ $\{\forall, \wedge, \vee, \neq\}$	logspace si toutes les boucles, NP-complet sinon
$\{\exists, \wedge\}$ $\{\exists, \wedge, =\}$	$\{\forall, \vee\}$ $\{\forall, \vee, \neq\}$	**Conjecture de la dichotomie** des CSP (Ptime ou NP-complet). Résolue dans le cas des graphes (Hell et Nešetřil), le cas booléen (Schaefer), le cas à trois éléments (Bulatov) et le cas conservatif (Bulatov). Il suffit de trouver un algorithme polynomial pour les graphes orientés admettant un terme de Siggers pour résoudre la conjecture.
$\{\exists, \wedge, \neq\}$	$\{\forall, \vee, =\}$	NP-complet si $\|A\| \geq 3$, se réduit aux théorème de Schaefer sinon.
$\{\exists, \forall, \wedge\}$ $\{\exists, \forall, \wedge, =\}$	$\{\exists, \forall, \vee\}$ $\{\exists, \forall, \vee, \neq\}$	Une dichotomie entre Ptime et Pspace-complet est démontrée dans le cas booléen (Schaefer, Creignou). **En général, il n'y a pas de conjecture précise**. Des résultats partiels montrent que les complexités Ptime, NP-complet, et Pspace-complet sont atteintes. Le cas des graphes non orienté reste ouvert.
$\{\exists, \forall, \wedge, \neq\}$	$\{\exists, \forall, \vee, =\}$	Pspace-complet pour $\|A\| \geq 3$, se réduit à la classification de Schaefer pour QSat sinon.
$\{\forall, \exists, \wedge, \vee\}$		Pour QCSP avec disjonction, nous démontrons une tetrachotomie non triviale entre logspace, NP-complet, co-NP-complet et Pspace-complet [30].
$\{\forall, \exists, \wedge, \vee, =\}$ $\{\forall, \exists, \wedge, \vee, \neq\}$ $\{\neg, \exists, \forall, \wedge, \vee, =\}$		Ptime pour $\|A\| \leq 1$, Pspace-complet sinon.
$\{\neg, \exists, \forall, \wedge, \vee\}$		Ptime quand $\mathcal{B}$ ne contient que des relations vides ou complète , Pspace-complet sinon

FIGURE 4 – Complexité de CSP, QCSP et du model checking en général (pour lire la complexité du fragment dual, remplacer NP par co-NP et *vice versa*).

6 Autres questions

L'approche algébrique permet d'attaquer d'autres questions de complexité que la conjecture de la dichotomie. En particulier, dans le cas booléen, Nadia Creignou *et. al.* montrent qu'on peut s'appuyer sur le treillis de Post pour caractériser la complexité de nombreux problèmes [15]. Parfois, on dispose d'un théorème similaire au théorème de Jeavons et on peut travailler *a priori* sur le treillis, c'est la cas pour la *circumscription* ou l'*abduction*. Dans d'autres cas, on peut s'appuyer sur le treillis pour réaliser l'étude mais on ne remarque qu'*a posteriori* que le résultat est compatible avec la structure du treillis, par exemple pour le *dénombrement* ou *l'énumération* des solutions de Sat. D'autres questions importantes ne sont pas compatibles avec la structure du treillis, c'est le cas par exemple de MaxSat, de la complexité paramétrée de Sat ou encore de son approximation.

Pour des problèmes qui généralisent CSP, par exemple les *problèmes de contraintes quantifiées* (QCSP), on peut adapter la méthode algébrique et travailler avec des clones relationnels particuliers, correspondant à une clôture par interprétation avec les symboles $\exists, \wedge, =$ mais aussi avec $\forall$. Puisque les clones relationnels ont plus de relations, ceci signifie qu'on aura moins de polymorphismes. Dans ce cas, les *polymorphismes surjectifs* caractérisent la complexité [3]. Pour le cas des QCSP booléens, la classification est complète, toujours à l'aide du treillis de Post [13]. En général, La classification de QCSP reste incomplète et est au moins aussi difficile que la conjecture de la dichotomie. On observe jusqu'à présent une *trichotomie* avec les complexités P, NP-complet et Pspace-complet.

On reformule souvent QCSP comme le problème de *model checking* pour le fragment de la logique du premier ordre utilisant les symboles $\exists, \forall, \wedge, =$. Le *model checking problem* prend en entrée une formule ϕ et en paramètre une structure $\mathcal{B}$ et demande si $\mathcal{B}$ est un modèle de ϕ. Comme pour le CSP non-uniforme, on considère que $\mathcal{B}$ est fixé et on cherche à connaître la complexité du problème pour chaque $\mathcal{B}$. Martin a étudié la complexité de ce problème pour de nombreux fragments de la logique du premier ordre [33]. Il a établi que soit un fragment est facilement classifiable, ou se réduit au théorème de Schaefer, soit il s'agit de fragments riches du point de vue de la complexité (voir Figure 4). Il reste 3 cas : CSP $(\exists, \wedge, =)$, QCSP $(\forall, \exists, \wedge, =)$ et *QCSP avec disjonction* $(\forall, \exists, \wedge, \vee)$. Dans le cas de CSP ou QCSP la présence ou l'absence du symbole $=$ n'a pas d'effet sur la complexité. Par contre dans ce dernier cas, l'absence d'égalité est cruciale (en présence de l'égalité le problème est facilement classifiable, il est

Pspace-complet si $\mathcal{B}$ a deux éléments ou plus et dans P sinon).

D'autres correspondances de Galois que la correspondance $\mathsf{Pol} - \mathsf{Inv}$ ont étés étudiées en algèbre universelle. Il n'y a pas vraiment de méthode générique pour démontrer les théorèmes techniques nécessaires mais Ferdinand Börner propose des recettes assez complètes de mise en œuvre [2]. Il s'avère que dans le cas de *QCSP avec disjonction* $(\forall, \exists, \wedge, \vee)$ puisqu'on ne dispose plus de l'égalité, on ne travaille plus avec une notion de préservation par une fonction comme pour les polymorphismes avec CSP mais avec des *hyper-fonctions* : c'est-à-dire des fonctions de Dom dans l'ensemble de ses parties, ce qui rend les choses plus complexes. D'un autre côté, la présence de $\vee$ signifie qu'on peut se contenter des hyper-fonctions qui sont unaires ; et, celle de $\forall$, comme pour le cas de QCSP, signifie qu'on peut imposer une notion adéquate de *surjectivité*.

L'expressivité de $\mathcal{B}$ et donc sa complexité pour le problème de QCSP avec disjonction est caractérisée par ses *hyper-endomorphismes surjectifs*. Pour chaque valeur finie de Dom, on se ramène à l'étude d'un treillis *fini*. Pour le cas booléen, celui-ci est trivial et on observe une dichotomie entre Pspace-complet et Logspace. Pour le cas à trois éléments, on peut calculer les éléments importants du treillis *à la main* et on obtient une tetrachotomie entre Logspace, NP-complete, Co-NP-complete et Pspace-complete. De plus, la complexité s'explique de manière uniforme par la possibilité d'*éliminer les quantificateurs* (par exemple, élimination des $\forall$ mais pas des $\exists$ donne un problème NP-complet) [31]. Pour quatre éléments, l'explosion combinatoire empêche de faire ce calcul à la main et en passant à une preuve assistée par ordinateur, on peut démontrer une tetrachotomie [34].

Finalement, en définissant une notion adéquate de *cœur quantifié*, en terme de propriétés de relativisation, et en le caractérisant algébriquement, on est capable de faire abstraction du treillis et de se ramener à l'étude de cas génériques semblables au cas à 4 éléments. On obtient ainsi une tetrachotomie pour n'importe quel domaine [30]. Le méta-problème est NP-complet, ce qui n'est pas forcément insurmontable pour un problème dont la complexité est Pspace-complet.

7 Conclusion

Classifier la complexité des problèmes de satisfaction de contraintes est une question centrale en informatique théorique qui a des liens forts avec les mathématiques (algèbre, combinatoire, logique) et qui en informatique dépasse largement le cadre de l'intelligence artificielle du fait des liens importants avec les bases de données. Mathématiquement très riche et très actif, ce domaine dépasse le cadre de la question de la dichotomie, même si cette conjecture est importante.

Nous avons fait un rapide survol de différents résultats théoriques. Nous avons vu que deux types de restrictions permettant d'obtenir un problème polynomial pour une instance $(\mathcal{A}, \mathcal{B})$: soit le graphe des contraintes $\mathcal{A}$ est arborescent, soit le langage des contraintes $\mathcal{B}$ présente de bonnes propriétés algébriques. Plus récemment, la question des *classes polynomiales hybrides* a pris de l'importance : il s'agit d'identifier des classes polynomiales nouvelles où $\mathcal{A}$ et $\mathcal{B}$ sont restreints simultanément (voir par exemple [11]).

Même si du point de vue pratique on pourrait reprocher à certaines hypothèses d'être irréalistes, de nombreux concepts ont un impact concret. Par exemple, la notion de cœur d'une structure et la notion de paire duale ont des applications en bases de données dans le cadre de l'intégration d'information [37, 38]. Par ailleurs, on voit apparaître des résultats qui font des hypothèses plus réalistes, par exemple sur la manière dont les contraintes sont codées [10], ou qui prennent en compte des contraintes globales [8].

Références

[1] M. Bodirsky. Constraint satisfaction problems with infinite templates. In Creignou et al. [14], pages 196–228.

[2] F. Börner. Basics of galois connections. In Creignou et al. [14], pages 38–67.

[3] F. Börner, A. A. Bulatov, H. Chen, P. Jeavons, and A. A. Krokhin. The complexity of constraint satisfaction games and QCSP. *Inf. Comput.*, 207(9) :923–944, 2009.

[4] A. A. Bulatov. H-coloring dichotomy revisited. *Theor. Comput. Sci.*, 349(1) :31–39, 2005.

[5] A. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *J. ACM*, 53(1) :66–120, 2006.

[6] A. A. Bulatov. Complexity of conservative constraint satisfaction problems. *ACM Trans. Comput. Log.*, to appear.

[7] A. A. Bulatov, A. A. Krokhin, and B. Larose. Dualities for constraint satisfaction problems. In Creignou et al. [14], pages 93–124.

[8] A. A. Bulatov and D. Marx. Constraint satisfaction problems and global cardinality constraints. *Commun. ACM*, 53(9) :99–106, 2010.

[9] A. A. Bulatov and M. Valeriote. Recent results on the algebraic approach to the csp. In Creignou et al. [14], pages 68–92.

[10] H. Chen and M. Grohe. Constraint satisfaction with succinctly specified relations. *J. Comput. Syst. Sci.*, 76(8) :847–860, 2010.

[11] M. C. Cooper, P. G. Jeavons, and A. Z. Salamon. Generalizing constraint satisfaction on trees : Hybrid tractability and variable elimination. *Artif. Intell.*, 174(9-10) :570–584, 2010.

[12] B. Courcelle. Graph rewriting : An algebraic and logic approach. In *Handbook of Theoretical Computer Science, Volume B : Formal Models and Sematics (B)*, pages 193–242. 1990.

[13] N. Creignou, S. Khanna, and M. Sudan. *Complexity classifications of boolean constraint satisfaction problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[14] N. Creignou, P. G. Kolaitis, and H. Vollmer, editors. *Complexity of Constraints - An Overview of Current Research Themes [Result of a Dagstuhl Seminar]*, volume 5250 of *Lncs*. Springer, 2008.

[15] N. Creignou and H. Vollmer. Boolean constraint satisfaction problems : When does post's lattice help ? In Creignou et al. [14], pages 3–37.

[16] V. Dalmau, P. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite-variable logics. In P. Van Hentenryck, editor, *CP*, volume 2470 of *Lncs*, pages 310–326. Springer, 2002.

[17] R. Dechter. Bucket elimination : A unifying framework for reasoning. *Artif. Intell.*, 113(1-2) :41–85, 1999.

[18] R. G. Downey and M.R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, 1999.

[19] T. Feder, F. R. Madelaine, and I. A. Stewart. Dichotomies for classes of homomorphism problems involving unary functions. *Theor. Comput. Sci.*, 314(1-2) :1–43, 2004.

[20] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction : a study through datalog and group theory. *SIAM J. Comput.*, 28, 1999.

[21] E. C. Freuder. A sufficient condition for backtrack-bounded search. *J. ACM*, 32(4) :755–761, 1985.

[22] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1), 2007.

[23] M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *SODA*, pages 289–298. ACM Press, 2006.

[24] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. OUP, 2004.

[25] P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1–2) :185–204, 1998.

[26] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *J. ACM*, 44(4) :527–548, 1997.

[27] R. E. Ladner. On the structure of polynomial time reducibility. *J. ACM*, 22(1) :155–171, 1975.

[28] F. R. Madelaine. Universal structures and the logic of forbidden patterns. *Logical Methods In Computer Science*, 5(2), 2009.

[29] F. R. Madelaine. On the containment of forbidden patterns problems. In *CP*, pages 345–359, 2010.

[30] F. R. Madelaine and B. Martin. A tetrachotomy for positive first-order logic without equality. In *Logic in Computer Science*. IEEE Computer Society, 2011, à paraître.

[31] F. R. Madelaine and B. Martin. The complexity of positive first-order logic without equality. *ACM Trans. Comput. Log.*, to appear.

[32] F. R. Madelaine and I. A. Stewart. Constraint satisfaction, logic and forbidden patterns. *SIAM J. Comput.*, 37(1) :132–163, 2007.

[33] B. Martin. First-order model checking problems parameterized by the model. In *CiE*, pages 417–427, 2008.

[34] B. Martin and J. Martin. The complexity of positive first-order logic without equality II : The four-element case. In *CSL*, pages 426–438, 2010.

[35] D. Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. In L. J. Schulman, editor, *STOC*, pages 735–744. ACM, 2010.

[36] F. Scarcello, G. Gottlob, and G. Greco. Uniform constraint satisfaction problems and database theory. In Creignou et al. [14], pages 156–195.

[37] B. ten Cate, L. Chiticariu, P. G. Kolaitis, and W. Chiew Tan. Laconic schema mappings : Computing the core with sql queries. *PVLDB*, 2(1) :1006–1017, 2009.

[38] B. ten Cate, P. G. Kolaitis, and W. Chiew Tan. Database constraints and homomorphism dualities. In D. Cohen, editor, *CP*, volume 6308 of *Lncs*, pages 475–490. Springer, 2010.