TECHNISCHE
UNIVERSITÄT
WIEN

DIPLOMARBEIT

Constraint Satisfaction Problems Solvable by Local Consistency Methods

ausgeführt am

Institut für
Diskrete Mathematik und Geometrie
TU Wien

unter der Anleitung von

Associate Prof. Dipl.-Ing. Dr.techn. Pinsker Michael

durch

Benedikt Spiegel
Matrikelnummer: 01427242

Wien, 06.01.2023

Verfasser	Betreuer

Kurzfassung

Das Bedingungserfüllungsproblem, kurz CSP, einer vorgegebenen relationalen Sprache, eine Menge mit hier endlich vielen Relationen, ist ein Entscheidungsproblem. Es fragt, ob Variablen eine Wertebelegung zulassen, sodass gewisse, aus der Sprache gebildete Bedingungen erfüllt sind. Die Schwierigkeit eines solchen Entscheidungsproblems wird festgelegt durch die Polymorphismen der entsprechenden relationalen Sprache, das sind Homomorphismen von endlichen Potenzen der Sprache auf sich selbst. Diese Arbeit soll eine mehrheitlich eigenständig lesbare Charakterisierung endlicher relationaler Sprachen bieten, deren CSP sich bereits durch das Überprüfen auf lokale Konsistenz lösen lässt. Die Terminologie inhaltlich relevanter Veröffentlichungen wird in dieser Arbeit vereinheitlicht und Beweise werden möglichst ausführlich erklärt.

Wir definieren die Begriffe der Breite und der relationalen Breite einer endlichen Sprache. Falls eine Sprache beschränkte relationale Breite hat, geben beide Ausdrücke grob gesprochen die Anzahl an Variablen an, für die konsistente Belegungen existieren müssen, damit eine Instanz des CSP lösbar ist. Wir beweisen, dass jede endliche relationale Sprache genau dann beschränkte relationale Breite hat, wenn sie in gewissem Sinne nicht das Lösen von linearen Gleichungen über einem endlichen Primkörper simulieren kann. In diesem Fall kann das entsprechende CSP einer endlichen Sprache bereits durch Betrachtung der zulässigen Belegungen von nur jeweils drei Variablen gleichzeitig gelöst werden. Endliche Sprachen mit beschränkter relationaler Breite können auch durch die Existenz von Polymorphismen charakterisiert werden, welche gewisse, nicht-triviale Gleichungen erfüllen. Dies ermöglicht in weiterer Folge Aussagen über die Entscheidbarkeit der Klasse aller endlicher relationaler Sprachen mit beschränkter relationaler Breite.

Abstract

The constraint satisfaction problem, short CSP, of a given constraint language, a set together with finitely many relations, is a decision problem. It asks whether variables can be assigned values such that constraints built from the constraint language are satisfied. The hardness of this decision problem is determined by the homomorphisms from finite powers of the constraint language to itself, called polymorphisms. This thesis aims to unify terminology of relevant papers and explain proofs in detail to provide a mostly self-contained characterization of finite constraint languages that give rise to CSPs solvable by local consistency methods.

To that end, we define the notions of width and of relational width of a finite constraint language. For languages of bounded relational width, both notions indicate the number of variables upon which consistency must be enforced at a time to solve the corresponding CSP. We prove that a finite constraint language has bounded relational width iff it cannot, in some sense, simulate the problem of solving linear equations over a finite prime field. In that case, the CSP over a finite constraint language is already solvable by local propagation by considering only three variables at a time. Lastly, we show that finite constraint languages of bounded relational width are characterized by the existence of polymorphisms that satisfy certain non-trivial identities. This provides insight to the meta-question of deciding bounded relational width for finite constraint languages.

Acknowledgement

I would like to express my gratitude to my supervisor, Professor Michael Pinsker, for his guidance, support and patience. Our meetings really helped me and I always enjoyed experiencing his enthusiasm for CSPs. Secondly, I would like to thank Tomáš Nagy for his effort and many helpful suggestions.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 06.01.2023

Benedikt Spiegel

Contents

1 Introduction

Let $\mathbb{D}$ be a *constraint language*, i.e., a non-empty set D, called the *domain*, together with finitely many relations $(R^{\mathbb{D}})_{R\in\tau}$ on D. The *constraint satisfaction problem over* $\mathbb{D}$, short CSP($\mathbb{D}$), is the decision problem that asks whether given variables can be assigned elements from D such that finitely many constraints built from relations of $\mathbb{D}$ are satisfied. This framework can be utilized to describe a broad field of problems which are varying in complexity depending on the constraint language. Three examples shall demonstrate this.

Example 1.0.1. Let $\mathbb{D} = (\{0,1\}; =, \neq)$. An instance of CSP($\mathbb{D}$) could consist of the constraints

$$x_1 = x_2, \quad x_2 = x_3, \quad x_3 \neq x_1.$$

Clearly, this particular instance is not satisfiable since transitivity of $=$ in $\{0,1\}$ immediately yields the contradiction $x_1 \neq x_1$ which is false for any value of x_1 in $\{0,1\}$. Any instance of CSP($\mathbb{D}$) can be decided considering three variables at a time and propagating restrictions upon these variables, basically enforcing transitivity of $=$. This problem is said to be solvable by local consistency methods.

Example 1.0.2. Let $\mathbb{D} = (\mathbb{Z}_p; \{0\}, R_0, R_1)$, where $R_d = \{(a,b,c) \in (\mathbb{Z}_p)^3 : a+b+c = d\}$ for $d \in \mathbb{Z}_p$. Then an instance of CSP($\mathbb{D}$) could be

$$x_1 = 0, \quad x_3 = 0, \quad x_2 + x_3 + x_4 = 0, \quad x_1 + x_3 + x_4 = 1, \quad x_1 + x_4 + x_5 = 1.$$

CSP($\mathbb{D}$) corresponds to solving a given system of linear equations in $\mathbb{Z}_p$. While this is still solvable in polynomial time, Theorem 3.1.1 proves that it is not solvable using local consistency methods.

Example 1.0.3. Let $\mathbb{D} = (\{0,1\}; R_{0,0,0}, R_{1,0,0}, R_{0,1,0}, \ldots, R_{1,1,1})$, where for all $i,j,k \in \{0,1\}$, we define $R_{i,j,k} = \{0,1\}^3 \setminus \{(i,j,k)\}$. Since, e.g., $i \vee \neg j \vee k$ equals 1 in the 2-element Boolean algebra iff $(i,j,k) \in R_{0,1,0}$, an instance of CSP($\mathbb{D}$) can be interpreted as a set of clauses, e.g.,

$$x_1 \vee x_2 \vee \neg x_3, \quad x_2 \vee \neg x_4 \vee x_5, \quad \neg x_1 \vee \neg x_2 \vee x_4$$

which have to be satisfied simultaneously. Hence CSP($\mathbb{D}$) is exactly the problem 3-SAT that is well-known to be NP-complete.

Which features of a constraint language $\mathbb{D}$ determine the complexity of CSP($\mathbb{D}$)? This question seems of natural interest to any theoretical computer scientist. In the past years a beautiful Galois connection lead to great success in answering this question. An operation

f on D is called a *polymorphism* of $\mathbb{D}$ if it *preserves* every relation R of $\mathbb{D}$:

$$\begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{pmatrix}, \begin{pmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{pmatrix}, \ldots, \begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{nk} \end{pmatrix} \in R \quad \Rightarrow \quad \begin{pmatrix} f(a_{11}, \ldots, a_{1k}) \\ f(a_{21}, \ldots, a_{2k}) \\ \cdots \\ f(a_{n1}, \ldots, a_{nk}) \end{pmatrix} \in R.$$

The smaller the set of relations of $\mathbb{D}$, the bigger is its corresponding set of polymorphisms, $\mathrm{Pol}(\mathbb{D})$. The fact that the complexity of $\mathrm{CSP}(\mathbb{D})$ is determined by $\mathrm{Pol}(\mathbb{D})$ enabled the use of Universal Algebra. In particular, the complexity of $\mathrm{CSP}(\mathbb{D})$ depends on the identities satisfied by the polymorphisms. In the case of Example 1.0.3 the only polymorphisms of $\mathbb{D}$ are projections. For the constraint language $\mathbb{D}$ in Example 1.0.2, there exists at least one *weak near unanimity (WNU)* polymorphism w, i.e., a polymorphism w satisfying the equations

$$w(y, x, \ldots, x) = w(x, y, x, \ldots, x) = \cdots = w(x, \ldots, x, y)$$

for all x, y in the domain of $\mathbb{D}$. For the constraint language in Example 1.0.1, there even exist WNU polymorphisms of all arities $k \geq 3$.

Zhuk [Zhu20] and Bulatov [Bul17] independently solved a central problem regarding CSPs, originally stated by Feder and Vardi. Today its solution is known as the finite-domain dichotomy theorem.

Theorem 1.0.4. *Let $\mathbb{D}$ be a finite constraint language. If there exists a WNU polymorphism of $\mathbb{D}$, then $\mathrm{CSP}(\mathbb{D})$ can be solved in polynomial time. Otherwise, $\mathrm{CSP}(\mathbb{D})$ is NP-complete.*

While the assumption $P \neq NP$ implies the existence of intermediate problems within NP by Ladner's theorem [Lad75], such problems do not occur as finite-domain CSPs by above dichotomy theorem.

This thesis will focus on characterizing finite constraint languages with CSPs in P that can be solved by decision procedures considering only a fixed number of variables at a time. Such languages will be said to have *bounded relational width*. The constraint language from Example 1.0.1 has bounded relational width. The main contribution of this work is to give a largely self-contained introduction to the notion of *relational width* including important results of the past years and providing extensive proofs. We aim to unify terminology and notation, e.g., for the CSP framework or the definition of relational width, which is varying within the original papers cited in this thesis. In particular, some older papers do not use terminology from [BOP15] such as *pp-constructions* and *minion homomorphisms*, see Definition 2.1.33 and Definition 2.1.39.

A summary of the most important findings covered in this thesis is given in Corollary 6.1.8. It includes the following characterization of finite-domain constraint languages with bounded relational width via weak near unanimity operations which follows from [MM08] and [BK14]. We will give a proof using results from [Bar14].

Theorem 1.0.5. *Let $\mathbb{D}$ be a finite constraint language. Then $\mathbb{D}$ has bounded relational width iff there exist WNU polymorphisms of all arities $k \geq 3$ for $\mathbb{D}$.*

Let us end this introduction by giving a hint at the content of the following chapters.

In Chapter 2 basic definitions that are used throughout the thesis will be introduced. We then give a formal definition of CSPs and we introduce the notions of *(bounded) width* and *(bounded) relational width* using a game and algorithms. Constraint languages with a CSP solvable by such methods are preserved by certain constructions. In particular, we prove that pp-constructions do not change the complexity of the CSP of a language significantly, see Theorem 2.4.9.

Chapter 3 shall explain why the CSP from Example 1.0.2 and any language with a CSP that can simulate solving linear equations over $\mathbb{Z}_p$ for some prime p, does not have bounded relational width. A similar result was originally shown in [FV98]. We give full details, besides some graph theoretic results, of a proof from [Bod21] using the game introduced in Chapter 2 instead. Theorem 3.2.5 describes languages that do not have bounded relational width.

In Chapter 4 we will prove that the inability of a constraint language to simulate solving linear equations already implies that this language does have bounded relational width. This was proved in [BK14] and the result was improved in [Bar14]. The CSP given in Example 1.0.1 is solvable by looking at three variables at a time and propagating information. This is no coincidence. Whenever $\mathbb{D}$ has bounded relational width it is always sufficient to consider three variables at once in order to solve CSP($\mathbb{D}$), see Corollary 4.0.6. We follow the proof in [Bar14].

Chapter 5 concludes the proof that the CSP over any finite constraint language can either be solved by local propagation considering only one variable at a time, by local propagation considering only three variables at a time, or the language does not have bounded relational width at all. We follow and adapt the argumentation from [Dal09].

In Chapter 6 we show that a finite constraint language $\mathbb{D}$ has bounded relational width iff WNU polymorphisms of $\mathbb{D}$ of all arities $k \geq 3$ exist, or equivalently, if WNU polymorphisms v, w of $\mathbb{D}$ of arity 3 and 4 exist which additionally satisfy the equations $v(y, x, x) = w(y, x, x, x)$ for all x, y in the domain of $\mathbb{D}$.

We summarize equivalent properties to bounded relational width for finite constraint languages in Corollary 6.1.8. Finally, we can state an algorithm that decides whether a *core* constraint language, i.e., one whose unary polymorphisms are bijective, has bounded relational width in polynomial time and show that this decision problem is NP-complete if it is extended to arbitrary finite constraint languages.

2 Basic Definitions and Results

2.1 Relational Structures and Polymorphism Clones

Definition 2.1.1. A *signature* τ is a set of relation and function symbols with associated finite arities. A *τ-structure* $\mathfrak{D}$ or *structure over* τ is a set D, the *domain* or *universe* of $\mathfrak{D}$, together with a relation $R^{\mathfrak{D}} \subseteq D^k$ for each relation symbol R of τ of arity k and a function $f^{\mathfrak{D}} : D^k \to D$ for each function symbol f of τ of arity k. We will use the same characters R and f for symbols and their respective relations or functions when there is no danger of confusion.

Signatures only containing function symbols are called *functional signatures* and structures over a functional signature are called *algebras*. Signatures only containing relation symbols are *relational signatures* and structures over relational signatures are *relational structures*. In the context of CSPs we will call relational structures over finite relational signatures *constraint languages*. Any kind of structure is said to be *finite* if its domain is finite.

Notation 2.1.2. We will use different fonts to distinguish different kinds of structures; capital letters $\mathfrak{D}$ in Fraktur typeface for arbitrary structures, capital letters $\mathbf{D}$ in bold typeface for algebras, and capital letters $\mathbb{D}$ in Blackboard bold typeface for relational structures. Unless otherwise stated the corresponding normal font letter D denotes the domain of the structure $\mathfrak{D}, \mathbf{D}$ or $\mathbb{D}$.

Remark 2.1.3. The signature of a structure will not always be of importance or can be chosen arbitrarily. It will sometimes be convenient to write $\mathfrak{D} = (D; R_1^{\mathfrak{D}}, R_2^{\mathfrak{D}}, \dots, f_1^{\mathfrak{D}}, f_2^{\mathfrak{D}}, \dots)$ or $(D, \Gamma, \mathcal{O})$ for the structure with the domain D, the relations $R_1^{\mathfrak{D}}, R_2^{\mathfrak{D}}, \dots$ and the functions $f_1^{\mathfrak{D}}, f_2^{\mathfrak{D}}, \dots$, or relations given by the set Γ and functions given by the set $\mathcal{O}$.

Definition 2.1.4. Let $\mathfrak{D}$ be a τ-structure and $\mathfrak{E}$ be a τ'-structure with $\tau \subseteq \tau'$ such that both structures have the same domain and for every function symbol f of τ and every relation symbol R of τ, we have $f^{\mathfrak{D}} = f^{\mathfrak{E}}$ and $R^{\mathfrak{D}} = R^{\mathfrak{E}}$. Then $\mathfrak{D}$ is the *τ-reduct* or simply a *reduct* of $\mathfrak{E}$ and $\mathfrak{E}$ is a *τ'-expansion* or simply an *expansion* of $\mathfrak{D}$. A τ-structure $\mathfrak{D}$ is said to be *of finite type* if its signature τ is finite. The reducts of finite type of a relational structure are constraint languages.

Definition 2.1.5. Let $\mathfrak{D}$ and $\mathfrak{E}$ be structures over the same signature τ. A *homomorphism from $\mathfrak{D}$ to $\mathfrak{E}$* is a map $h : D \to E$ such that for all integers n, all n-ary relation symbols R and all n-ary function symbols f of τ, and all $a_1, \dots, a_n \in D$, we have

- $(h(a_1), \dots, h(a_n)) \in R^{\mathfrak{E}}$ whenever $(a_1, \dots, a_n) \in R^{\mathfrak{D}}$, and
- $f^{\mathfrak{E}}(h(a_1), \dots, h(a_n)) = h(f^{\mathfrak{D}}(a_1, \dots, a_n))$.

We write $\mathfrak{D} \xrightarrow{hom} \mathfrak{E}$ if there exists a homomorphism from $\mathfrak{D}$ to $\mathfrak{E}$. An *embedding* of $\mathfrak{D}$ into $\mathfrak{E}$ is an injective homomorphism $\iota : D \to E$ satisfying $(\iota(a_1), \ldots, \iota(a_n)) \in R^{\mathfrak{E}}$ if and only if $(a_1, \ldots, a_n) \in R^{\mathfrak{D}}$ for every relation symbol R of τ. A surjective embedding is called an *isomorphism*. An *endomorphism* of $\mathfrak{D}$ is a homomorphism from $\mathfrak{D}$ to $\mathfrak{D}$. The set of all endomorphisms on $\mathfrak{D}$ is denoted by $\mathrm{End}(\mathfrak{D})$. An *automorphism* of $\mathfrak{D}$ is an isomorphism from $\mathfrak{D}$ to $\mathfrak{D}$. The set of all automorphisms of $\mathfrak{D}$ is denoted by $\mathrm{Aut}(\mathfrak{D})$.

Definition 2.1.6. Two structures $\mathfrak{D}$ and $\mathfrak{E}$ are called *isomorphic* if there exists an isomorphism ϕ from $\mathfrak{D}$ to $\mathfrak{E}$. The structures are *homomorphically equivalent* if there exist homomorphisms from $\mathfrak{D}$ to $\mathfrak{E}$ and from $\mathfrak{E}$ to $\mathfrak{D}$.

Definition 2.1.7. Let $\mathfrak{D}$ and $\mathfrak{E}$ be structures over the same signature τ with domains D and E, respectively. We can define a τ-structure $\mathfrak{D} \times \mathfrak{E}$ with domain $D \times E$ by

$$((d_1, e_1), \ldots, (d_n, e_n)) \in R^{\mathfrak{D}\times\mathfrak{E}} \quad \text{iff} \quad (d_1, \ldots, d_n) \in R^{\mathfrak{D}} \text{ and } (e_1, \ldots, e_n) \in R^{\mathfrak{E}}$$

for every n-ary relation symbol R of τ and

$$f^{\mathfrak{D}\times\mathfrak{E}}((d_1, e_1), \ldots, (d_n, e_n)) = (f^{\mathfrak{D}}(d_1, \ldots, d_n), f^{\mathfrak{E}}(e_1, \ldots, e_n))$$

for every n-ary function symbol f of τ.

The product $\mathfrak{D} \times \mathfrak{D}$ is also denoted by $\mathfrak{D}^2$, the k-fold product $\mathfrak{D} \times \cdots \times \mathfrak{D}$, defined analogously, is denoted by $\mathfrak{D}^k$. These structures are called *powers* of $\mathfrak{D}$.

More generally, let I be a non-empty index set and let $(\mathfrak{D}_i)_{i\in I}$ be structures over the same signature τ with the domains D_i, respectively. We define the τ-structure $\mathfrak{C} = \prod_{i\in I} \mathfrak{D}_i$ with domain $\prod_{i\in I} D_i$ by

$$((d_1^{(i)})_{i\in I}, \ldots, (d_n^{(i)})_{i\in I}) \in R^{\mathfrak{C}} \quad \text{iff} \quad (d_1^{(i)}, \ldots, d_n^{(i)}) \in R^{\mathfrak{D}_i} \text{ for every } i \in I,$$

for every n-ary relation symbol R of τ, and

$$f^{\mathfrak{C}}((d_1^{(i)})_{i\in I}, \ldots, (d_n^{(i)})_{i\in I}) = (f^{\mathfrak{D}_i}(d_1^{(i)}, \ldots, d_n^{(i)}))_{i\in I}$$

for every n-ary function symbol f of τ. If $\mathfrak{D} = \mathfrak{D}_i$ for every $i \in I$, we also write $\mathfrak{D}^I$ for $\prod_{i\in I} \mathfrak{D}_i$.

Definition 2.1.8. Let $\mathbb{D}$ and $\mathbb{E}$ be relational τ-structures. Then $\mathbb{D}\cup\mathbb{E}$ denotes the relational τ-structure with domain $D \cup E$ and relations $R^{\mathbb{D}\cup\mathbb{E}} := R^{\mathbb{D}} \cup R^{\mathbb{E}}$ for every relation symbol R of τ.

Definition 2.1.9. Let $\mathbb{D}$ be a relational τ-structure, E a nonempty set and $h : D \to E$ a mapping. Then $h(\mathbb{D})$ denotes the τ-structure with domain $h(D)$ and relations

$$R^{h(\mathbb{D})} := \{(h(d_1), \ldots, h(d_k)) : (d_1, \ldots, d_k) \in R^{\mathbb{D}}\}$$

for every relation symbol R of τ. Clearly, h is a homomorphism from $\mathbb{D}$ to $h(\mathbb{D})$.

Definition 2.1.10. Let $\mathfrak{D}$ and $\mathfrak{E}$ be structures over the same signature τ with domains D and E. The structure $\mathfrak{E}$ is said to be a *substructure of* $\mathfrak{D}$, denoted by $\mathfrak{E} \leq \mathfrak{D}$, if $E \subseteq D$, $R^{\mathfrak{E}} = E^n \cap R^{\mathfrak{D}}$ for every n-ary relation symbol $R \in \tau$ and $f^{\mathfrak{E}} = (E^n \times E) \cap f^{\mathfrak{D}}$ for every n-ary function symbol $f \in \tau$. A set $E \subseteq D$ which is the domain of a substructure of $\mathfrak{D}$ is called a *subuniverse*. A substructure $\mathfrak{A} \leq \mathfrak{D}^k$ of a power of $\mathfrak{D}$ is called a *subpower*. The universe of a subpower is called an *invariant* under $\mathfrak{D}$, an *invariant relation* under $\mathfrak{D}$, or also a *subpower* of $\mathfrak{D}$.

Definition 2.1.11. Let $\mathbf{D}$ be an algebra. An equivalence relation $\sim \subseteq D \times D$ is a *congruence* of $\mathbf{D}$ if $\sim$ is a subuniverse of $\mathbf{D}^2$. A congruence is called *maximal* if it is properly contained only in the trivial congruence D^2.

Remark 2.1.12. The *diagonal* $\{(a, a) : a \in D\}$ is always a congruence and hence every finite algebra with at least two elements has a maximal congruence.

Definition 2.1.13. Let τ be a functional signature. An *identity* (over τ) is a τ-sentence of the form

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_m : s(x_1, \ldots, x_n) = t(y_1, \ldots, y_m)$$

where s and t are τ-terms and $x_1, \ldots, x_n, y_1, \ldots, y_m$ are not necessarily distinct variables. A *height 1 identity* is an identity where the terms s and t are of height 1, i.e., the identity is of the form

$$\forall x_1, \ldots, x_n, y_1, \ldots, y_m : f(x_1, \ldots, x_n) = g(y_1, \ldots, y_m)$$

where f and g are function symbols in τ. An algebra $\mathbf{D}$ is a *model* of a set of identities if those identities hold in $\mathbf{D}$. An identity is *satisfied by operations* $f_1, \ldots, f_k$ on a set D if the function symbols of the identity can be replaced by operations of $f_1, \ldots, f_k$ such that the resulting equation holds for every assignment of the variables occurring in the equation to D.

Definition 2.1.14. A *variety* is a class of algebras over the same functional signature that is closed under homomorphic images, subalgebras, and products. By Birkhoff's HSP theorem [Bir35], a class of algebras of the same signature τ is a variety iff it is the class of models of some set of identities over τ.

Let $\mathbf{D}$ be an algebra. We write $\mathcal{V}(\mathbf{D}) = \mathrm{HSP}\,\mathbf{D}$ for the *variety generated by* $\mathbf{D}$, i.e., the smallest variety containing $\mathbf{D}$.

Definition 2.1.15. A set of operations $\mathcal{D}$ on a nonempty set D is called a *(function) clone* if $\mathcal{D}$ contains all projections $\pi_i^n : D^n \to D, (x_1, \ldots, x_n) \mapsto x_i$ for $n \in \mathbb{N} \setminus \{0\}$ and $i \in \{1, \ldots, n\}$ and is closed under composition, i.e., for any function $f \in \mathcal{D}$ of arity m and any functions $g_1, \ldots, g_m$ all of arity k in $\mathcal{D}$ the map

$$f(g_1, \ldots, g_m) : \begin{cases} D^k & \to D, \\ (x_1, \ldots, x_k) & \mapsto f(g_1(x_1, \ldots, x_k), \ldots, g_m(x_1, \ldots, x_k)) \end{cases}$$

lies in $\mathcal{D}$. Function clones will be denoted by calligraphic font letters, usually by the same letter as the corresponding domain (we also use calligraphic font for other objects, e.g., sets

of functions that are not necessarily a function clone). For an arbitrary set $\mathcal{O}$ of operations on a set D we write $\langle\mathcal{O}\rangle$ for the smallest clone containing $\mathcal{O}$. If $\mathbf{D}$ is a τ-algebra, we write $\mathrm{Clo}(\mathbf{D}) := \langle\{f^{\mathbf{D}} : f \in \tau\}\rangle$ for the *clone of term operations* of $\mathbf{D}$ and $\mathrm{Clo}_k(\mathbf{D})$ for all operations in $\mathrm{Clo}(\mathbf{D})$ of arity k.

Definition 2.1.16. Two algebras $\mathbf{D}, \mathbf{E}$ with the same domain are called *term-equivalent* if $\mathrm{Clo}(\mathbf{D}) = \mathrm{Clo}(\mathbf{E})$ holds.

Definition 2.1.17. Let τ be a relational signature. A first-order τ-formula ϕ is called *primitive positive* or a *pp-formula* if it is an existentially quantified conjunction of atomic τ-formulas $\psi_1, \ldots, \psi_l$, i.e.,

$$\phi(x_1, \ldots, x_n) = \exists x_{n+1}, \ldots, x_m(\psi_1 \wedge \cdots \wedge \psi_l),$$

where atomic formulas are of the form $R(y_1, \ldots, y_k)$ with $y_i \in \{x_1, \ldots, x_m\}$ for all $i \in \{1, \ldots, k\}$ and R in τ, or of the form $y = y'$ with $y, y' \in \{x_1, \ldots, x_m\}$. A pp-formula without free variables is a *primitive positive sentence*.

Definition 2.1.18. A relation R is *pp-definable* in a relational τ-structure $\mathbb{D}$ if it is definable by a pp-formula $\phi(x_1, \ldots, x_n)$ over τ, i.e.,

$$\forall a_1, \ldots, a_n \in D: \quad (a_1, \ldots, a_n) \in R \quad \Longleftrightarrow \quad \mathbb{D} \models \phi(a_1, \ldots, a_n).$$

Definition 2.1.19. Let Γ be a set of relations on a nonempty domain D. We write $\langle\Gamma\rangle$ for the set of all pp-definable relations of (D, Γ). Any set of relations which is closed under pp-definitions is called a *relational clone*.

Definition 2.1.20. A k-ary function $f : D^k \to D$ is said to *preserve* an n-ary relation $R \subseteq D^n$, or equivalently R is *closed* or *invariant* under f, if R is a subuniverse of the structure $(D, \{f\})^n$. We abbreviate this by $f \rhd R$.

In that case applying f to the rows of a matrix $(a_{ij}) \in D^{n\times k}$ whose columns are tuples of R yields a column that is again a tuple of R.

$$\begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{pmatrix}, \begin{pmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{pmatrix}, \ldots, \begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{nk} \end{pmatrix} \in R \quad \Rightarrow \quad \begin{pmatrix} f(a_{11}, \ldots, a_{1k}) \\ f(a_{21}, \ldots, a_{2k}) \\ \cdots \\ f(a_{n1}, \ldots, a_{nk}) \end{pmatrix} \in R. \tag{2.1}$$

Definition 2.1.21. Let $\mathbb{D}$ be a relational τ-structure and k a positive integer. A function $f : D^k \to D$ is called a *polymorphism* of $\mathbb{D}$ if f preserves every relation of $\mathbb{D}$. We define

$$\mathrm{Pol}_k(\mathbb{D}) := \{f : D^k \to D \mid \forall R \in \tau, f \rhd R^{\mathbb{D}}\}$$

to be the set of all k-ary polymorphisms of $\mathbb{D}$. Then

$$\mathrm{Pol}(\mathbb{D}) := \bigcup_{k=1}^{\infty} \mathrm{Pol}_k(\mathbb{D})$$

is the *polymorphism clone* of $\mathbb{D}$. We use the same notion on mere sets of relations, i.e., for a set Γ of relations on D we write $\mathrm{Pol}(\Gamma)$ for the set of operations that preserve every relation in Γ.

Remark 2.1.22. Let $\mathbb{D}$ be a relational structure on a domain D. Then the polymorphism clone $\mathrm{Pol}(\mathbb{D})$ of $\mathbb{D}$ is a clone.

Definition 2.1.23. Let $\mathbf{D}$ be an algebra. We write $\mathrm{Inv}(\mathbf{D})$ for the set of all invariant relations under $\mathbf{D}$. The same notion is used for a set of functions $\mathcal{O}$ on a common domain D, i.e., we write $\mathrm{Inv}(\mathcal{O})$ for the set of relations on D that are preserved by every operation in $\mathcal{O}$.

Remark 2.1.24. The relation $\triangleright$ induces an antitone Galois connection between sets of operations and sets of relations on a fixed common domain D via the order reversing functions Inv and Pol. The equations

$$\mathrm{Inv}(\mathrm{Pol}(\mathrm{Inv}(\mathcal{O}))) = \mathrm{Inv}(\mathcal{O})$$

and

$$\mathrm{Pol}(\mathrm{Inv}(\mathrm{Pol}(\Gamma))) = \mathrm{Pol}(\Gamma)$$

hold as a consequence. The following two theorems describe the closure operators $\mathrm{Inv} \circ \mathrm{Pol}$ and $\mathrm{Pol} \circ \mathrm{Inv}$, we refer to, e.g., [Sze86] for proofs.

Theorem 2.1.25. *Let $\mathcal{O}$ be a set of operations on a finite set D. Then $\langle \mathcal{O} \rangle = \mathrm{Pol}(\mathrm{Inv}(\mathcal{O}))$.*

Theorem 2.1.26. *Let Γ be a set of relations on a finite set D. Then $\langle \Gamma \rangle = \mathrm{Inv}(\mathrm{Pol}(\Gamma))$.*

Definition 2.1.27. A finite relational structure $\mathbb{E}$ is a *core* if all of its endomorphisms are automorphisms. A constraint language that is a core is also called a *core constraint language*. Let $\mathbb{D}$ and $\mathbb{E}$ be relational structures over the same signature. Then $\mathbb{E}$ is called a *core of* $\mathbb{D}$ if $\mathbb{E}$ is a core and homomorphically equivalent to $\mathbb{D}$.

Lemma 2.1.28. *Every finite relational structure has a core and all of its cores are isomorphic.*

Proof. Let $\mathbb{D}$ be a finite relational structure. Assume $\mathbb{D}$ is no core. Then there exists an endomorphism $h \in \mathrm{End}(\mathbb{D}) \setminus \mathrm{Aut}(\mathbb{D})$ by definition of core. Note that selfmaps on finite sets are injective if and only if they are surjective. Therefore any surjective endomorphism f must in particular permute the tuples of any relation R of $\mathbb{D}$, i.e., $(a_1, \ldots, a_n) \in R$ if and only if $(f(a_1), \ldots, f(a_n)) \in R$ for every n-ary relation R, implying that f is an automorphism. Thus h cannot be surjective. We define $\mathbb{D}'$ to be the induced substructure of $\mathbb{D}$ with domain $D' := h(D)$ by restricting relations to the new domain. Now, as observed, $\mathbb{D}'$ has strictly smaller domain than $\mathbb{D}$ and the structures are homomorphically equivalent by h and $h' : D' \to D, x \mapsto x$. Repeating this construction at most $|D|$ times yields a finite chain of successively strictly smaller and homomorphically equivalent relational structures $\mathbb{D}_0 := \mathbb{D}$ and $\mathbb{D}_{k+1} := (\mathbb{D}_k)'$ until the assumption $\mathrm{Aut}(\mathbb{D}_k) \subsetneq \mathrm{End}(\mathbb{D}_k)$ fails. The last relational structure of such sequence is a core of $\mathbb{D}$.

Let $\mathbb{E}$ and $\mathbb{E}'$ be two cores of the structure $\mathbb{D}$ and let $g : \mathbb{D} \to \mathbb{E}$, $h : \mathbb{E} \to \mathbb{D}$, $g' : \mathbb{D} \to \mathbb{E}'$, $h' : \mathbb{E}' \to \mathbb{D}$ be homomorphisms. The endomorphisms $g \circ h' \circ g' \circ h$ and $g' \circ h \circ g \circ h'$ of $\mathbb{E}$ and $\mathbb{E}'$ are automorphisms. Thus $g' \circ h : \mathbb{E} \to \mathbb{E}'$ and $g \circ h' : \mathbb{E}' \to \mathbb{E}$ are bijective homomorphisms. Finiteness of both structures implies coinciding sizes of the relations $R^{\mathbb{E}}$ and $R^{\mathbb{E}'}$ for every relation symbol R. Hence both maps must be isomorphisms. □

Definition 2.1.29. Let $\mathbb{D}$ be a core with domain D. We say $\mathbb{E}$ is the *singleton expansion* of $\mathbb{D}$ if $\mathbb{E}$ is obtained from $\mathbb{D}$ by adding all singleton relations, i.e., the relations $\{a\}$ for every $a \in D$, to $\mathbb{D}$.

Definition 2.1.30. An operation f is called *idempotent* if $f(x, \dots, x) = x$ holds for all x in the domain of f. A clone is *idempotent* if it contains only idempotent operations. A constraint language is called *idempotent* if it contains all singleton relations, implying that its polymorphism clone is idempotent. The *idempotent reduct* of a clone is the clone consisting of all of its idempotent operations.

Definition 2.1.31. Let $\mathbb{D}$ and $\mathbb{E}$ be relational structures with possibly different signatures. We say $\mathbb{D}$ *pp-interprets* $\mathbb{E}$, or that $\mathbb{E}$ is *pp-interpretable* in $\mathbb{D}$, if there exists $n \geq 1$ and a mapping f from a subset of D^n onto E such that all of the following relations are pp-definable in $\mathbb{D}$:

- the domain of f;
- the preimage of the equality relation on E under f (the kernel of f), viewed as a $2n$-ary relation on D;
- the preimage of every relation in $\mathbb{E}$ under f, where the preimage of a k-ary relation under f is again regarded as a kn-ary relation on D.

A similar but less general notion of interpretations are pp-powers. We require that the partial surjective mapping f from above definition of pp-interpretations is an isomorphism. Let us give a precise explanation what that means.

Definition 2.1.32. Let $\mathbb{D}$ and $\mathbb{E}$ be relational structures with possibly different signatures. We say that $\mathbb{E}$ is a *pp-power* of $\mathbb{D}$ if $\mathbb{E}$ is isomorphic to a structure with domain D^n, where $n \geq 1$, whose relations are pp-definable from $\mathbb{D}$; as before, a k-ary relation on D^n is regarded as kn-ary relation on D.

Definition 2.1.33. Let $\mathbb{D}$ and $\mathbb{E}$ be finite relational structures. We say that $\mathbb{D}$ *pp-constructs* $\mathbb{E}$, or that $\mathbb{E}$ is *pp-constructible* from $\mathbb{D}$, if there exists a sequence $\mathbb{D} = \mathbb{C}_1, \mathbb{C}_2, \dots, \mathbb{C}_k = \mathbb{E}$ such that for every $1 \leq i < k$,

- $\mathbb{C}_i$ pp-interprets $\mathbb{C}_{i+1}$, or
- $\mathbb{C}_{i+1}$ is homomorphically equivalent to $\mathbb{C}_i$, or
- $\mathbb{C}_i$ is a core, and $\mathbb{C}_{i+1}$ is obtained from $\mathbb{C}_i$ by adding a singleton unary relation.

According to [BOP15] any pp-construction can be achieved with only two constructions.

Theorem 2.1.34. *The following are equivalent for finite relational structures* $\mathbb{D}$, $\mathbb{E}$.

(i) $\mathbb{E}$ *is pp-constructible from* $\mathbb{D}$.

(ii) $\mathbb{E}$ *is homomorphically equivalent to a pp-power of* $\mathbb{D}$.

Proof. Corollary 3.10 in [BOP15]. □

Definition 2.1.35. Let $\mathcal{D}, \mathcal{E}$ be clones. A map $\xi : \mathcal{D} \to \mathcal{E}$ is called a *clone homomorphism* if

1. ξ preserves arities,
2. $\xi(\pi_i^n) = \pi_i^n$ holds for all projections π_i^n, $1 \leq i \leq n$ and $n \in \mathbb{N} \setminus \{0\}$, and
3. ξ preserves compositions, i.e.,

$$\xi(f \circ (g_1, \ldots, g_n)) = \xi(f) \circ (\xi(g_1), \ldots, \xi(g_n))$$

for all n-ary functions f and all m-ary functions g of $\mathcal{D}$.

We write $\mathcal{D} \leq_{ch} \mathcal{E}$ if there exists a clone homomorphism $\xi : \mathcal{D} \to \mathcal{E}$, and $\mathcal{D} \sim_{ch} \mathcal{E}$ if $\mathcal{D} \leq_{ch} \mathcal{E} \leq_{ch} \mathcal{D}$.

Remark 2.1.36. Clone homomorphisms preserve identities. For example, if a clone $\mathcal{D}$ has a Maltsev operation, i.e., there exists an $m \in \mathcal{D}$ such that $x = m(x, y, y) = m(y, y, x)$ holds for all x, y in the domain of m, and $\xi : \mathcal{D} \to \mathcal{E}$ is a clone homomorphism, then $\xi(m)$ is a Maltsev operation of $\mathcal{E}$.

Definition 2.1.37. Let $\mathcal{D}$ be a function clone. We denote by $\mathrm{H}(\mathcal{D})$ all function clones obtained by letting $\mathcal{D}$ act naturally on the classes of an invariant equivalence relation on the domain of $\mathcal{D}$. We denote by $\mathrm{S}(\mathcal{D})$ all function clones obtained by letting $\mathcal{D}$ act on an invariant subset of its domain via restriction. We denote by $\mathrm{P}_{\mathrm{fin}}(\mathcal{D})$ the function clones such that $\mathcal{D}$ acts component-wise on finite powers of its domain. Finally, the operator $\mathrm{E}(\mathcal{D})$ yields all function clones gained by extending $\mathcal{D}$, i.e., by adding functions to $\mathcal{D}$. We use constructions like $\mathrm{H}\,\mathrm{S}\,\mathrm{P}_{\mathrm{fin}}$ to describe the classes gained by consecutively applying these operators to (classes) of function clones.

The following theorem is stated in [BOP15]. According to this paper, it is as a consequence of the Galois connection between relational structures and function clones, see Remark 2.1.24, and Birkhoff's HSP theorem [Bir35]. In [BP15], a more general version of the below theorem is proved with inspirations from [BJK05].

Theorem 2.1.38 ([BOP15, Theorem 1.1])**.** *Let $\mathbb{D}$ and $\mathbb{E}$ be finite relational structures. Then the following are equivalent.*

1. *$\mathbb{E}$ is pp-interpretable in $\mathbb{D}$*
2. $\mathrm{Pol}(\mathbb{E}) \in \mathrm{E}\,\mathrm{H}\,\mathrm{S}\,\mathrm{P}_{\mathrm{fin}}(\mathrm{Pol}(\mathbb{D}))$
3. $\mathrm{Pol}(\mathbb{D}) \leq_{ch} \mathrm{Pol}(\mathbb{E})$.

Definition 2.1.39. Let $\mathcal{D}, \mathcal{E}$ be clones. A map $\xi : \mathcal{D} \to \mathcal{E}$ is a *minion homomorphism* or *h1 clone homomorphism* if

1. ξ preserves arities, and

2. ξ preserves composition with projections, that is, for all operations $f \in \mathcal{D}$ and projections $\pi^n_{i_1}, \dots, \pi^n_{i_m}$, we have

$$\xi(f \circ (\pi^n_{i_1}, \dots, \pi^n_{i_m})) = \xi(f) \circ (\pi^n_{i_1}, \dots, \pi^n_{i_m}).$$

We write $\mathcal{D} \leq_{h1} \mathcal{E}$ if there exists a minion homomorphism from $\mathcal{D}$ to $\mathcal{E}$.

Remark 2.1.40. Minion homomorphisms preserve height 1 identities. For example, if $\mathcal{D}$ has a ternary WNU operation w, i.e., the equations $w(y,x,x) = w(x,y,x) = w(x,x,y)$ hold for all x, y in the domain of w, and $\xi : \mathcal{D} \to \mathcal{E}$ is a minion homomorphism, then $\xi(w)$ is a WNU operation of $\mathcal{E}$.

In order to describe the corresponding algebraic perspective for pp-constructions, in particular to mimic homomorphic equivalence, reflections were introduced in [BOP15].

Definition 2.1.41. Let $\mathbf{D}$ be a τ-algebra. Let E be a set and $h_1 : E \to D$ and $h_2 : D \to E$ functions. Then the τ-algebra $\mathbf{E}$ with domain E and a function

$$f^{\mathbf{E}} : (x_1, \dots, x_k) \mapsto h_2(f^{\mathbf{D}}(h_1(x_1), \dots, h_1(x_k)))$$

for every k-ary function $f^{\mathbf{D}}$ of $\mathbf{D}$ is called a *reflection* of $\mathbf{D}$.

Definition 2.1.42. For a class $\mathcal{K}$ of algebras, we denote by $\mathrm{R}(\mathcal{K})$ the class of all reflections of algebras in $\mathcal{K}$. The construction in Definition 2.1.41 can be extended to sets of operations on a fixed domain without a signature. If $\mathcal{D}$ is a function clone, we write $\mathrm{R}(\mathcal{D})$ for the sets of functions that can be obtained from $\mathcal{D}$ via reflection.

Remark 2.1.43. A reflection of a clone must not necessarily be a clone again. This issue is resolved by applying the operator E not only to clones but also to mere sets of functions.

Theorem 2.1.44 ([BOP15, Theorem 1.8])**.** *Let $\mathbb{D}$ and $\mathbb{E}$ be finite relational structures. Then the following are equivalent.*

1. *$\mathbb{E}$ is pp-constructible in $\mathbb{D}$;*
2. *$\mathrm{Pol}(\mathbb{E}) \in \mathrm{E}\,\mathrm{R}\,\mathrm{P}_{\mathrm{fin}}(\mathrm{Pol}(\mathbb{D}))$;*
3. *$\mathrm{Pol}(\mathbb{D}) \leq_{h1} \mathrm{Pol}(\mathbb{E})$.*

2.2 Constraint Satisfaction Problems

We have already given three examples of Constraint Satisfaction Problems in the introduction, Examples 1.0.1, 1.0.2, and 1.0.3. We want to formalize the framework in this section.

Definition 2.2.1. Let $\mathbb{D}$ be a constraint language. An *instance of* $\mathrm{CSP}(\mathbb{D})$ is a triple $I = (V, D, \mathcal{C})$, where V is a finite set of *variables*, D is the domain of $\mathbb{D}$, $\mathcal{C}$ is finite list of *constraints* $C = (s, R)$, where the *scope* s is a finite tuple of variables and the *constraint relation* R is a relation of $\mathbb{D}$ whose arity equals the length of s. An instance I is called *trivial* if it contains a constraint with empty constraint relation.

Remark 2.2.2. Sometimes, especially in Chapter 4, we will just speak of an instance $I = (V, D, \mathcal{C})$ without mentioning a constraint language $\mathbb{D}$ or that I is an instance of CSP($\mathbb{D}$). In that case, the exact form of the constraint language is not of importance or should be clear from the context.

Definition 2.2.3. Let $\mathbb{D}$ be a constraint language. Let $I = (V, D, \mathcal{C})$ be an instance of CSP($\mathbb{D}$). Any function $f : V \to D$ is called an *assignment*. It is said to *satisfy* a constraint $C = (s, R)$ if $f(s) \in R$ where $f(s)$ is the tuple obtained by component-wise application of f to s. Furthermore, f is called a *solution* of the instance I if f satisfies all constraints $C \in \mathcal{C}$.

Notation 2.2.4. For two instances $I = (V, D, \mathcal{C})$ and $I' = (V', D', \mathcal{C}')$ we write $I \sqsubseteq I'$ if $V = V'$, $D = D'$, and for any constraint $(s, R') \in \mathcal{C}'$ there exists some relation $R \subseteq R'$ such that $(s, R) \in \mathcal{C}$. Note that any solution of I is also a solution of I'.

Notation 2.2.5. For a constraint $C = (s, R)$ with scope $s = (x_1, \ldots, x_n)$ we will use set-theoretic phrases like s *contains (is contained in)* a set W of variables or define functions on s to abbreviate analogue statements for the set $\{x_1, \ldots, x_n\}$. The scope $s = (x_1, \ldots, x_n)$ of a constraint *contains (is contained in)* a set $W \subseteq V$ if $W \subseteq \{x_1, \ldots, x_n\}$ (or $W \supseteq \{x_1, \ldots, x_n\}$). Similarly, we write $f : s \to D$ for a function $f : \{x_1, \ldots, x_n\} \to D$.

Definition 2.2.6. Let $\mathbb{D}$ be a constraint language. Then CSP($\mathbb{D}$) is the problem of deciding whether a given instance $I = (V, D, \mathcal{C})$ such that every constraint relation of I is a relation of $\mathbb{D}$ has a solution.

Example 2.2.7. We can write the instance in Example 1.0.1 as $I = (\{x_1, x_2, x_3\}, \{0, 1\}, \mathcal{C})$ with $\mathcal{C} = \{((x_1, x_2), =), ((x_2, x_3), =), ((x_3, x_1), \neq)\}$. It does not have a solution. The instance in Example 1.0.2 does have a solution $f(x_1) = f(x_3) = f(x_5) = 0$, $f(x_2) = -1$, $f(x_4) = 1$.

Remark 2.2.8. Let $\mathbb{D}$ be a constraint language. An instance $I = (V, D, \mathcal{C})$ of CSP($\mathbb{D}$) with constraints $\{(s_1, R_1), \ldots, (s_m, R_m)\}$ and scopes $s_i = (x^i_1, \ldots, x^i_{k_i})$, where k_i is the arity of R_i for every $i \in \{1, \ldots, m\}$, can be translated into a primitive positive sentence

$$\phi_I := \exists x_1, \ldots, x_n \, R_1(x^1_1, \ldots, x^1_{k_1}) \wedge \cdots \wedge R_m(x^m_1, \ldots, x^m_{k_m})$$

where $\{x_1, \ldots, x_n\}$ is a full list of all variables in $s_1, \ldots, s_m$. The instance I now has a solution if and only if the formula ϕ_I is true in $\mathbb{D}$ by definition.

This very natural view of CSPs as truth of primitive positive sentences or satisfiability of conjunctions of atomic formulas in a relational structure immediately suggests versatility and a broad field of problems that can be described using CSPs. Bodirsky and Grohe proved in [BG08] that every computational decision problem is polynomial-time Turing-equivalent to a constraint satisfaction problem with an infinite template.

Remark 2.2.9. Let $\mathbb{D}$ be a constraint language and let $I = (V, D, \mathcal{C})$ be an instance of CSP($\mathbb{D}$). Denote the signature of $\mathbb{D}$ by τ. We construct another τ-structure $\mathbb{E}_I$ as follows. The relational structure $\mathbb{E}_I$ has universe V and for every relation symbol R of τ, the relation $R^{\mathbb{E}_I}$ shall contain exactly those tuples s such that $(s, R^{\mathbb{D}})$ is a constraint of I.

Then $h : V \to D$ is a solution of I if and only if h is a homomorphism from $\mathbb{E}_I$ to $\mathbb{D}$. This perception of instances as relational structures gives opportunity to compare instances of CSP($\mathbb{D}$) through homomorphisms. If I, J are instances of CSP($\mathbb{D}$) and h is a homomorphism from $\mathbb{E}_J$ to $\mathbb{E}_I$, then a solution f of I yields a solution $f \circ h$ of J. Hence solving J is easier than solving I. In Chapter 5 we will make use of this observation. For some constraint languages, showing that certain instances which are actually easier than I, i.e., their associated relational structures map homomorphically to I, do not have a solution already yields the unsolvability of I.

Example 2.2.10. Let k be a positive integer and let $\mathcal{G} = (G; E)$ be an undirected graph. A *k-colouring* of $\mathcal{G}$ is an assignment $h : G \to \{1, \dots, k\}$ such that for any two adjacent vertices v, w of $\mathcal{G}$, the colours $h(v)$ and $h(w)$ are different. Hence, an assignment $h : G \to \{1, \dots, k\}$ is a k-colouring iff it is a homomorphism from $\mathcal{G}$ to the constraint language

$$\mathbb{D}_{\text{k-COL}} := (\{1, \dots, k\}, \neq).$$

If $\mathcal{G}$ has a k-colouring, we say that $\mathcal{G}$ is *k-colourable*.

The problem of deciding whether a given graph is k-colourable is called *k-colourability*. This problem is described by CSP($\mathbb{D}_{\text{k-COL}}$).

2.3 Local Consistency Methods

This section shall clarify what is meant by local consistency methods in the title of this thesis. A short, historically motivated definition of width via datalog programs is followed by the pebble game, which will be used in the proof that solving linear equations is comparably hard, and relational width. The part concerning datalog up to Definition 2.3.8 is not needed for the rest of the thesis.

2.3.1 Datalog and width

Datalog is a framework heavily studied studied in database theory. It can be described as the language of logical programming without function symbols, see [Ull88]. We give a short definition following [Bod21].

Definition 2.3.1. A *database program* consists of a finite set of *rules*, each rule is usually written in the form

$$\psi :- \phi_1, \dots, \phi_m$$

where $m \geq 0$ and $\psi, \phi_1, \dots, \phi_m$ are atomic formulas over some relational signature. In this context we will call such formulas *predicates*. The relation symbol R of a predicate $R(x_1, \dots, x_n)$ is the *name* or *symbol* of the predicate and refers to a relation. In practice, this relation represents storage.

The formula ψ is called the *head* of the rule and $\phi_1, \dots, \phi_m$ is called the *body*. The variables from the head $\psi = R(x_1, \dots, x_n)$ must occur in the body.

The relation symbols that occur in the head of a rule are the names of *intensional database predicates* or short *IDBs*. The relation symbols that only occur in the body of a

rule but not in the head of any rule are the names of *extensional database predicates* or *EDBs*. The underlying relations corresponding to IDBs represent the information that a program derives from initial data corresponding to the EDBs.

A datalog program has *width* (k, l) if all IDBs of its rules are at most k-ary, and if there are at most l distinct variables in the body of each rule (the names of k and l might be switched in older literature and are chosen to fit our upcoming definition of relational width (k, l)).

Definition 2.3.2. Let Π be a datalog program, let τ be the set of symbols of its EDBs, let σ be the set of symbols of its IDBs, and let $\mathbb{D}$ be a finite relational τ-structure. Then the program Π can be used to define a $(\tau \cup \sigma)$-expansion $\Pi(\mathbb{D})$ of $\mathbb{D}$ such that

$$\Pi(\mathbb{D}) \models \forall \overline{x}((\phi_1 \wedge \cdots \wedge \phi_m) \to \psi) \quad \text{holds for every rule} \quad (\psi :- \phi_1, \ldots, \phi_m) \text{ of } \Pi \tag{2.2}$$

and such that the relations of any $(\tau \cup \sigma)$-expansion of $\mathbb{D}$ with property (2.2) contain those of $\Pi(\mathbb{D})$.

Remark 2.3.3. Instead of the above definition as minimal $(\tau \cup \sigma)$-structure satisfying property (2.2), we can construct $\Pi(\mathbb{D})$ the following way.

Let $\mathbb{D}_0$ be the $(\tau \cup \sigma)$-expansion obtained from $\mathbb{D}$ by adding an empty relation for every relation symbol in σ. For every $i \geq 0$, we can now inductively construct a new structure $\mathbb{D}_{i+1}$ from $\mathbb{D}_i$ by adding tuples to the relations of $\mathbb{D}_i$ belonging to the IDBs of Π. Whenever

$$Q(\overline{x}) :- R_1(\overline{x}_1), \ldots, R_m(\overline{x}_m)$$

is a rule of Π and the conjunction of the atomic formulas in its body holds in $\mathbb{D}_i$ under an assignment f of the variables of the rule to the domain of $\mathbb{D}_i$, we add the tuple $f(\overline{x})$, where f is applied component-wise to $\overline{x}$, to the relation Q of $\mathbb{D}_{i+1}$. Since $\mathbb{D}$ is finite, after finitely many steps, say r iterations, no more tuples are added, i.e., $\mathbb{D}_i = \mathbb{D}_{i+1}$ holds for $i \geq r$. We set $\Pi(\mathbb{D}) := \mathbb{D}_r$.

Example 2.3.4. The following program Π checks if a finite graph with edge relation *edge* is not 2-colourable:

$$\begin{aligned} \text{oddpath}(x, y) &:- \text{edge}(x, y) \\ \text{evenpath}(x, y) &:- \text{oddpath}(x, z), \text{edge}(z, y) \\ \text{oddpath}(x, y) &:- \text{evenpath}(x, z), \text{edge}(z, y) \\ \text{not2colourable} &:- \text{oddpath}(x, x) \end{aligned}$$

The program will compute the binary relation corresponding to the IDB oddpath(x, y) of the input graph to store vertices which are connected by a path of odd length. If there exists a vertex v of the graph such that v is connected to itself via a path of odd length, the empty tuple will be added to the relation of the IDB not2colourable of arity zero. In that case, the graph is not 2-colourable. Therefore, 2-colourability can be solved by a datalog program of width $(2, 3)$ as the above program has at most 2 distinct variables in the head, and at most 3 distinct variables in the body of each rule.

Example 2.3.4 shows how a datalog program can be used as decision procedure. We require a distinguished IDB p of arity zero to be part of the datalog program to do so. In above example this predicate is not2colourable and it indicates whether the problem (colouring the graph) has a solution or not.

Let us specify how a datalog program Π with distinguished IDB p of arity zero would apply to an instance $I = (V, D, \mathcal{C})$ of CSP($\mathbb{D}$) using our notation. The procedure would start with EDBs $R(\overline{x})$ for every constraint $(\overline{x}, R)$ of the instance and an empty relation on D of appropriate arity for every IDB symbol of Π. Proceeding as described in Remark 2.3.3 yields the relational structure $\Pi(\mathbb{D}_I)$, where $\mathbb{D}_I$ is exactly I interpreted as relational structure. The instance is unsatisfiable if the corresponding relation of the predicate p in $\Pi(\mathbb{D}_I)$ is nonempty and the program Π is *sound* for CSP($\mathbb{D}_I$). Let us define what sound means.

Definition 2.3.5. Let $\mathbb{D}$ be a finite constraint language. A datalog program Π is *sound* for CSP($\mathbb{D}$) if it does not categorize satisfiable instances of CSP($\mathbb{D}$) as unsatisfiable. A program Π *solves* CSP($\mathbb{D}$) if it correctly detects all unsatisfiable instances and is sound.

Definition 2.3.6. A finite constraint language $\mathbb{D}$ has width (k, l) for $k \leq l$ if there exists a datalog program of width (k, l) that solves CSP($\mathbb{D}$). It has bounded width if it has width (k, l) for some k and l.

To determine whether given constraint language $\mathbb{D}$ has width (k, l), the definition suggests that one has to consider all datalog programs of width (k, l) which are sound for $\mathbb{D}$ and check whether one of the programs solves the problem. Luckily in [FV98, Theorem 17] Feder and Vardi argue that there is one *canonical datalog program of width* (k, l) for every choice of $k \leq l$, which subsumes all the inferences any sound datalog program of width (k, l) would perform.

Theorem 2.3.7 ([FV98, Theorem 17])**.** *For every finite constraint language $\mathbb{D}$ and positive integers $k \leq l$, there is a canonical datalog program of width (k, l) with the following property: if any datalog program of width (k, l) solves* CSP($\mathbb{D}$)*, then the canonical one does.*

Instead of a rigorous proof, we want to describe how the canonical datalog program of width (k, l) operates. It is today known as the (k, l)-*consistency algorithm.*

Definition 2.3.8. Let $\mathbb{D}$ be a constraint language, let $I = (V, D, \mathcal{C})$ be an instance of CSP($\mathbb{D}$) and let $W \subseteq V$. A mapping $f : W \to D$ is a *partial solution for I on W* if f solves every constraint of I which has a scope contained in W.

Remark 2.3.9. Note that the definition of a partial solution for an instance on a set W of variables does not depend on constraints with scopes containing variables that are not in W.

Notation 2.3.10. Let W, D be sets, let $\mathcal{F} \subseteq D^V$ be a set of functions and let $W \subseteq V$. We will sometimes write $F_{|_W} = \{f_{|_W} : f \in \mathcal{F}\}$ for the set of functions of $\mathcal{F}$ restricted to W.

The (k, l)-consistency algorithm with input instance I roughly operates in two steps.

1. Find the set $\mathcal{F}$ of all partial solutions of I on subsets $W \subseteq V$ of at most l elements.

2. Eliminate assignments from $\mathcal{F}$ that are not consistent on sets of up to k variables, i.e., for every set W of up to k variables and sets $W_1 \supseteq W$ and $W_2 \supseteq W$ of up to l variables, the equation $(\mathcal{F}_{W_1})_{|W} = (\mathcal{F}_{W_2})_{|W}$ holds, where $\mathcal{F}_{W_i}$ denotes the partial solutions in $\mathcal{F}$ with domain W_i for $i = 1, 2$.

For sake of completeness, we cite a possible implementation of this procedure given in [BK14].

Algorithm 1 (k,l)-consistency

```
1: procedure (k,l)-consistency(instance I = (V, D, C))
2:     F = all functions from at most l-element subsets of V into D;
3:     for all f ∈ F do
4:         for all ((x_1, ..., x_n), R) ∈ C do
5:             if x_1, ..., x_n ∈ dom f and (f(x_1), ..., f(x_n)) ∉ R then
6:                 F = F \ {f}
7:                 break;
8:     repeat
9:         for all f ∈ F do
10:            for all W ⊆ V of at most l elements do
11:                if (|dom f| ≤ k, dom f ⊆ W and there is no g ∈ F with dom g = W and
12:                g_{|dom f} = f) or
13:                (W ⊆ dom f and f_{|W} ∉ F) then
14:                    F = F \ {f}
15:                    break;                          ▷ proceed to the next f ∈ F
16:    until F was not altered
17:    if F = ∅ then
18:        return NO
19:    else
20:        return YES
21: end procedure
```

A reformulation of Theorem 2.3.7 could be stated as follows.

Theorem 2.3.11. *Let $\mathbb{D}$ be a finite constraint language. $\mathbb{D}$ has width (k, l) iff every instance of* CSP($\mathbb{D}$) *is correctly decided by the (k, l)-consistency algorithm, Algorithm 1.*

2.3.2 A pebble game

As an instrument to prove that the CSP over certain languages within P is not solvable by any datalog program, or equivalently, does not have bounded width, we make use of a pebble game. The version given here is based on [Bod21].

Definition 2.3.12. Let $\mathbb{D}$ be a constraint language, $I = (V, D, \mathcal{C})$ an instance of CSP($\mathbb{D}$) and $0 < k < l$ integers. The (k, l)-pebble game involves two players, namely the *Spoiler*, trying to show that the instance is unsatisfiable, and the *Duplicator*, trying to hinder Spoiler from finding a contradiction. Spoiler and Duplicator each have l pebbles. They take turns

placing them on variables of I (Spoiler cannot place two pebbles on the same variable) and elements of D, respectively. The repeated placement of pebbles corresponds to Spoiler choosing a set of variables W_i in every round $i \in \{0, 1, \dots\}$ and Duplicator choosing a partial assignment h_i from W_i to D afterwards.

The game starts in round 0 with no pebbles placed, i.e., $W_0 = \emptyset = h_0$. In round $i > 0$ Spoiler may choose a set W_i of at most l elements such that $W := W_{i-1} \cap W_i$ has at most k elements. Duplicator has to choose a mapping $h_i : W_i \to D$ such that $(h_i)_{|W} = (h_{i-1})_{|W}$. In the pebble world, this corresponds to Spoiler leaving up to k pebbles unchanged, removing all of the other pebbles and, possibly, placing new pebbles. Duplicator must not change the pebbles corresponding to unchanged pebbles of Spoiler but may put pebbles on new elements if the corresponding ones were removed or newly placed by Spoiler.

Spoiler wins the game if at some point of the game h_i is not a partial solution of I. Duplicator wins if the game continues forever.

Definition 2.3.13. Let $\mathbb{D}$ be a constraint language and let I be an instance of CSP($\mathbb{D}$). Spoiler has a *winning strategy* for the (k, l)-pebble game on I if no matter what Duplicator plays, Spoiler can choose sets of variables such that after finitely many rounds Duplicator has to pick a map that is not a partial solution of I. Duplicator has a *winning strategy* if no matter what sets of variables Spoiler chooses, Duplicator can always pick a map that is a partial solution of I on the chosen set.

Definition 2.3.14. Let $\mathbb{D}$ be a constraint language, let I be an instance of CSP($\mathbb{D}$) and let $0 < k \leq l$ be integers. A (k, l)-*consistent family* for I is a nonempty set $\mathcal{H}$ of partial solutions of I such that

(CF1) $\mathcal{H}$ is closed under restrictions of its members, and

(CF2) for every set of variables W of size at most k, every partial solution $h : W \to D$ in $\mathcal{H}$ and every superset $W' \supseteq W$ of at most l elements, there exists a partial solution $h' : W' \to D$ in $\mathcal{H}$ which extends h.

Lemma 2.3.15. *Let $\mathbb{D}$ be a constraint language, let I be an instance of* CSP($\mathbb{D}$) *and let $0 < k < l$ be integers. Duplicator has a winning strategy for the (k, l)-pebble game on I iff there exists a (k, l)-consistent family for I.*

Proof. If there exists a (k, l)-consistent family $\mathcal{H}$ for I and Duplicator chooses a mapping $h_i \in \mathcal{H}$ in the i-th round of the pebble game, Duplicator can choose a new partial assignment h_{i+1} from $\mathcal{H}$ no matter which pebbles Spoiler removes or places by definition of (k, l)-consistent families. Hence, Duplicator has a winning strategy.

For the converse implication, we assume there is a winning strategy for Duplicator. Denote $\mathcal{H}_W$ the possible choices of partial solutions $h_1 : W \to D$ of Duplicator in round 1 of the pebble game, considering the case that Spoiler had started with $W_1 = W$, such that Duplicator does have a winning strategy from there on. $\mathcal{H}_W$ is nonempty for every set of variables W of size at most l. Define

$$\mathcal{H} = \bigcup_{\substack{W \subseteq V \\ |W| \leq l}} \mathcal{H}_W.$$

We want to prove that this set is a (k,l)-consistent family. $\mathcal{H}$ contains only partial solutions and is nonempty. Disregarding some of the values of a mapping only makes the game easier for Duplicator and thus $\mathcal{H}$ is closed under restrictions. Let $h \in \mathcal{H}_W$ for some W with at most k variables and take an arbitrary $W' \supseteq W$ of size at most l. Assume that Spoiler started the game with $W_1 = W$, $W_2 = W'$ and Duplicator chose $h_1 = h$. Then $h_1 \in \mathcal{H}_W$ implies that there exists a choice h_2 for Duplicator extending h to all of W' such that Duplicator still has a winning strategy from there on, i.e., $h_2 \in \mathcal{H}_{W'} \subseteq \mathcal{H}$ holds. □

Theorem 2.3.16. *Let $\mathbb{D}$ be a finite constraint language, let I be an instance of* CSP($\mathbb{D}$) *and let $0 < k < l$ be integers. The (k,l)-consistency algorithm returns YES on input I iff Duplicator has a winning strategy in the (k,l) pebble game on I.*

Proof. If (k,l)-consistency returns YES, the algorithm yields a nonempty set of partial solutions $\mathcal{F}$ that is no longer altered in step two of the algorithm and must therefore be a (k,l)-consistent family for I.

Let $\mathcal{H}$ be a (k,l)-consistent family for I. Since none of the maps in $\mathcal{H}$ would be removed in the (k,l)-consistency algorithm, the algorithm run on I would yield YES. With Lemma 2.3.15 this concludes the proof. □

Corollary 2.3.17. *Let $0 < k < l$ be integers. A finite constraint language $\mathbb{D}$ has width (k,l) iff Spoiler wins the (k,l) pebble game on every unsatisfiable instance of* CSP($\mathbb{D}$). *A finite constraint language $\mathbb{D}$ has bounded width iff there exists some integer $l > 1$ such that Spoiler wins the $(l-1,l)$ pebble game on every unsatisfiable instance of* CSP($\mathbb{D}$).

Proof. This is an immediate consequence of Theorem 2.3.16 and Lemma 2.3.15. □

2.3.3 Relational width

As Barto describes in [Bar14], the notions of width and relational width are very similar but, depending on k and l, do not necessarily coincide. We will highlight the exact difference in Remark 2.3.35 but anticipate some information as motivation for the new definition.

A finite constraint language has bounded width iff it has bounded relational width. The notion of width (k,l) does not depend on constraints of arity higher than l, a behavior that is rather unaesthetic. Additionally, Barto shows that Corollary 4.0.6 does not hold if one substitutes width for relational width by giving a counterexample [Bar14, Example 4.7.], i.e. an instance of bounded width that does not have width $(2,3)$.

In [Bar14], Barto defines constraints with scope $W \subseteq V$ as subsets of D^W. An assignment $V \to D$ satisfies such constraint if its restriction to the scope W is an element of the constraint. To be able to simulate repetitions of variables, Barto only considers the CSPs over constraint languages containing the equality relation. If $\mathbb{D}$ is such constraint language, CSP($\mathbb{D}$) consists of all instances I such that for any constraint $C \subseteq D^W$ of I, the elements of W can be written in an ordered tuple $(x_1, \ldots, x_{|W|})$ such that the $|W|$-ary relation $\{(f(x_1), \ldots, f(x_{|W|})) \mid f \in C\}$ is a relation of $\mathbb{D}$. Using this notation, the possible values on a subset $W' \subseteq W$ of variables for any assignment f satisfying a constraint $C \subseteq D^W$ are given by the projection $C_{|_{W'}} = \{g|_{W'} \mid g \in C\}$ of the constraint onto $D^{W'}$. We opted for a different notation of CSPs but still want the benefits of translating constraints into suitable sets of partial assignments. This motivates below definition.

Definition 2.3.18. Let $C = (s, R)$ be a constraint on a domain D and W a set of variables contained in the scope s. We say that the constraint C *permits the function* $f : W \to D$ *on the set* W if there exists a solution $g : s \to D$ of C such that $g_{|W} = f$. We define

$$\mathrm{Sol}_W(C) = \{f \in D^W : C \text{ permits } f \text{ on } W\}.$$

Remark 2.3.19. Let $C = (s, R)$ be a constraint and let Z be a set of variables with $Z \subseteq W$ that is also contained in s. Then the equation

$$\mathrm{Sol}_Z(C) = (\mathrm{Sol}_W(C))_{|Z} = \{f_{|Z} : f \in \mathrm{Sol}_W(C)\}$$

holds. For any assignment f satisfying C and an arbitrary set W contained in s, we have $f_{|W} \in \mathrm{Sol}_W(C)$.

Definition 2.3.20. Let $\mathbb{D}$ be a constraint language and let $0 < k \leq l$ be natural numbers. We say an instance $I = (V, D, \mathcal{C})$ of CSP($\mathbb{D}$) is (k, l)-*minimal* if:

(M1) Every subset $W \subseteq V$ of at most l elements is contained in the scope of some constraint.

(M2) For every subset $W \subseteq V$ of at most k elements and every pair of constraints C_1 and C_2 with scopes containing W, the equation $\mathrm{Sol}_W(C_1) = \mathrm{Sol}_W(C_2)$ holds.

(M3) For every constraint (s, R) and every tuple $a \in R$, there exists a solution $f : s \to D$ of the constraint $(s, \{a\})$, i.e., $(s, \{a\})$ corresponds to a partial assignment.

A (k, k)-minimal instance is also called k-*minimal*.

Properties (M2) and (M3) together can be subsumed by the equivalent requirement (M2') which is shorter but maybe less comprehensible.

(M2') For every subset $W \subseteq V$ of at most k elements and every pair of constraints $C_1 = (s_1, R_1)$ and $C_2 = (s_2, R_2)$ with scopes containing W, we have for all a

$$a \in R_i \quad \Rightarrow \quad \exists f \in \mathrm{Sol}_W(C_j) : (s_i, \{a\}) \text{ permits } f \text{ on } W$$

whenever $\{i, j\} = \{1, 2\}$.

Remark 2.3.21. Any (k, l)-minimal instance is (k', l')-minimal for any $0 < k' \leq l'$ with $k' \leq k$ and $l' \leq l$. This is immediately clear since the conditions (M1) and (M2) are both stronger for respectively higher k and l.

In order to properly utilize the above definition, we need a procedure to transform an arbitrary instance into an associated (k, l)-minimal instance with the same set of solutions in polynomial time. Algorithm 2, which we will also call the (k, l)-*minimality algorithm*, suggests a way to do this computation. Lines 2 to 4 add constraints to the instance to ensure (M1). Lines 5 to 13 shrink the enriched instance to one satisfying (M2').

The algorithm terminates because there are only finitely many tuples to be removed, supposed that the input I is an instance of CSP($\mathbb{D}$) for a finite constraint language $\mathbb{D}$. If no tuples were removed in lines 5 to 13, the condition (M2') must already be satisfied.

Algorithm 2 (k,l)-minimality

```
1: procedure (k,l)-minimality(instance I = (V, D, C))
2:     for all W ⊆ V, |W| ≤ l do
3:         if W = {v_1, ..., v_n} not contained in any constraint C ∈ C then
4:             add ((v_1, ..., v_n), D^n) to C.
5:     repeat
6:         for all W ⊆ V, |W| ≤ k do
7:             for all C_2 ∈ C containing W do
8:                 calculate Sol_W(C_2)
9:                 for all C_1 = (s_1, R_1) ∈ C containing W do
10:                    for all a ∈ R_1 do
11:                        if the constraint (s_1, {a}) does not permit any f ∈ Sol_W(C_2) then
12:                            remove a from R_1
13:    until no tuples were removed
14:    return the modified I' = (V, D, C)
15: end procedure
```

Lemma 2.3.22. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of* CSP($\mathbb{D}$). *Algorithm 2 returns a (k,l)-minimal instance I' with the same set of solutions as the input instance I. If I is (k,l)-minimal, the algorithm returns the same instance I.*

Proof. We already argued why the algorithm returns a (k,l)-minimal instance and it is immediately clear that the algorithm does not alter a (k,l)-minimal instance.

Let f be a solution of the input instance I. We want to show that f solves the output instance I'. For $W \subseteq V$ of arbitrary size and any constraint C containing W we have $f_{|W} \in \mathrm{Sol}_W(C)$ at every stage of the algorithm. In particular lines 10 to 12 only remove tuples $a \in R_1$ such that $(s_1, \{a\})$ does not permit f restricted to subsets of s_1.

On the other hand, algorithm 2 only adds new constraints or shrinks existing ones, making the instance progressively harder to satisfy. Therefore, any solution of the output instance must also be a solution of the input instance. □

Lemma 2.3.23. *Algorithm 2 runs in polynomial time.*

Proof. The number of subsets $W \subseteq V$ of size at most l is bound by $\mathcal{O}(|V|^l)$. Hence, the runtime of the algorithm is polynomial with regard to the size of I and the exponent is determined by l. □

Lemma 2.3.24. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I, J be instances of* CSP($\mathbb{D}$). *Then $I \sqsubseteq J$ implies (k,l)-minimality$(I) \sqsubseteq (k,l)$-minimality(J).*

Proof. Lines 2 to 4 of Algorithm 2 applied to I and J yields instances I_{M1} and J_{M1} with $J \sqsubseteq J'$ again. Hence we may assume I and J to be instances satisfying property (M1) already. Write $I_{min} = (k,l)$-minimality(I). Then $I_{min} \sqsubseteq I \sqsubseteq J$ holds. Since only finitely many tuples have to be removed from J to obtain (k,l)-minimality(J), we are left to show

that whenever a tuple is removed from the constraint relation of a constraint of J, the resulting instance J' still satisfies $I_{min} \sqsubseteq J'$.

Say the tuple a is removed from the constraint relation of a constraint $C_1^{(J)} = (s_1, R_1^{(J)})$ in line 12 of Algorithm 2 because there exists a constraint $C_2^{(J)} = (s_2, R_2^{(J)})$ and $(s_1, \{a\})$ does not permit any $f \in \mathrm{Sol}_W(C_2^{(J)})$ on a set $W \subseteq s_1$. By the assumption $I_{min} \sqsubseteq J$, there exist constraints $C_1 = (s_1, R_1^{(I_{min})})$ and $C_2 = (s_2, R_2^{(I_{min})})$ of I_{min} such that $R_1^{(I_{min})} \subseteq R_1^{(J)}$ and $R_2^{(I_{min})} \subseteq R_2^{(J)}$ hold. Now, $a \in R_1^{(I_{min})}$ together with the (k,l)-minimality of I_{min} would imply that $(s_1, \{a\})$ permits a $f \in \mathrm{Sol}_W(C_2^{(I_{min})}) \subseteq \mathrm{Sol}_W(C_2^{(J)})$. Such f does not exist. Hence, we have $a \notin R_1^{(I_{min})}$ and $I_{min} \sqsubseteq J'$ follows. □

Corollary 2.3.25. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of* CSP($\mathbb{D}$). *Then (k,l)-minimality(I) is nontrivial iff there exists a nontrivial (k,l)-minimal instance I' with $I' \sqsubseteq I$.* □

Lemma 2.3.26. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of* CSP($\mathbb{D}$). *The instance (k,l)-minimality(I) does not depend on the order of removals in algorithm 2.*

Proof. Let (k,l)-minimality$_1$ and (k,l)-minimality$_2$ be two versions of Algorithm 2 starting from line 5 with possibly different but fixed orders of removals, i.e., there respectively exists a different, fixed order for the loops in lines 5 to 13. Now

$$(k,l)\text{-minimality}_1(I) = (k,l)\text{-minimality}_2 \circ (k,l)\text{-minimality}_1(I) \sqsubseteq (k,l)\text{-minimality}_2(I)$$

holds due to property (M2') of (k,l)-minimality$_1(I)$ and Lemma 2.3.24. Symmetry implies (k,l)-minimality$_2(I) \sqsubseteq (k,l)$-minimality$_1(I)$ and equality follows since both instances are (k,l)-minimal. □

Definition 2.3.27. Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of CSP($\mathbb{D}$). The instance (k,l)-minimality(I) is called the *(k,l)-minimal instance associated to I*.

Lemma 2.3.28. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of* CSP($\mathbb{D}$). *Then any constraint relation of (k,l)-minimality(I) is invariant under* Pol($\mathbb{D}$).

Proof. It suffices to prove that for any two constraints $C_1 = (s_1, R_1), C_2 = (s_2, R_2)$ with scopes containing a set W and whose constraint relations are subpowers of Pol($\mathbb{D}$), the constraint relation R_1' gained by removing tuples from R_1 in lines 5 to 13 of the (k,l)-minimality algorithm is still invariant under Pol($\mathbb{D}$). To this end, we define the biggest subset $R_1^{=} \subseteq R_1$ such that $(s_1 = (v_1, \ldots, v_n), R_1^{=})$ satisfies (M3) by

$$R_1^{=} = \{(a_1, \ldots, a_n) \in R_1 \mid \forall i, j \in \{1, \ldots, n\} : (v_i = v_j \Rightarrow a_i = a_j)\}$$

and observe that it is invariant under Pol($\mathbb{D}$). Now, $\mathrm{Sol}_{s_1}(C_1)$ interpreted as relation $\{(f(w_1, \ldots, w_k) \mid f \in \mathrm{Sol}_{s_1}(C_1)\}$, where $w_1, \ldots, w_k$ is a list of the unique variables in s_1, is

merely a projection of $R_1^=$ onto the set $\{w_1, \ldots, w_k\}$ and again an invariant relation. Similarly, $\mathrm{Sol}_{s_2}(C_2)$, the projections $\mathrm{Sol}_W(C_1) = (\mathrm{Sol}_{s_1}(C_1))_{|W}$ and $\mathrm{Sol}_W(C_2) = (\mathrm{Sol}_{s_2}(C_2))_{|W}$, and the intersection $\mathrm{Sol}_W(C_1) \cap \mathrm{Sol}_W(C_2)$ are invariant under $\mathrm{Pol}(\mathbb{D})$. Finally,

$$R_1' = \{a \in R_1^= \mid \exists f \in \mathrm{Sol}_W(C_1) \cap \mathrm{Sol}_W(C_2), (s_1, \{a\}) \text{ permits } f \text{ on } W\}$$

must be invariant under $\mathrm{Pol}(\mathbb{D})$. □

We can extend Algorithm 2 to a decision procedure described in Algorithm 3, called the (k,l)-*decision algorithm*.

Algorithm 3 (k,l)-decision

```
1: procedure (k,l)-decision(instance I = (V, D, C))
2:     I' = (k, l)-minimality(I)
3:     if I' is trivial then
4:         return No
5:     else
6:         return Yes
7: end procedure
```

If the output instance of Algorithm 2 is trivial, the original instance has no solution by Lemma 2.3.22. If the output is non-trivial, we cannot generally conclude whether or not the input instance is solvable. The (k,l)-decision algorithm with fixed parameters k and l might not be able to refute all instances that are not solvable.

Definition 2.3.29. Let $0 < k \leq l$ be integers. A finite constraint language $\mathbb{D}$ has *relational width* (k,l) if the (k,l)-decision algorithm decides every instance of $\mathrm{CSP}(\mathbb{D})$ correctly. $\mathbb{D}$ has relational width k if it has relational width (k,k). Furthermore, $\mathbb{D}$ has *bounded relational width* if there exist k and l such that $\mathbb{D}$ has relational width (k,l).

Theorem 2.3.30. *Let $\mathbb{D}$ be a finite constraint language with bounded relational width. Then $\mathrm{CSP}(\mathbb{D})$ is in P.*

Proof. Algorithm 3 runs in polynomial time. □

Remark 2.3.31. If a finite constraint language $\mathbb{D}$ has relational width (k,l) for some $0 < k \leq l$, then it also has relational width (k',l') for any $l' \geq k' > 0$ with $k' \geq k$ and $l' \geq l$. This is due to the fact that the (k',l')-decision algorithm is in general stronger than (k,l)-decision.

Definition 2.3.32. Let $0 < k \leq l$ be integers and let I be a (k,l)-minimal instance. Let W be a set of at most k variables and let C be a constraint of I containing W. Then by (M2), the set $\mathrm{Sol}_W(C)$ does not depend on the particular choice of C. We define

$$\mathrm{P}_W^{(I)} = \mathrm{Sol}_W(C).$$

We omit the instance I and abbreviate $\mathrm{P}_W^{(I)}$ by P_W if there is no risk of confusion.

We will continue with a definition almost identical to (k,l)-consistent families, a fitting notion to characterize instances that are correctly decided by the (k,l)-consistency algorithm and hence appropriate for constraint languages of width (k,l). The notion of (k,l)-*minimal family* will be suitable for the relational width (k,l) counterpart. We will illuminate the exact difference and stress the added, first requirement as a replacement for only considering partial solutions in (k,l)-consistent families.

Definition 2.3.33. Let $\mathbb{D}$ be a constraint language, $I = (V, D, \mathcal{C})$ and $0 < k \leq l$ integers. A nonempty set $\mathcal{P}$ of partial mappings from V to D is a (k,l)-*minimal family for* I if

(MF1) for every constraint $((x_1, \ldots, x_n), R)$ of I and every $h \in \mathcal{P}$ with domain W, there exists a tuple $(d_1, \ldots, d_n)$ such that $h(x_i) = d_i$ for every $i \in \{1, \ldots, n\}$ with $x_i \in W$ and such that for every set $W' \subseteq \{x_1, \ldots, x_n\}$ of size at most k, the map $h' : W' \to D$ with $h'(x_i) = d_i$ for every $i \in \{1, \ldots, n\}$ with $x_i \in W'$ is in $\mathcal{P}$,

(MF2) $\mathcal{P}$ is closed under restrictions of its members, and

(MF3) for every set of variables W of size at most k, every $h : W \to D$ in $\mathcal{P}$ and every superset $W' \supseteq W$ of at most l elements, there exists a partial mapping $h' : W' \to D$ in $\mathcal{P}$ extending h.

Lemma 2.3.34. *Let $0 < k \leq l$ be integers, let $\mathbb{D}$ be a finite constraint language and let I be an instance of* $\mathrm{CSP}(\mathbb{D})$. *The instance I has a (k,l)-minimal family iff (k,l)-*decision(I) *returns YES.*

Proof. If (k,l)-minimality(I) is nontrivial, the partial mappings

$$\mathcal{P} := \bigcup_{W \subseteq V, |W| \leq l} \mathrm{P}_W$$

form a (k,l)-minimal family.

Let $\mathcal{P}$ be a (k,l)-minimal family for I with respect to $\mathbb{D}$. Write $\mathcal{P}_W$ for those partial mappings in $\mathcal{P}$ with domain W for every subset $W \subseteq V$. We will now define a nontrivial (k,l)-minimal instance $I' \sqsubseteq I$ such that $\mathcal{P}$ is also a (k,l)-minimal family for I' with respect to $\mathbb{D}$. To that end, put a constraint (s, R) in I' for every scope s of a constraint occurring in I where the constraint relation R contains exactly those tuples witnessing the first property in the definition of (k,l)-minimal families for $\mathcal{P}$. Furthermore, we enforce (M1) and add a constraint (s, R) for every $s = (x_1, \ldots, x_l)$ such that $W = \{x_1, \ldots, x_l\}$ is a set of l variables that is not already contained in a scope of a constraint of I and define $R = \{((h(x_1), \ldots, h(x_l)) : h \in \mathcal{P}_W\}$.

Clearly, $I' \subseteq I$ holds. The missing property (M2') is an immediate consequence of the constructed constraint relations of I'. □

Remark 2.3.35. As promised, we revisit the difference of (k,l)-minimal families and (k,l)-consistent families. It was already remarked that partial solutions completely neglect constraints with scopes containing a high number of variables, see Remark 2.3.9. Property (MF1) in the definition of (k,l)-minimal families respects those constraints.

The requirement that mappings in a (k,l)-consistent family need to be partial solutions is stronger than property (MF1) of (k,l)-minimal families if we are considering scopes with $k+1$ to l elements.

If k is greater than or equal to the highest arity of any relation of the constraint language $\mathbb{D}$ in play, the notions (k,l)-minimal family and (k,l)-consistent families coincide.

Theorem 2.3.36. *A constraint language $\mathbb{D}$ has bounded width width iff it has bounded relational width.*

Proof. This is a consequence of the last observation in Remark 2.3.35, of Lemma 2.3.34 and Theorem 2.3.16. □

From now on, we will only use the notion of bounded relational width unless for historical reasons.

2.4 Constructions Preserving Bounded Relational Width

In [LZ07] Larose and Zádori use that bounded width is preserved by pp-interpretations and that varieties generated by finite bounded width algebras only contain bounded width algebras. We extend the notion bounded relational width from constraint languages to relational structures with possibly infinite signature. We then show that bounded relational width is even preserved by pp-constructions.

Definition 2.4.1. A finite relational structure $\mathbb{D}$ has *bounded relational width* if every reduct of $\mathbb{D}$ of finite type has bounded relational width.

Definition 2.4.2. A finite algebra $\mathbf{D}$ has *bounded relational width* if the relational structure $(D; \mathrm{Inv}(\mathbf{D}))$ has bounded relational width, i.e., every constraint language with base set D and whose relations are invariant under $\mathbf{D}$ has bounded relational width.

Theorem 2.4.3 ([LZ07])**.** *Let $\mathbb{D}$ be a finite constraint language with bounded width. Then any constraint language with the same base set and relations that are pp-definable from $\mathbb{D}$ has bounded width.*

Theorem 2.4.4 ([LZ07])**.** *Every finite algebra in the variety generated by a bounded width algebra has bounded width.*

By Theorem 2.1.34, a finite relational structure $\mathbb{E}$ is pp-constructible from another finite relational structure $\mathbb{D}$ iff it is homomorphically equivalent to a pp-power of this structure. We will continue by showing that those two constructions preserve bounded relational width.

Lemma 2.4.5. *Let $0 < k \leq l$ be integers. Let $\mathbb{D}$ and $\mathbb{E}$ be homomorphically equivalent finite constraint languages. Then $\mathbb{D}$ has relational width (k,l) iff $\mathbb{E}$ has relational width (k,l).*

Proof. Assume $\mathbb{D}$ has relational width (k,l). Let $I_E = (V, E, \mathcal{C})$ be an instance of CSP($\mathbb{E}$) and $g : \mathbb{D} \to \mathbb{E}$ and $h : \mathbb{E} \to \mathbb{D}$ homomorphisms. We have to prove that I_E has a solution, provided the associated (k,l)-minimal instance $I'_E := (k,l)$-minimality(I_E) is nontrivial. If I'_E is nontrivial, the associated (k,l)-minimal instance $I'_D := (k,l)$-minimality(I_D) of $I_D := (V, D, \mathcal{C})$ is nontrivial. Indeed, the tuple $(h(a_1), \ldots, h(a_n))$ is not removed from a constraint relation during the calculation of (k,l)-minimality(I_D) if the tuple $(a_1, \ldots, a_n)$ is not removed in the same step of the calculation of (k,l)-minimality(I_E). Since $\mathbb{D}$ has relational width (k,l), the non-triviality of I'_D implies the existence of a solution f of I_D. Thus $g \circ f$ is a solution of I_E and $\mathbb{E}$ has relational width (k,l). Due to symmetry the converse implication holds as well. □

Lemma 2.4.6. *Let $0 < k \leq l$ be integers. A finite constraint language has relational width (k,l) iff its core has relational width (k,l).*

Proof. Lemma 2.4.5. □

The next lemma is an adaptation of a result from [LZ07]. We use relational width instead of width and slightly different terminology.

Lemma 2.4.7. *Let $\mathbb{D}$ be a finite constraint language with bounded relational width. Every constraint language $\mathbb{E}$ with the same domain as $\mathbb{D}$ and whose relations are pp-definable over $\mathbb{D}$ has bounded relational width.*

Proof. Let $0 < k \leq l$ be integers and let $\mathbb{D}$ be a finite constraint language with relational width (k,l). A constraint language $\mathbb{E}$ that has only relations which are pp-definable from the relations of $\mathbb{D}$ can be obtained from $\mathbb{D}$ by repeating the constructions listed below finitely many times.

(1) removing a relation from $\mathbb{D}$,

(2) adding a a relation obtained by permuting the components of a relation to $\mathbb{D}$,

(3) adding the intersection of two relations of the same arity to $\mathbb{D}$,

(4) adding the product of two relations to $\mathbb{D}$,

(5) adding the equality relation to $\mathbb{D}$,

(6) adding a relation to $\mathbb{D}$ obtained by projecting an n-ary relation to its first $n-1$ components.

Therefore, it suffices to prove that any constraint language $\mathbb{E}$ obtained by $\mathbb{D}$ through any one of these steps still has bounded relational width.

Case 1: Let $\mathbb{E}$ be the structure obtained from $\mathbb{D}$ by removing a relation. Any instance I of CSP($\mathbb{E}$) is an instance of CSP($\mathbb{D}$). Therefore, the (k,l)-decision algorithm correctly decides I.

Case 2: Let $\mathbb{E}$ be the structure obtained from $\mathbb{D}$ by adding a relation

$$R^{\mathbb{E}} := \{(a_{\pi(1)}, \ldots, a_{\pi(n)}) : (a_1, \ldots, a_n) \in R^{\mathbb{D}}\}$$

where $R^{\mathbb{D}}$ is a n-ary relation of $\mathbb{D}$ and π is a permutation of $\{1, \ldots, n\}$. We can transform an instance $I_{\mathbb{E}}$ of CSP($\mathbb{E}$) into an instance $I_{\mathbb{D}}$ of CSP($\mathbb{D}$) with the same set of solutions by replacing constraints of the form $C_{\mathbb{E}} = ((v_1, \ldots, v_n), R^{\mathbb{E}})$ of $I_{\mathbb{E}}$ with corresponding constraints $C_{\mathbb{D}} = ((v_{\pi^{-1}(1)}, \ldots, v_{\pi^{-1}(n)}), R^{\mathbb{D}})$ in $I_{\mathbb{D}}$ and keeping all other constraints unchanged. The (k, l)-decision algorithm correctly decides $I_{\mathbb{D}}$ and since $\mathrm{Sol}_W(C_{\mathbb{E}}) = \mathrm{Sol}_W(C_{\mathbb{D}})$ holds for every set W contained in the scopes of both constraints, the algorithm also correctly decides $I_{\mathbb{E}}$.

Case 3: Let $\mathbb{E}$ be the structure obtained from $\mathbb{D}$ by adding the intersection $R^{\mathbb{E}} := R_1^{\mathbb{D}} \cap R_2^{\mathbb{D}}$ of two relations $R_1^{\mathbb{D}}, R_2^{\mathbb{D}}$ of $\mathbb{D}$ with the same arity. Any instance $I_{\mathbb{E}}$ of CSP($\mathbb{E}$) can be transformed into an instance $I_{\mathbb{D}}$ of CSP($\mathbb{D}$) with the same set of solutions by replacing every constraint of the form $(s, R^{\mathbb{E}})$ of $I_{\mathbb{E}}$ by two constraints $(s, R_1^{\mathbb{D}})$ and $(s, R_2^{\mathbb{D}})$, and keeping all other constraints unchanged. If $I_{\mathbb{E}}$ has no solutions, so does $I_{\mathbb{D}}$ and the (k, l)-decision algorithm refutes $I_{\mathbb{D}}$. Since $I_{\mathbb{E}} \sqsubseteq I_{\mathbb{D}}$ holds, the (k, l)-decision algorithm also refutes $I_{\mathbb{E}}$ by Lemma 2.3.24.

Case 4: Let $\mathbb{E}$ be the structure obtained by adding the product

$$R^{\mathbb{E}} = \{(a_1, \ldots, a_{m+n}) : (a_1, \ldots, a_m) \in R_1^{\mathbb{D}}, (a_{m+1}, \ldots, a_{m+n}) \in R_2^{\mathbb{D}}\}$$

of two relations $R_1^{\mathbb{D}}, R_2^{\mathbb{D}}$ of $\mathbb{D}$ to $\mathbb{D}$.

Assume $I_{\mathbb{E}}$ is an instance of CSP($\mathbb{E}$) that has a nontrivial associated (k, l)-minimal instance (k, l)-minimality($I_{\mathbb{E}}$). We define an instance $I_{\mathbb{D}}$ of CSP($\mathbb{D}$) with the same set of solutions as $I_{\mathbb{E}}$ by replacing every constraint of the form $C = (s, R^{\mathbb{E}})$ in $I_{\mathbb{E}}$ by two constraints $C^{(1)} = (s_1, R_1^{\mathbb{D}})$ and $C^{(2)} = (s_2, R_2^{\mathbb{D}})$, where s_1 is the m-tuple consisting of the first m entries of s and s_2 is the n-tuple consisting of the last n entries of s, and keeping all other constraints unchanged. Furthermore, define $J_{\mathbb{E}}$ to be the instance of CSP($\mathbb{E}$) which contains $C^{(1)}$ and $C^{(2)}$, defined as above, in addition to all the constraints of $I_{\mathbb{E}}$ and observe that $J_{\mathbb{E}} \sqsubseteq I_{\mathbb{D}}$ holds. The proof is complete by Lemma 2.3.24 if we can show that adding of $C^{(1)}$ and $C^{(2)}$ to $I_{\mathbb{E}}$ is negligible for the outcome of the (k, l)-decision algorithm, i.e., (k, l)-minimality($J_{\mathbb{E}}$) is still nontrivial.

By Lemma 2.3.26, we may assume that during the execution of the (k, l)-minimality algorithm every comparison in lines 5 to 13 of constraints $C_1 = C^{(i)}$ and an arbitrary constraint C_2, or an arbitrary constraint C_1 and $C_2 = C^{(j)}$ with $i, j \in \{1, 2\}$ is respectively subsequent to the comparison of $C_1 = C$ and C_2, or C_1 and $C_2 = C$. For any tuple a that would be removed from the constraint relation of C_1 in the case $C_1 = C^{(i)}$, the algorithm removes all tuples starting with a from the constraint relation of C while comparing $C_1 = C$ and C_2 if i equals 1, and all tuples ending with a if i equals 2. Thus, we have $\mathrm{Sol}_W(C) \subseteq \mathrm{Sol}_W(C^{(i)})$ for $i \in \{1, 2\}$ and all sets W of at most k elements contained in $C^{(i)}$ throughout the execution of the algorithm. Now, consider the case that constraints C_1 and $C_2 = C^{(i)}$ are compared during execution of the algorithm. Let a be a tuple that would theoretically be removed from the constraint relation of C_1 in line 12 of the (k, l)-minimality algorithm

because $(s, \{a\})$, where s is the scope of C_1, does not permit any $f \in \mathrm{Sol}_W(C^{(i)})$. Then $(s, \{a\})$ also does not permit any $f \in \mathrm{Sol}_W(C)$ and hence a is not in the constraint relation of C_1 or was removed earlier, e.g., when considering the constraints C_1 and C. Hence (k, l)-minimality$(J_{\mathbb{E}})$ is still nontrivial.

Case 5: Let $\mathbb{E}$ be the structure obtained from $\mathbb{D}$ by adding the equality relation. We will prove that $\mathbb{E}$ has relational width $(2k, 2l)$ under the additional assumption $l > 1$. Let $I_{\mathbb{E}} = (V, D, \mathcal{C})$ be an instance of CSP$(\mathbb{E})$ that has a nontrivial associated $(2k, 2l)$-minimal instance $J_{\mathbb{E}} := (2k, 2l)$-minimality$(I_{\mathbb{E}})$. We define θ to be the reflexive, symmetric and transitive closure of the binary relation $\{(x, y) \in V^2 \mid ((x, y), =) \in \mathcal{C}\}$ on the variables of $I_{\mathbb{E}}$. Write $[v]_\theta$ for the equivalence class of a variable v with respect to θ, and $V_{/\theta}$ for the set of equivalence classes of V. If $s = (v_1, \ldots, v_n)$ is a scope and $C = (s, R)$ a constraint, we use the notations

$$s^{(\theta)} = ([v_1]_\theta, \ldots, [v_n]_\theta) \quad \text{and} \quad C^{(\theta)} = (s^{(\theta)}, R).$$

We transform $I_{\mathbb{E}}$ into an instance $I_{\mathbb{D}}$ of CSP$(\mathbb{D})$ with variables $V_{/\theta}$, domain D, and a constraint $C^{(\theta)}$ for every constraint C of $I_{\mathbb{E}}$ that has a constraint relation different from equality. Similarly, we transform $J_{\mathbb{E}}$ into an instance $J_{\mathbb{D}}$ of CSP$(\mathbb{D})$.

Clearly $J_{\mathbb{E}} \sqsubseteq I_{\mathbb{E}}$ holds and this implies $J_{\mathbb{D}} \sqsubseteq I_{\mathbb{D}}$. By definition, $I_{\mathbb{E}}$ has a solution iff $I_{\mathbb{D}}$ has a solution. Furthermore, $J_{\mathbb{D}}$ is surely nontrivial if $J_{\mathbb{E}}$ contains any constraint with a constraint relation different from equality, e.g., a ternary relation, and the additional assumption $l > 1$ yields the existence of such constraint. Thus it suffices to show that $J_{\mathbb{D}}$ is (k, l) minimal by Corollary 2.3.25.

In order to prove (M1), (M2) and (M3) of $J_{\mathbb{D}}$, we firstly claim that for any variables $x, y \in V$ and any constraint C in $J_{\mathbb{E}}$ containing $\{x, y\}$, the implication

$$(x, y) \in \theta \quad \Rightarrow \quad \forall f \in \mathrm{Sol}_{\{x,y\}}(C) : (f(x) = f(y)) \tag{2.3}$$

holds. Indeed, $(x, y) \in \theta$ implies the existence of an $r > 0$ and variables $x = v_0, \ldots, v_r = y$ such that either $v_{i-1} = v_i$ or $v_i = v_{i-1}$ is a constraint of $I_{\mathbb{E}}$ for every $i \in \{1, \ldots, r\}$. The claim now follows by the $(2, 3)$-minimality of $J_{\mathbb{E}}$.

Property (M1) of $J_{\mathbb{D}}$ is an immediate consequence of the same attribute of $J_{\mathbb{E}}$. If C is a constraint of $J_{\mathbb{E}}$ and g a solution of C, the function $g^{(\theta)} : V_{/\theta} \to D, [v]_\theta \mapsto g(v)$ is well-defined and a solution of $C^{(\theta)}$ by (2.3). Hence, (M3) holds for $J_{\mathbb{D}}$. In order to prove (M2), we assume $W^\theta \subseteq V_{/\theta}$ to be a set of at most k classes of variables, and ${C_1}^{(\theta)}, {C_2}^{(\theta)}$ to be constraints of $J_{\mathbb{D}}$ with scopes ${s_1}^{(\theta)}, {s_2}^{(\theta)}$ containing W^θ. We choose constraints $C_1, C_2 \in J_{\mathbb{E}}$ with scopes s_1, s_2 that have representatives of the variables of ${s_1}^{(\theta)}, {s_2}^{(\theta)}$ in corresponding positions. Furthermore, we choose a set $W_1 \subseteq V$ of at most k variables that is contained in s_1 and contains at least one representative of every class $[v]_\theta$ in W^θ. Such set exists by definition of ${s_1}^{(\theta)}$. Analogously, we can pick a set W_2 of at most k variables that is contained in s_2 and contains at least one representative of every class $[v]_\theta$ in W^θ.

If ${f_1}^{(\theta)} \in \mathrm{Sol}_{W^\theta}({C_1}^{(\theta)})$, there exists a solution $g^{(\theta)} : {s_1}^{(\theta)} \to D$ of ${C_1}^{(\theta)}$ such that ${g^{(\theta)}}|_{W^\theta} = {f_1}^{(\theta)}$ holds. By definition, the map $g_1 : s_1 \to D, v \to g^{(\theta)}([v]_\theta)$ solves C_1 and

hence $f_1 := g_{1|_{W_1}} \in \mathrm{Sol}_{W_1}(C_1)$ follows. Since $W_1 \cup W_2$ has at most $2k$ elements, (M1) implies the existence of a constraint C_3 that has a scope s_3 which contains $W_1 \cup W_2$. Now, (M2) yields $f_1 \in \mathrm{Sol}_{W_1}(C_3)$. Let $g : s_3 \to D$ be a solution of C_3 extending f_1. Property (M2) implies $f_2 := g_{|_{W_2}} \in \mathrm{Sol}_{W_2}(C_2)$ and we choose a solution $g_2 : s_2 \to D$ of C_2 with $g_{2|_{W_2}} = f_2$. By (2.3), the function $g_2{}^{(\theta)} : s_2{}^{(\theta)} \to D, [v]_\theta \mapsto g_2(v)$ is well-defined, and it solves $C_2{}^{(\theta)}$. Therefore, $f_2{}^{(\theta)} := g_2{}^{(\theta)}{}_{|_{W^\theta}} \in \mathrm{Sol}_{W^\theta}(C_2{}^{(\theta)})$ holds and (2.3) applied to the constraint C_3 and the function g implies $f_1(x) = f_2(y)$ for all $x \in W_1$, $y \in W_2$ with $(x, y) \in \theta$. Hence, $f_1{}^{(\theta)} = f_2{}^{(\theta)}$ holds.

Case 6: Let $\mathbb{E}$ be the structure obtained from $\mathbb{D}$ by adding the projection

$$R_p^{\mathbb{E}} = \{(a_1, \ldots, a_{n-1}) \mid \exists a_n \in D,\ (a_1, \ldots, a_n) \in R_p^{\mathbb{D}}\}$$

of an n-ary relation $R_p^{\mathbb{D}}$ onto the first $n-1$ variables. Assume l to be greater or equal than the arity of any relation of $\mathbb{D}$ and that $\mathbb{D}$ has relational width (l, l). We want to show that $\mathbb{E}$ has relational width (l^2, l^2).

Let $I_{\mathbb{E}} = (V, D, \mathcal{C}^{\mathbb{E}})$ be an instance of CSP($\mathbb{E}$) such that $J_{\mathbb{E}} = (l^2, l^2)$-minimality$(I_{\mathbb{D}})$ is nontrivial. Denote the set of constraints of $I_{\mathbb{E}}$ that have the constraint relation $R_p^{\mathbb{E}}$ by $\mathcal{C}_p^{\mathbb{E}}$. We define the instance $I_{\mathbb{D}} = (V_{\mathbb{D}}, D, \mathcal{C}^{\mathbb{D}})$ of CSP($\mathbb{D}$) by replacing every constraint $C = ((v_1, \ldots, v_{n-1}), R_p^{\mathbb{E}}) \in \mathcal{C}_p^{\mathbb{E}}$ by a constraint $((v_1, \ldots, v_{n-1}, z_C), R_p^{\mathbb{D}})$, where z_C is a new unique variable not occurring in any constraint of $I_{\mathbb{E}}$, and keeping all other constraints of $I_{\mathbb{E}}$. The set of variables $V_{\mathbb{D}}$ of $I_{\mathbb{D}}$ consists of these variables z_C for every constraint $C \in \mathcal{C}^{\mathbb{E}}$ together with the variables V of $I_{\mathbb{E}}$. For every new variable $z \in V_{\mathbb{D}} \setminus V$, denote the unique constraint of $I_{\mathbb{D}}$ which contains z by

$$C_z = ((v_1^{(z)}, \ldots, v_{n-1}^{(z)}, z), R_p^{\mathbb{D}}).$$

The instance $I_{\mathbb{E}}$ has a solution iff $I_{\mathbb{D}}$ has a solution. Hence, it is sufficient to show that (l, l)-minimality$(I_{\mathbb{D}})$ is also nontrivial. We write $I'_{\mathbb{D}}$ for the instance obtained from $I_{\mathbb{D}}$ after executing lines 2 to 4 of the (l, l)-minimality algorithm, i.e., the instance $I'_{\mathbb{D}}$ satisfies (M1).

In the remaining part of the proof, for every subset $W \subseteq V_{\mathbb{D}}$ with at most l elements, we construct a nonempty class Ψ_W of functions from W to D such that

$$\Psi_W \subseteq \mathrm{Sol}_W(C)$$

holds at any stage of the algorithm for every constraint C of $I'_{\mathbb{D}}$ containing W. If such classes Ψ_W exist, (l, l)-minimality$(I_{\mathbb{D}})$ is clearly nontrivial.

Let $W_{\mathbb{D}} \subseteq V_{\mathbb{D}}$ be a set of at most l variables. Since l is greater or equal than the arity of any relation of $\mathbb{D}$, the set

$$W_{\mathbb{E}} := (W_{\mathbb{D}} \cap V) \cup \bigcup_{z \in W_{\mathbb{D}} \setminus V} \{v_1^{(z)}, \ldots, v_{n-1}^{(z)}\} \tag{2.4}$$

has at most l^2 elements. Hence there exists a constraint C in $J_{\mathbb{E}}$ containing $W_{\mathbb{E}}$. Note that $\mathrm{Sol}_{W_{\mathbb{E}}}(C)$ is nonempty and does not depend on the choice of C. Hence, we can define the set $\Psi_{W_{\mathbb{D}}}$ consisting of all functions $\psi : W_{\mathbb{D}} \to D$ such that there exists a $h \in \mathrm{Sol}_{W_{\mathbb{E}}}(C)$ with

(i) $\psi_{|_{W_\mathbb{D} \cap V}} = h_{|_{W_\mathbb{D} \cap V}}$, and

(ii) $\forall z \in W_\mathbb{D} \setminus V$, the function $h_{|_{\{v_1^{(z)}, \dots, v_{n-1}^{(z)}\}}} \cup \psi_{|_{\{z\}}}$ solves C_z.

$\Psi_{W_\mathbb{D}}$ is nonempty because $\mathrm{Sol}_{W_\mathbb{E}}(C)$ is nonempty and because $(h(v_1^{(z)}), \dots, h(v_{n-1}^{(z)})) \in R_p^\mathbb{E}$ implies that we can extend ψ to z such that $(h(v_1^{(z)}), \dots, h(v_{n-1}^{(z)}), \psi(z)) \in R_p^\mathbb{D}$ holds by definition of the relation $R_p^\mathbb{E}$.

Additionally, we claim that for any sets $W_\mathbb{D}^{(1)} \subseteq W_\mathbb{D}^{(2)}$ of at most l variables

$$\Psi_{W_\mathbb{D}^{(1)}} = \left(\Psi_{W_\mathbb{D}^{(2)}}\right)_{|_{W_\mathbb{D}^{(1)}}} := \{\psi_{|_{W_\mathbb{D}^{(1)}}} \mid \psi \in \Psi_{W_\mathbb{D}^{(2)}}\} \tag{2.5}$$

holds. Indeed, by construction of $\Psi_{W_\mathbb{D}^{(1)}}$ and $\Psi_{W_\mathbb{D}^{(2)}}$, the right side of the above equation must be contained in $\Psi_{W_\mathbb{D}^{(1)}}$. Let $W_\mathbb{E}^{(1)}$ and $W_\mathbb{E}^{(2)}$ respectively be the sets defined as in (2.4) for $W_\mathbb{D}^{(1)}$ and $W_\mathbb{D}^{(2)}$, and let C be a constraint of $J_\mathbb{E}$ containing $W_\mathbb{E}^{(2)} \supseteq W_\mathbb{E}^{(1)}$. Now, choose a $\psi_1 \in \Psi_{W_\mathbb{D}^{(1)}}$, then there exists a $h_1 \in \mathrm{Sol}_{W_\mathbb{E}^{(1)}}(C)$ such that ψ_1 is an extension of $h_{1|_{W_\mathbb{D}^{(1)} \cap V}}$. By definition, there exists a $h_2 \in \mathrm{Sol}_{W_\mathbb{E}^{(2)}}(C)$ with $h_{2|_{W_\mathbb{E}^{(1)}}} = h_1$. We extend $h_{2|_{W_\mathbb{D}^{(2)} \cap V}}$ to a function ψ_2 by

$$\psi_2(z) = \begin{cases} h_2(z), & \text{for } z \in W_\mathbb{D}^{(2)} \cap V, \\ \psi_1(z), & \text{for } z \in W_\mathbb{D}^{(1)} \setminus V, \\ \text{any } d \text{ with } (h(v_1^{(z)}), \dots, h(v_{n-1}^{(z)}), d) \in R_p^\mathbb{D}, & \text{for } z \in W_\mathbb{D}^{(2)} \setminus (W_\mathbb{D}^{(1)} \cup V). \end{cases}$$

The obtained ψ_2 is in $\Psi_{W_\mathbb{D}^{(2)}}$, extends ψ_1, and hence (2.5) follows.

Let $W \subseteq V_\mathbb{D}$ be a set of at most l variables and let C be a constraint of $I'_\mathbb{D}$ which contains W. Then C has one of the three forms $C_z = ((v_1^{(z)}, \dots, v_{n-1}^{(z)}, z), R_p^\mathbb{D})$, or (s, D^t) where t is the arity of s, or (s, R) with s contained in V and it is already a constraint of $I_\mathbb{D}$. For either one of those sorts, $\Psi_W \subseteq \mathrm{Sol}_W(C)$ follows easily.

We conclude by proofing that $\Psi_W \subseteq \mathrm{Sol}_W(C)$ holds even after termination of the (l,l)-minimality algorithm by induction on the loops in lines 5 to 13. Let $W' \subseteq V_\mathbb{D}$ be a set of at most l variables and $C_1 = (s_1, R_1), C_2 = (s_2, R_2)$ constraints with scopes containing W'. Assume that the tuple a will be removed from R_1 in line 12, i.e., $(s_1, \{a\})$ does not permit any $f \in \mathrm{Sol}_{W'}(C_2)$.

We want to show that $\Psi_W \subseteq \mathrm{Sol}_W((s_1, R_1 \setminus \{a\}))$ still holds for every set W contained in s_1 with at most l elements. To that end, fix such set W and choose a $\psi \in \Psi_W$. By (2.5) and the assumption that l is greater or equal than the arity of any relation in $\mathbb{D}$, there exists a $g \in \Psi_{s_1}$ with $g_{|_W} = \psi$. The induction hypothesis implies $g \in \mathrm{Sol}_{s_1}(C_1)$, i.e., $b = (\psi(v_1), \dots, \psi(v_t)) \in R_1$, where $s_1 = (v_1, \dots, v_t)$. Notice, that (2.5) and the induction hypothesis yield $g_{|_{W'}} \in \Psi_{W'} \subseteq \mathrm{Sol}_{W'}(C_2)$. Hence b is not removed from R_1 as $(s_1, \{b\})$ clearly permits $g_{|_{W'}}$ and we have $\psi \in \mathrm{Sol}_W((s_1, R_1 \setminus \{a\}))$. □

Lemma 2.4.8. *Let $\mathbb{D}$ be a finite constraint language. If $\mathbb{D}$ has bounded relational width, then any pp-power of $\mathbb{D}$ has bounded relational width.*

Proof. Let $\mathbb{E}'$ be a pp-power of the finite constraint language $\mathbb{D}$ with bounded relational width. Then there exists an integer $n \geq 1$ such that $\mathbb{E}'$ is isomorphic to a constraint language $\mathbb{E}$ with domain D^n whose relations are pp-definable over $\mathbb{D}$. It suffices to prove that $\mathbb{E}$ has bounded relational width. From now on, if $R^{\mathbb{E}}$ is a t-ary relation of $\mathbb{E}$, we write $R^{\mathbb{D}'}$ for its corresponding tn-ary relation, i.e., with the abbreviation $b_i = (a_1^{(i)}, \dots, a_n^{(i)}) \in D^n$, we have

$$(b_1, \dots, b_t) \in R^{\mathbb{E}} \iff (a_1^{(1)}, \dots, a_n^{(1)}, \dots, a_1^{(t)}, \dots, a_n^{(t)}) \in R^{\mathbb{D}'}.$$

The constraint language $\mathbb{D}'$ obtained by adding all the relations $R^{\mathbb{D}'}$ for every relation $R^{\mathbb{E}}$ to $\mathbb{D}$ has bounded relational width by Lemma 2.4.7, say it has relational width (k, l).

Let $I_{\mathbb{E}} = (V_{\mathbb{E}}, D^n, \mathcal{C}_E)$ be an instance of CSP($\mathbb{E}$) with the variables $V_{\mathbb{E}} = \{v^{(1)}, \dots, v^{(m)}\}$ such that its associated (k, l)-minimal instance $J_{\mathbb{E}} := (k, l)\text{-minimality}(I_{\mathbb{E}})$ is nontrivial. Finding a solution of $I_{\mathbb{E}}$ would imply that $\mathbb{E}$ has relational width (k, l).

To that end, we translate the instances $I_{\mathbb{E}}$ and $J_{\mathbb{E}}$ to instances $I_{\mathbb{D}'}$ and $J_{\mathbb{D}'}$. Both shall have variables

$$V_{\mathbb{D}'} := \{v_1^{(1)}, \dots, v_n^{(1)}, \dots, v_1^{(m)}, \dots, v_n^{(m)}\}$$

and the domain D. For every constraint $C^{\mathbb{E}} = ((v^{(i_1)}, \dots, v^{(i_t)}), R^{\mathbb{E}})$ of $I_{\mathbb{E}}$, we add a constraint

$$C^{\mathbb{D}'} := ((v_1^{(i_1)}, \dots, v_n^{(i_1)}, \dots, v_1^{(i_t)}, \dots, v_n^{(i_t)}), R^{\mathbb{D}'})$$

to $I_{\mathbb{D}'}$. Analogously, we add a constraint $C^{\mathbb{D}'}$ to $J_{\mathbb{D}'}$ for every constraint in $J_{\mathbb{E}}$. We have $J_{\mathbb{E}} \sqsubseteq I_{\mathbb{E}}$ which implies $J_{\mathbb{D}'} \sqsubseteq I_{\mathbb{D}'}$. If f is a solution of $I_{\mathbb{D}'}$, then $v^{(i)} \mapsto (f(v_1^{(i)}), \dots, f(v_n^{(i)}))$ is a solution of $I_{\mathbb{E}}$. By Corollary 2.3.25, it now suffices to show that $J_{\mathbb{D}'}$ is a nontrivial (k, l)-minimal instance.

The instance $J_{\mathbb{D}'}$ is clearly nontrivial and (M1) is an immediate consequence of (M1) for $J_{\mathbb{E}}$. Let

$$W_{\mathbb{D}'} = \{v_{j_1}^{(i_1)}, \dots, v_{j_k}^{(i_k)}\} \subseteq V_{\mathbb{D}'}$$

be a set of at most k variables, let $C_1^{\mathbb{D}'} = (s_1^{\mathbb{D}'}, R_1^{\mathbb{D}'})$ and $C_2^{\mathbb{D}'}$ be two constraints of $J_{\mathbb{D}'}$ containing $W_{\mathbb{D}'}$, and let

$$a = (a_1^{(1)}, \dots, a_n^{(1)}, \dots, a_1^{(t)}, \dots, a_n^{(t)}) \in R_1^{\mathbb{D}'}.$$

By construction, the constraints $C_1^{\mathbb{E}} = (s_1^{\mathbb{E}}, R_1^{\mathbb{E}})$ and $C_2^{\mathbb{E}} = (s_2^{\mathbb{E}}, R_2^{\mathbb{E}})$ both contain the set

$$W_{\mathbb{E}} = \{v^{(i_1)}, \dots, v^{(i_k)}\}.$$

With $b_i = (a_1^{(i)}, \dots, a_n^{(i)})$ for $i \in \{1, \dots, t\}$, we have $(b_1, \dots, b_t) \in R_1^{\mathbb{E}}$. By the (k, l)-minimality of $J_{\mathbb{E}}$, there exists a map $f_{\mathbb{E}} \in \mathrm{Sol}_{W_{\mathbb{E}}}(C_2^{\mathbb{E}})$ such that $(s_1^{\mathbb{E}}, \{b_1, \dots, b_t\})$ permits $f_{\mathbb{E}}$ on $W_{\mathbb{E}}$. Hence there exists a solution $g_{\mathbb{E}}$ of $C_2^{\mathbb{E}}$. For $q \in \{1, \dots, k\}$, we define

$$f_{\mathbb{D}'}(v_{j_q}^{(i_q)}) = (f_{\mathbb{E}}(v^{(i_q)}))_{j_q},$$

i.e., $f_{\mathbb{D}'}(v_{j_q}^{(i_q)})$ is the j_q-th component of the n-tuple $f_{\mathbb{E}}(v^{(i_q)})$. Analogously, set

$$g_{\mathbb{D}'}(v_j^{(i)}) = (g_{\mathbb{E}}(v^{(i)}))_j$$

for i, j such that $v_j^{(i)}$ is contained in the scope $s_2^{\mathbb{E}}$. Then $f_{\mathbb{D}'} \in \mathrm{Sol}_{W_{\mathbb{D}'}}(C_2^{\mathbb{D}'})$ follows since $g_{\mathbb{D}'}$ solves $C_1^{\mathbb{D}'}$. Finally, $(s_1^{\mathbb{D}'}, \{a\})$ permits $f_{\mathbb{D}'}$ on $W_{\mathbb{D}'}$ and symmetry yields (M2') for $J_{\mathbb{D}'}$. □

Theorem 2.4.9. *The class of finite constraint languages with bounded relational width is closed under pp-constructions.*

Proof. We proved that said class is closed under homomorphic equivalence and building pp-powers in Lemma 2.4.5 and Lemma 2.4.8. Hence, Theorem 2.1.34 yields the result. □

Corollary 2.4.10. *The class of finite relational structures with bounded relational width is closed under pp-constructions.*

Proof. Let $\mathbb{D}$ be a relational structure with bounded relational width and let $\mathbb{E}$ be pp-constructible from $\mathbb{D}$. Choose an arbitrary reduct $\mathbb{E}'$ of $\mathbb{E}$ that has finite type. It is sufficient to show that $\mathbb{E}'$ is pp-constructible from a reduct $\mathbb{D}'$ of $\mathbb{D}$ of finite type. To that end, by Theorem 2.1.34, we may consider a relational structure $\mathbb{C}$ which is homomorphically equivalent to $\mathbb{E}$ and a pp-power of $\mathbb{D}$. The reduct $\mathbb{C}'$ of $\mathbb{C}$ to the finite language of $\mathbb{E}'$ is homomorphically equivalent to $\mathbb{E}'$ and still a pp-power of $\mathbb{D}$. However, there are only finitely many relations of $\mathbb{D}$ needed to pp-define all of the finitely many relations of $\mathbb{C}'$. □

Corollary 2.4.11. *Let* $\mathbf{D}$, $\mathbf{E}$ *be finite algebras such that* $\mathrm{Clo}(\mathbf{E}) \in \mathrm{ERP}_{\mathrm{fin}}(\mathrm{Clo}(\mathbf{D}))$, *or equivalently,* $\mathrm{Clo}(\mathbf{D}) \leq_{h1} \mathrm{Clo}(\mathbf{E})$ *holds. If* $\mathbf{D}$ *has bounded relational width,* $\mathbf{E}$ *also has bounded relational width.*

Proof. This is an immediate consequence of Corollary 2.4.10 and Theorem 2.1.44. □

Corollary 2.4.12. *A finite relational structure* $\mathbb{D}$ *that is a core has bounded relational width iff the singleton expansion of* $\mathbb{D}$ *has bounded relational width.*

Proof. The singleton expansion of a finite core $\mathbb{D}$ and $\mathbb{D}$ are pp-constructible from one another. The result follows with Corollary 2.4.10. □

3 Necessary Conditions for Bounded Relational Width

The aim of this chapter is to find obstructions which hinder a constraint language from having bounded relational width.

3.1 Solving Linear Equations

We give an example of a constraint language which has a tractable CSP but does not have bounded relational width. This section is heavily based on [Bod21, Theorem 8.6.11]. A result, similar to the one from Bodirsky which is presented below, was originally stated by [FV98] using the notion of ability to count. The CSP over the constraint language which is defined in the next theorem may be interpreted as solving linear equations. This sets a well-known obstruction to bounded relational width. Any constraint language that pp-constructs the language below cannot have bounded relational width.

Theorem 3.1.1. *Let $(G; +, 0, -)$ be a finite abelian group and $a \in G \setminus \{0\}$. Define the relations*

$$R_c^k := \{(x_1, \dots, x_k) \in G^k \mid x_1 + \cdots + x_k = c\}$$

for any $c \in G$. Then $\mathbb{G} = (G; \{0\}, R_0^3, R_a^3)$ does not have bounded relational width.

Definition 3.1.2. Let $\mathbb{D}$ be a relational structure and $I = (V, D, \mathcal{C})$ be an instance of CSP($\mathbb{D}$). The *incidence graph* $\mathcal{G}(I)$ is the bipartite graph which has as vertex set the disjoint union of all variables V of I and all scopes of constraints of I, and (a, b) is an edge of $\mathcal{G}(I)$ iff a is a variable and b a tuple containing a or vice versa.

Moreover, we write $\mathcal{G}[S]$ for the incidence graph $\mathcal{G}(I[S])$, where $I[S]$ is the instance with the variables S, and $I[S]$ contains contains exactly those constraints of I with scopes fully contained in S.

Definition 3.1.3. The *girth* of an undirected graph is the length of its shortest cycle. We say that an instance I has *girth* k if all scopes of constraints of I consist of pairwise distinct variables and the bipartite graph $\mathcal{G}(I)$ has girth $2k$.

Definition 3.1.4. A graph is called *k-regular* if every vertex of the graph has precisely k neighbours. A 3-regular graph is also called *cubic*.

The next lemma will be the core of our proof that solving linear equations does not have bounded relational width. The proof formalizes the intuition that, roughly speaking, one needs a cycle of sufficiently small length in the incidence graph of an instance to find

a contradiction via local consistency methods. In the proof of Theorem 3.1.1 we indeed construct unsatisfiable instances with high girth such that Spoiler could not possibly win the pebble game.

Lemma 3.1.5. *Let $\mathbb{D}$ be a finite constraint language whose relations are at least of arity 3 and such that for every relation R of $\mathbb{D}$ of arity r, the projection of R onto any $r-1$ components is the full relation D^{r-1}. Suppose that for every integer l, there exists an unsatisfiable instance I_l of $\mathrm{CSP}(\mathbb{D})$ of girth at least $4l+1$. Then $\mathbb{D}$ does not have bounded relational width.*

It is rather intuitive that Spoiler should, in order to complicate the situation for Duplicator, place pebbles on variables that are adjacent to a node corresponding to a tuple containing already pebbled variables in the incidence graph of an instance I of $\mathrm{CSP}(\mathbb{D})$. This motivates the definition below.

Definition 3.1.6. Let $\mathbb{D}$ be a constraint language, let I be an instance of $\mathrm{CSP}(\mathbb{D})$, and let $0 < k < l$ and $m \geq 2$ be integers. During any point of the (k,l)-pebble game on I, a nonempty set S of variables of I is called *m-controlled* if the following conditions are satisfied.

1. m variables of I are pebbled and I has girth at least $4m+1$.
2. The incidence graph $\mathcal{G}[S]$ is a tree.
3. All but at most one of the elements that are leaves of $\mathcal{G}[S]$ are pebbled.

We say that a set S of variables is *controlled* if it is m-controlled for some m. Both notions depend on the instance I. If the instance is clear from context, we will omit it.

Example 3.1.7. Let $\mathbb{D}$ be a constraint language with a 3-ary relation R_3 and a 4-ary relation R_4. Let I be an instance of $\mathrm{CSP}(\mathbb{D})$ with the variables $\{x_1, x_2, \ldots, x_8\}$, and the constraints

$$((x_1, x_2, x_3), R_3), \quad ((x_3, x_4, x_5), R_3), \quad \text{and} \quad ((x_1, x_6, x_7, x_8), R_4).$$

Assume that during the (k,l)-pebble game on I, the variables $\{x_3, x_4, \ldots, x_8\}$ are pebbled. Figure 3.1 shows the incidence graph $\mathcal{G}(I)$ of I. Let $S = \{x_1, x_2, x_3, x_6, x_7, x_8\}$. The incidence graph $\mathcal{G}[S]$ is depicted by the subgraph with solid edges in Figure 3.1. The set S is 6-controlled as a set of variables of I and S is 4-controlled as a set of variables of $I[S]$. The set $\{x_1, x_2, \ldots, x_6\}$ is not controlled since x_1, x_2 are unpebbled leaves of $\mathcal{G}[\{x_1, x_2, \ldots, x_6\}]$.

Lemma 3.1.8. *Let $\mathbb{D}$ be a constraint language and let I be an instance of $\mathrm{CSP}(\mathbb{D})$ such that the scope of any constraint of I has at least arity 3. If S is an m-controlled set of variables of I during some version of the pebble game on I, then $|S| \leq 2m$.*

Proof. Denote the number of leaves of a graph $\mathcal{G}$ by $\lambda(\mathcal{G})$. Let $m \geq 2$ and let S be an m-controlled set. At most one of the leaves of $\mathcal{G}[S]$ is not pebbled. Hence, we have

$$m \geq \lambda(\mathcal{G}[S]) - 1.$$

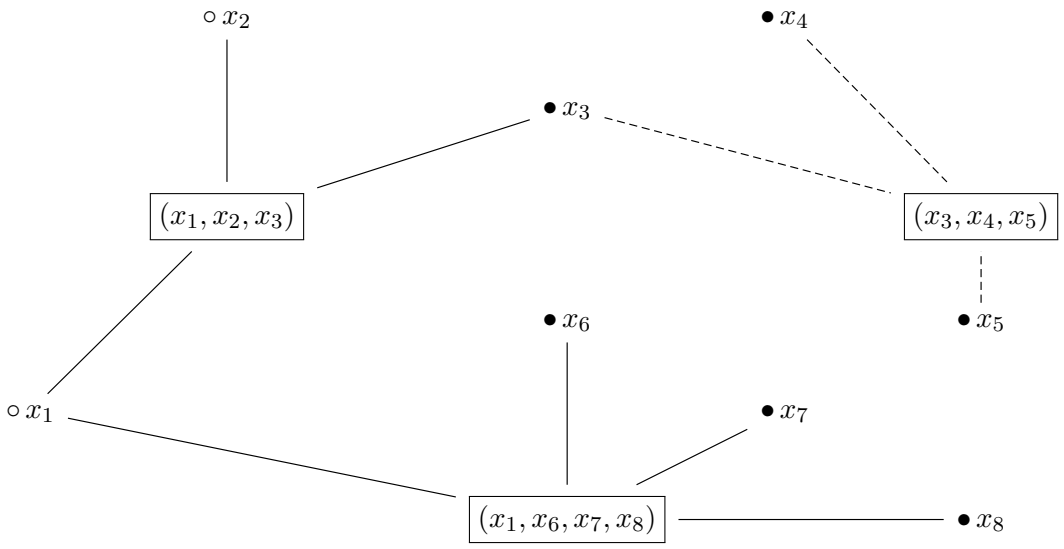

Figure 3.1: This figure shows the incidence graph of the instance in Example 3.1.7. Pebbles are represented by black bullets. Vertices that correspond to the scope of a constraint are represented by rectangles.

Let $W \subseteq S$ be a set of variables such that $\mathcal{G}[W]$ is a tree. We want to show that

$$2(\lambda(\mathcal{G}[W]) - 1) \geq |W| \tag{3.1}$$

holds. This immediately yields $2m \geq |S|$ in case $W = S$. We prove the inequality (3.1) by induction on the number of constraints of $I[W]$. If $I[W]$ contains all of the constraints of $I[S]$, we have $W = S$ since $\mathcal{G}[S]$ is connected.

Firstly, note that I has girth $4m + 1$ and thus every scope of a constraint of I contains pairwise distinct variables. If W is a set of variables such that $\mathcal{G}[W]$ is a tree and such that $I[W]$ has only one constraint, the equation $\lambda(\mathcal{G}[W]) = |W|$ holds. The tuple s contains at least 3 distinct variables and all of them are leaves of $\mathcal{G}[W]$. The inequality $2\lambda(\mathcal{G}[W]) - 2 \geq |W|$ follows.

Assume that there exists a constraint C of $I[S]$ that is not a constraint of $I[W]$. Denote the scope of C by s and write r for the arity of s. Since $I[S]$ is connected, we may choose C such that s is adjacent to a node x of $\mathcal{G}[W]$ in $\mathcal{G}[S]$, i.e., the variable x is contained in s. (In Figure 3.1, such situation is given if we set $W = \{x_1, x_2, x_3, x_6, x_7, x_8\}$, $s = (x_3, x_4, x_5)$ and $x = x_3$.) Adding the variables of s that are not already in W, those are $r - 1 \geq 2$ many, to W yields a strictly bigger set W' such that $\mathcal{G}[W']$ is still a tree. If x was a leaf of $\mathcal{G}[W]$, it is not anymore a leaf of $\mathcal{G}[W']$. All of the other leaves of $\mathcal{G}[W]$ are still leaves of $\mathcal{G}[W']$ and we added $r - 1 \geq 2$ new leaves. Hence, we have

$$2(\lambda(\mathcal{G}[W']) - 1) \geq 2(\lambda(\mathcal{G}[W]) - 1 + (r - 1) - 1) \geq |W| + 2(r - 1) - 2 \geq |W'|.$$

Since $I[S]$ contains only finitely many constraints, the lemma follows. □

Proof of Lemma 3.1.5. By Corollary 2.3.17, it is sufficient to prove that for any arbitrarily large integer l, Spoiler cannot win the $(l-1,l)$-pebble game on the unsatisfiable instance I_l of CSP($\mathbb{D}$) with girth at least $4l+1$ provided by the assumption of the lemma.

We will show via induction on the number $i \in \mathbb{N}$ of rounds played that Duplicator can maintain the following condition during the $(l-1,l)$-pebble game on I_l.

($\star$) Let S be a controlled set after Spoiler has removed pebbles in round i. Let $t \leq l-1$ be the number of pebbled leaves $x_1, \dots, x_t$ of $\mathcal{G}[S]$ and denote the answers of Duplicator by $h' : \{x_1, \dots, x_t\} \to D$. Then there exists a partial solution $h : S \to D$ of I_l that extends h' to all of S.

Since adding only one pebble at a time does not make the game easier for Duplicator, we may only consider such rounds. If the property ($\star$) holds for every $i \in \mathbb{N}$, Duplicator wins the $(l-1,l)$ pebble game on I_l. Indeed, assume $x_1, \dots, x_t$ are the pebbled vertices after Spoiler has potentially removed pebbles in round i. Let x be an unpebbled variable. Then the connected components of $\mathcal{G}[\{x_1, \dots, x_t, x\}]$, more specifically, the corresponding sets of variables are controlled. If Spoiler places a pebble on x, Duplicator can extend previous choices to a partial solution on $\{x_1, \dots, x_t, x\}$ by ($\star$).

In round 1 property ($\star$) is rather trivial as the only controlled sets are one-elementary.

Let $i \in \mathbb{N}$ and assume that ($\star$) holds for rounds $1, \dots, i$. Let x be the variable that Spoiler pebbles in round i. We have to prove that Duplicator can play a value $h(x)$ such that ($\star$) holds in round $i+1$. To that end, denote the controlled sets that have x as an unpebbled leaf in round i, right after Spoiler potentially removed pebbles, by $S_1, \dots, S_N$. Since $\{x\}$ is a controlled set, we have $N \geq 1$.

We claim that $S := S_1 \cup \dots \cup S_N$ is a controlled set as well. All leaves of $S_1, \dots, S_N$ except of x are pebbled and hence $\mathcal{G}[S]$ only has an unpebbled leaf if x is still a leaf of $\mathcal{G}[S]$. The graph $\mathcal{G}[S]$ is connected since every tree $\mathcal{G}[S_1], \dots, \mathcal{G}[S_N]$ shares the vertex x. To prove that $\mathcal{G}[S]$ has no cycles, note, that the sets $S_1, \dots, S_N$ are m-controlled sets of variables of I_l for some $m \leq l$. Hence, Lemma 3.1.8 yields $|S_1 \cup S_2| \leq 2l + 2l = 4l$. Since I_l has girth at least $4l+1$, the graph $\mathcal{G}[S_1 \cup S_2]$ does not have any cycles, i.e., $S_1 \cup S_2$ is controlled. Applying this argument inductively on $(S_1 \cup \dots \cup S_\alpha)$ and $S_{\alpha+1}$ for $\alpha \in \{1, \dots, N-1\}$ yields that S is controlled.

By induction hypothesis ($\star$) for the round i, there exists a partial solution h on S extending previous choices of Duplicator. Duplicator places the pebble corresponding to x on $h(x)$. We will use this specific partial mapping h in the following part of the proof and call it h_S from here on.

Let T be a controlled set in round $i+1$ right after Spoiler has potentially removed pebbles. It is sufficient to consider cases where T contains x and does have an unpebbled leave, call it y. For any other T, we already know that ($\star$) holds because it did in the previous round.

We have to prove that there exists a partial solution $h : T \to D$ which extends the choices Duplicator made for pebbled variables. To that end, the induction hypothesis and secondly the assumption of the lemma that any relation R of arity r in $\mathbb{D}$ has the property

$$\forall d_1, \dots, d_{i-1}, d_{i+1}, \dots d_r \in D \, \exists d_i \in D : (d_1, \dots, d_r) \in R \tag{3.2}$$

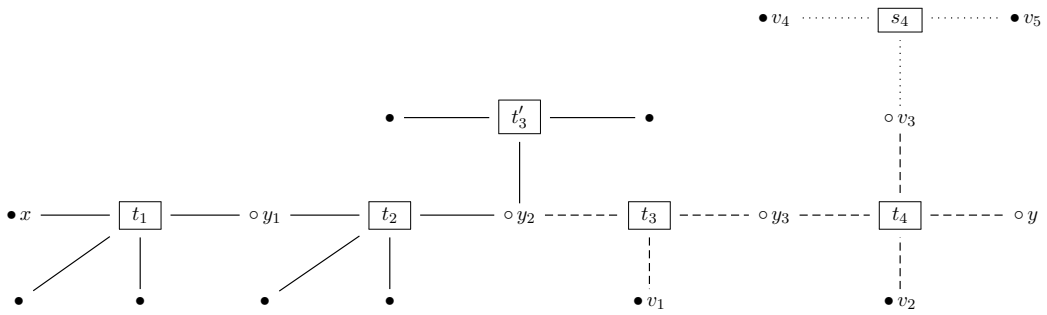

Figure 3.2: This figure shows a possible incidence graph $\mathcal{G}[T]$ in the last part of the proof of Lemma 3.1.5. Pebbled variables are represented by black bullets. Unpebbled variables are represented by circles. Vertices that correspond to the scope of a constraint are represented by rectangles. The subgraph with solid edges depicts $\mathcal{G}[T_S]$. After removing t_3, y_3, t_4, y and edges that are incident to them from $\mathcal{G}[T]$, the connected components that do not contain x are exactly the incidence graphs of the controlled sets $V_1 = \{v_1\}$, $V_2 = \{v_2\}$ and $V_3 = \{v_3, v_4, v_5\}$.

for every position $i \in \{1, \dots r\}$ will be useful.

There is a unique path

$$(x = y_0,\ t_1,\ y_1,\ t_2,\ \dots\ ,y_{M-1},\ t_M,\ y_M = y)$$

connecting x to y in $\mathcal{G}[T]$ where $t_1, \dots, t_M$ are scopes and $y_0, \dots, y_M$ are variables. Since $y_0 = x$ is pebbled and $y_M = y$ is unpebbled, there exists a largest integer $q \in \{0, \dots, M-1\}$ such that y_q is pebbled or such that $q \geq 1$ and y_q is adjacent to a third scope t'_{q+1}, distinct from t_q, t_{q+1}, in $\mathcal{G}[T]$. If we remove the edge (y_q, t_{q+1}) from $\mathcal{G}[T]$, the set T_S of variables that occur in the connected component of $\mathcal{G}[T]$ which contains x, is a controlled set with pebbles on all leaves of $\mathcal{G}[T_S]$. Hence, $T_S \subseteq S$ holds and we define $h_{|_{T_S}} := h_{S|_{T_S}}$.

Now, h is defined everywhere on T except on $y_{q+1}, \dots, y_M$ and possibly on unpebbled variables in branches starting from $t_{q+1}, \dots, t_M$. If we remove $t_{q+1}, y_{q+1}, t_{q+2}, \dots, t_M, y$ and all edges that are incident to them from $\mathcal{G}[T]$, the connected components $\mathcal{G}_1, \dots, \mathcal{G}_L$ of $\mathcal{G}[T]$ that do not contain x are incidence graphs of controlled sets $V_1, \dots, V_L$, respectively. The induction hypothesis yields partial solutions of I_l on each of these sets and we extend h to $V_1 \cup \dots \cup V_L$ accordingly. Now, assume that $\alpha \in \{q, \dots, M-1\}$ is minimal such that h is defined on y_α but not on $y_{\alpha+1}$. Then h is defined for every variable in $t_{\alpha+1}$ except on $y_{\alpha+1}$. Property (3.2) yields a suitable choice for $h(y_{\alpha+1})$. Repeating this argument shows that h can be extended to a partial solution on all of T. □

Proof of Theorem 3.1.1. The relation $\{(x, y) \in G^2 \mid x + y = 0\}$ has a primitive positive definition in $\mathbb{G}$. Hence, for $i \in \{1, 2, 3\}$, the $3 + 2i$-ary relation

$$S_a^i := \left\{ (x_1, \ldots, x_{3+2i}) \in G^{3+2i} \mid x_1 + x_2 + x_3 = a \wedge \bigwedge_{s \in \{1,\ldots,i\}} x_{2s+2} + x_{2s+3} = 0 \right\}$$

is also pp-definable in $\mathbb{G}$. By Theorem 2.4.3, it suffices to prove that the constraint language

$$\mathbb{D} = (G; R_0^3, S_0^1, S_0^2, S_0^3, R_a^3, S_a^1, S_a^2, S_a^3)$$

does not have bounded relational width.

We want to apply Lemma 3.1.5. To that end, note that relations in $\mathbb{D}$ are of arity greater or equal than 3 and the projection of a relation R_c^i or S_c^i onto all except one of its components always yields the full relation G^{r-1} of adequate arity $r - 1$. Thus, we only have to prove that we can construct unsatisfiable instances I_k of CSP($\mathbb{D}$) of girth at least $4k + 1$ for every integer k.

By [Big98], there exists a finite cubic graph $(V_k; E_k)$ of girth at least $4k + 1$ for any positive integer k. We orient the edges of the graph arbitrarily. The variables of I_k will be all the pairs in $V_k \times E_k$. For readability, we write v_e instead of (v, e) for a variable of I_k. For every $v \in V_k$ with three incoming edges e_1, e_2, e_3, we add a constraint $((v_{e_1}, v_{e_2}, v_{e_3}), R_0^3)$ to I_k. For each $v \in V_k$ with two incoming edges e_2, e_3 and one outgoing edge $e_1 = (v, w)$, we add the constraint $((v_{e_1}, v_{e_2}, v_{e_3}, v_{e_1}, w_{e_1}), S_0^1)$ to I_k. Likewise, for each $v \in V_k$ with one or no incoming edge and two or three outgoing edges, we add a constraint involving the relation S_0^2 or S_0^3. Finally, we change exactly one constraint for only one $i \in \{1, 2, 3\}$ of the form (s, S_0^i) to (s, S_a^i).

Suppose for contradiction that h is a solution of I_k. We sum over all constraints interpreted as equations. Since every variable u_e of I_k appears exactly once in a sum of 3 variables and exactly once in a sum of 2 variables, the sum over all equations yields

$$2 \left(\sum_{e \in E_k, u \in e} h(u_e) \right) = a,$$

as there is exactly one constraint with a constraint relation S_a^i in I_k. For every edge $e = \{u, v\} \in E_k$, the equation $h(u_e) + h(v_e) = 0$ must hold and hence, we also have

$$2 \left(\sum_{e \in E_k, u \in e} h(u_e) \right) = \sum_{\{u,v\} \in E_k} (h(u_e) + h(v_e)) = 0.$$

Thus, h cannot be a solution.

Any cycle in the bipartite incidence graph of I_k of length at least $2m$ yields a cycle of length m in the undirected graph $(V_k; E_k)$ and hence I_k has girth at least $4k + 1$. □

3.2 Affine Clones

Definition 3.2.1. Let R be a ring with the multiplicative identity 1 and let M be an R-module. A map $f : M^k \to M$ is called *affine* if there exist coefficients $r_1, \dots, r_n \in R$ with $r_1 + \cdots + r_n = 1$ such that

$$f(x_1, \dots, x_n) = \sum_{i=1}^{n} r_i x_i.$$

The set of affine functions on M is called an *affine clone*.

Definition 3.2.2. An algebra $\mathbf{D}$ is *affine* if it is term-equivalent to an affine clone.

Corollary 3.2.3. *Let* $\mathbf{D}$ *be a finite affine algebra. Then* $\mathbf{D}$ *does not have bounded relational width.*

Proof. Say $\mathbf{D}$ is term-equivalent to an affine clone $\mathcal{D}$ of a finite R-module M. We apply Theorem 3.1.1 to the abelian group $(M; +, 0, -)$. The relations $\{0\}, R_0^3$ and R_a^3 defined in said theorem are invariant under $\mathcal{D}$, hence also under $\mathbf{D}$ and we conclude that $\mathbf{D}$ does not have bounded relational width. □

We can extend the above theorem and give a more explicit description of the affine spaces in play.

Definition 3.2.4. Let p be a prime. Then $\mathrm{Aff}(\mathbb{Z}_p)$ denotes the set of affine subspaces of $\mathbb{Z}_p$, i.e., solution spaces of linear equations over $\mathbb{Z}_p$.

Theorem 3.2.5. *Let* $\mathbb{D}$ *be a finite constraint language such that* $\mathrm{Pol}(\mathbb{D})$ *is idempotent. Then the following are equivalent and imply that* $\mathbb{D}$ *does not have bounded relational width.*

1. $\mathrm{Pol}(\mathbb{D})$ *has a clone homomorphism to an affine clone.*
2. $\mathrm{Pol}(\mathbb{D})$ *has a minion homomorphism to an affine clone.*
2'. $\mathrm{Pol}(\mathbb{D})$ *has a minion homomorphism to the idempotent reduct of* $\mathrm{Clo}(\mathbb{Z}_p; +)$ *for some prime* p*,*
3. $\mathbb{D}$ *pp-constructs* $(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ *for some prime* p*.*

Proof.

(1. ⇒ 2.) This implication is trivial as every minion homomorphism is also a clone homomorphism.

(2. ⇒ 1.) We will delay the proof of this implication and bring it as part of Theorem 6.1.7. Meanwhile, we will of course not use it in any proofs.

(2. $\Rightarrow$ 2'.) Since the composition of two minion homomorphisms is again a minion homomorphism, it is sufficient to prove that for any affine clone there exists a prime p such that we can write down a minion homomorphism to the idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$.

Let R be a ring with the multiplicative identity 1 and let M be an arbitrary finite R-module. The set Φ of all multiplications $\phi_r(m) = rm$ for $r \in R$ on M together with the binary operation defined by $\phi_r +_\Phi \phi_s := \phi_{r+s}$ for $r, s \in R$ is a finite abelian group, where the distributive law on M ensures that $+_\Phi$ is well-defined and that $\phi : r \mapsto \phi_r$ is a homomorphism from $(R; +)$ to $(\Phi; +_\Phi)$. By the fundamental theorem of finite abelian groups, see, e.g., [Rob12], there exists an isomorphism h from $(\Phi; +_\Phi)$ to the direct sum $\bigoplus_{i=1}^{n} \mathbb{Z}_{p_i}^{e_i}$ of finitely many cyclic groups of prime-power order. Visualising the above, we have

$$R \xrightarrow{\phi} (\Phi; +_\Phi) \xrightarrow{h} \bigoplus_{i=1}^{n} \mathbb{Z}_{p_i}^{e_i} \xrightarrow{\pi_1} \mathbb{Z}_{p_1}^{e_1} \xrightarrow{\pi} \mathbb{Z}_{p_1}$$

with the projection π_1 onto the first coordinate and $\pi : \mathbb{Z}_{p_1}^{e_1} \to \mathbb{Z}_{p_1}, x \mapsto x \bmod (p_1)$. The composition $\psi' = \pi \circ \pi_1 \circ h \circ \phi(1)$ is a homomorphism from $(R, +)$ to $(\mathbb{Z}_p, +)$ with $p = p_1$. It preserves addition but might not map $1 \in R$ to $1 \in \mathbb{Z}_p$. Luckily, $\mathbb{Z}_p$ is a field and multiplying by the inverse of $\psi'(1)$ yields another homomorphism $\psi := (\psi'(1))^{-1} \cdot \tilde{\psi}$ with $\psi(1) = 1$.

We can use ψ to define a minion homomorphism Ψ from the affine clone over M to the idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$ by

$$\Psi : \sum_{i=1}^{m} r_i x_i \mapsto \sum_{i=1}^{m} \psi(r_i) x_i.$$

Since ψ preserves addition, Ψ is well-defined. Moreover, $\sum_{i=1}^{m} r_i = 1$ implies

$$\sum_{i=1}^{m} \psi(r_i) = \psi\left(\sum_{i=1}^{m} r_i\right) = 1,$$

and Ψ preserves height 1 identities. Ψ preserves arities and hence, it is indeed a minion homomorphism.

(2'. $\Rightarrow$ 2.) The idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$ consists of functions in $\mathrm{Clo}(\mathbb{Z}_p; +)$ which are idempotent. Every operation f of arity n in $\mathrm{Clo}(\mathbb{Z}_p; +)$ can be written as

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} a_i x_i$$

with $a_1 \ldots, a_n \in \mathbb{Z}_p$. Hence, the equivalence

$$\forall x \in \mathbb{Z}_p : f(x, \ldots, x) = x \iff \forall x \in \mathbb{Z}_p : \sum_{i=1}^{n} a_i x = 1x \iff \sum_{i=1}^{n} a_i = 1$$

shows that the idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$ is indeed an affine clone.

(2. $\iff$ 3.) This follows from Theorem 2.1.44 if we can prove that $\mathrm{Pol}(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ equals the idempotent reduct $\mathcal{D}$ of $\mathrm{Clo}(\mathbb{Z}_p; +)$. To that end, let $f \in \mathcal{D}$, i.e. $f = \sum_{i=1}^{n} a_i x_i$ with $\sum_{i=1}^{n} a_i = 1$, and let

$$R = \{(x^{(1)}, \ldots, x^{(k)}) \in \mathbb{Z}_p^k \mid b_1 x^{(1)} + \cdots + b_k x^{(k)} = b\},$$

where $b, b_1, \ldots b_k \in \mathbb{Z}_p$, be the solution space of a linear equation. Then simple calculations yield $f \rhd R$ and since f and R were arbitrary, $\mathcal{D}$ is contained in $\mathrm{Pol}(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$.

For the other inclusion, we show that any k-ary function f preserving the relations $R_1 := \{1\}$ and $R_2 := \{(x, y, z) \in (\mathbb{Z}_p)^3 \mid x - y = z\}$ must already be in $\mathcal{D}$. Indeed, $f \rhd R_2$ yields that f is additive and, furthermore, linear as multiplication on the vector space $\mathbb{Z}_p^k$ can be achieved by reiterated addition. Hence, the linear form f has a representation $f(x_1, \ldots, x_k) = a_1 x_1 + \cdots + a_k x_k$ and $f \rhd R_1$ yields $f(1, \ldots, 1) = a_1 + \cdots + a_k = 1$, i.e., f is idempotent. □

The following example from [BOP15] shows that the notion of pp-constructibility is strictly weaker than pp-interpretability. As a consequence, if there exists a minion homomorphism from a clone $\mathcal{D}$ to an affine clone, we cannot necessarily find a clone homomorphism from $\mathcal{D}$ to the same affine clone by Theorem 2.1.44 and Theorem 2.1.38. This observation in combination with Theorem 3.2.5 indicates that the notion of clone homomorphisms or, equivalently, on the perspective of relational structures, the notion of pp-interpretations might actually be too narrow for our purposes. Pp-constructions still preserve bounded relational width. A coarser classification via minion homomorphisms and pp-constructions allows us to consider very specific affine clones in Theorem 3.2.5, namely, idempotent reducts of $\mathrm{Clo}(\mathbb{Z}_p; +)$. Pp-constructions still preserve bounded relational width.

Example 3.2.6. Consider the relational structure $\mathbb{D}$ with domain $D = \mathbb{Z}_2^2$ consisting of ternary relations $R_{(a,b)}$, $(a, b) \in \mathbb{Z}_2^2$ defined by

$$R_{(a,b)} = \{(\mathbf{x}, \mathbf{y}, \mathbf{z}) \in (\mathbb{Z}_2^2)^3 : \mathbf{x} + \mathbf{y} + \mathbf{z} = (a, b)\},$$

and unary singleton relations $\{(a, b)\}$, $(a, b) \in \mathbb{Z}_2^2$. Let $\mathbb{D}'$ be the reduct of $\mathbb{D}$ formed by the relations $R_{(0,0)}$, $R_{(1,0)}$, $\{(0, 0)\}$, and $\{(1, 0)\}$. Trivially, $\mathbb{D}'$ is pp-definable from $\mathbb{D}$. Now, take the relational structure $\mathbb{E}$ with $E = \mathbb{Z}_2$ and relations R_a, $a \in \mathbb{Z}_2$, where

$$R_a = \{(x, y, z) \in (\mathbb{Z}_2)^3 : x + y + z = a\},$$

together with singletons $\{0\}$, $\{1\}$.

We will show that $\mathbb{E}$ is pp-constructible but not pp-interpretable from $\mathbb{D}$. Note that $\mathrm{Pol}(\mathbb{D})$ consists of all idempotent affine operations of the module $\mathbb{Z}_2^2$ over $\mathrm{End}(\mathbb{Z}_2^2)$ and $\mathrm{Pol}(\mathbb{E})$ of all idempotent affine operations of the abelian group $\mathbb{Z}_2$.

The mappings $D \to E$, $(x_1, x_2) \mapsto x_1$ and $E \to D$, $x \mapsto (x, 0)$ are homomorphisms from $\mathbb{D}'$ to $\mathbb{E}$ and from $\mathbb{E}$ to $\mathbb{D}'$, respectively. (In fact, $\mathbb{E}$ is the core of $\mathbb{D}'$.) Therefore, $\mathbb{E}$ is homomorphic equivalent to a pp-power of $\mathbb{D}$.

Suppose that $\mathbb{D}$ pp-interprets $\mathbb{E}$ and f is a mapping from $C \subseteq D^n$ onto E witnessing this. Let α be the kernel of f - it is an equivalence relation on C with two equivalence classes (E

has two elements). By the definition of pp-interpretation, both C and α are pp-definable from $\mathbb{D}$. Since $\mathbb{D}$ contains the singleton unary relations, both equivalence classes of α are pp-definable as well. Thus, C is a pp-definable relation which is a disjoint union of two pp-definable relations. This is impossible as it is easily seen that the cardinality of any relation pp-definable from $\mathbb{D}$ is a power of 4.

4 Sufficient Conditions for Bounded Relational Width

Thanks to Theorem 3.2.5, we now know that any constraint language $\mathbb{D}$ which can simulate solving linear equations over $\mathbb{Z}_p$ for some prime p does not have bounded with. Proving the contrary implication shall be the task of this chapter. We will follow the idea presented in [BK14] which was further improved in [Bar14]. Every finite constraint language which does not pp-construct $(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ does not only have bounded relational width but even relational width $(2, 3)$.

Intuition suggests that it is rather difficult to derive information from the negative statements: $\mathbb{D}$ does not pp-construct $(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ for any prime p, or $\mathbb{D}$ does not have relational width (k, l) for any positive integers $k \leq l$. This motivates the following notion.

Definition 4.0.1. A lattice L is *meet semi-distributive*, or is SD($\wedge$), iff for all $x, y, z \in L$, the equation $x \wedge y = x \wedge z$ implies $x \wedge y = x \wedge (y \vee z)$. A variety $\mathcal{V}$ is SD($\wedge$) if the congruence lattice of every algebra in $\mathcal{V}$ is SD($\wedge$). An algebra $\mathbf{D}$ is SD($\wedge$) if $\mathcal{V}(\mathbf{D})$ is SD($\wedge$).

The property SD($\wedge$) is a substantial assumption in Theorem 4.2.5 and Theorem 4.2.7 although we omit the proofs. The lemma below is a consequence of [LZ07][Lemma 4.1] and [HM88][Theorem 9.10], slightly reformulated to fit the terminology of this thesis.

Lemma 4.0.2. *Let* $\mathbf{D}$ *be a finite, idempotent algebra. The following are equivalent:*

1. $\mathrm{Clo}(\mathbf{D})$ *does not have a clone homomorphism to an affine clone.*
2. $\mathbf{D}$ *is* SD($\wedge$).

Let us start by outlining the main results and strategy of this chapter following the proof in [Bar14]. In Lemma 4.1.7, we show that every $(2, 3)$-minimal instance is a *Prague instance*, see Definition 4.1.5. The following theorem is the core result of this chapter. Its rather technical proof will be dealt with in the coming sections.

Theorem 4.0.3. *Let* $\mathbf{D}$ *be a finite, idempotent* SD($\wedge$) *algebra. Then every nontrivial Prague instance which has only constraint relations that are invariant under* $\mathbf{D}$ *has a solution.*

Corollary 4.0.4. *Let* $\mathbf{D}$ *be a finite, idempotent* SD($\wedge$) *algebra. Then every* $(2, 3)$*-minimal instance which has only constraint relations that are invariant under* $\mathbf{D}$ *has a solution.*

Proof. This is an immediate consequence of Lemma 4.1.7 and Theorem 4.0.3. □

Corollary 4.0.5. *Let* $\mathbb{D}$ *be a finite constraint language with all singletons. Furthermore, interpret* $\mathrm{Pol}(\mathbb{D})$ *as an algebra on* D *by choosing an arbitrary signature. Then the following are equivalent.*

1. Pol($\mathbb{D}$) *is* SD($\wedge$).
2. $\mathbb{D}$ *has relational width* $(2,3)$.
3. $\mathbb{D}$ *has bounded relational width.*

Proof. To prove the implication "1 $\implies$ 2", assume that Pol($\mathbb{D}$) is SD($\wedge$) and let I be an instance of CSP($\mathbb{D}$). By Lemma 2.3.28, the $(2,3)$-minimal instance J associated to I has only constraint relations which are invariant under Pol($\mathbb{D}$). If J is nontrivial, then it has a solution by Corollary 4.0.4, as required. The implication "2 $\implies$ 3" is trivial and "3 $\implies$ 1" follows from Theorem 3.2.5 and Lemma 4.0.2. □

Corollary 4.0.6. *Let $\mathbb{D}$ be a finite constraint language. If $\mathbb{D}$ has bounded relational width, then $\mathbb{D}$ has relational width $(2,3)$.*

Proof. Let $\mathbb{D}$ be a finite constraint language with bounded relational width. Lemma 2.4.6 and Corollary 2.4.12 yield that the singleton expansion $\mathbb{D}'$ of the core of $\mathbb{D}$ has bounded relational width. By Corollary 4.0.5, $\mathbb{D}'$ has relational width $(2,3)$ and thus, also $\mathbb{D}$ has relational width $(2,3)$ as any instance of the core of $\mathbb{D}$ is also an instance of $\mathbb{D}'$. □

4.1 Prague Instances

Notation 4.1.1. Let I be a 1-minimal instance. Recall Definition 2.3.32. We write

$$\mathrm{P}_x^{(I)} := \{f(x) \mid f \in \mathrm{P}_{\{x\}}^{(I)}\}$$

for the set of all permissible assignments for x. We omit the instance I and write P_x instead of $\mathrm{P}_x^{(I)}$ if the instance is clear from the context.

Definition 4.1.2. Let $I = (V, D, \mathcal{C})$ be a 1-minimal instance. A *pattern* of length $k-1 > 0$ from x_1 to x_k (in I) is a tuple

$$p = (x_1, C_1, x_2, C_2, \ldots, x_{k-1}, C_{k-1}, x_k),$$

where $x_i \in V$ for every $i \in \{1, \ldots, k\}$, $C_i \in \mathcal{C}$, and $\{x_i, x_{i+1}\}$ is contained in the scope of C_i for every $i \in \{1, \ldots, k-1\}$. The set of all variables in p is denoted by $[\![p]\!]$, i.e., $[\![p]\!] = \{x_1, \ldots, x_k\}$. The pattern is *closed* if $x_1 = x_k$.

A *realization of p in I* is a tuple $(f_1, \ldots, f_{k-1})$ of functions such that for every $i \in \{1, \ldots, k-1\}$, the function $f_i : s_i \to D$ solves the constraint $C_i = (s_i, R_i)$, and such that $f_i(x_{i+1}) = f_{i+1}(x_{i+1})$ holds. We say that this realization *connects $f_1(x_1)$ to $f_{k-1}(x_k)$* and that *p connects $a \in \mathrm{P}_x$ to $b \in \mathrm{P}_y$* if there exists a realization of p that connects a to b.

For $X \subseteq V, x \in X$, and $a, b \in \mathrm{P}_x$, we say that *a and b are connected in X* if there exists a pattern p from x to x which connects a to b such that $[\![p]\!] \subseteq X$. (The notion depends on the variable x, which should always be clear from the context.)

If p is a pattern from x to y and $A \subseteq \mathrm{P}_x$, we define a subset $A + p$ of P_y by

$$A + p = \{b \in \mathrm{P}_y : \exists a \in A,\ p \text{ connects } a \text{ to } b\}.$$

If $p = (x_1, C_1, x_2, \ldots, C_{k-1}x_k)$ and $q = (y_1, C'_1, y_2, \ldots, C'_{l-1}, y_l)$ are patterns such that $x_k = y_1$, we define

$$p + q := (x_1, C_1, x_2, \ldots, C_{k-1}, x_k, C'_1, y_2, \ldots, C'_{l-1}, y_l)$$

and

$$-p := (x_k, C_{k-1}, x_{k-1}, \ldots, C_1, x_1).$$

For a closed pattern p and a positive integer m, we write

$$m \times p := \underbrace{p + p + \cdots + p}_{m\times}.$$

Notation 4.1.3. We write $A - p$ for $A + (-p)$. Since $(A + p) + q = A + (p + q)$ (whenever the expressions make sense), we can simply write $A + p + q$.

Remark 4.1.4. For $x, y \in V$, a pattern p from x to y, a set $A \subseteq \mathrm{P}_x$ of possible assignments, and a constraint C whose scope contains x, observe

1. $\mathrm{P}_x + p = \mathrm{P}_y$,
2. $A \subseteq A + p - p$, and
3. $A + (x, C, x) = A$.

Definition 4.1.5. An instance $I = (V, D, \mathcal{C})$ is a *Prague instance* if

(P1) I is 1-minimal, and

(P2) for any $x \in V$, any closed pattern p from x to x and any $a, b \in \mathrm{P}_x$, whenever a and b are connected in $[\![p]\!]$, then there exists an integer $k > 0$ such that $k \times p$ connects a to b.

Lemma 4.1.6. *If $I = (V, D, \mathcal{C})$ is a $(2, 3)$-minimal instance, $x, y \in V$ and $h \in \mathrm{P}_{\{x,y\}}$ then any pattern $q = (x = x_1, C_1, \ldots, x_l = y)$ connects $a = h(x)$ to $b = h(y)$.*

Proof. We prove this statement via induction on the length l of q. For $l = 1$, simply extend h to a solution of C_1.

Let $l > 1$ and assume the statement holds for patterns of length $l - 1$. Since the instance is $(2, 3)$-minimal, there exists a constraint C of I such that its scope s contains $\{x_1, x_2, x_l\}$, and there exists a solution g of C extending h to s. We set $h' = g_{|\{x_2,x_l\}}$ and $a' = g(x_2) = h'(x_2)$. By (M2), we have $g_{|\{x_1,x_2\}} \in \mathrm{Sol}_{\{x_1,x_2\}}(C_1)$ and hence there exists a solution f_1 of C_1 extending $g_{|\{x_1,x_2\}}$ to the scope of C_1. The induction hypothesis yields a realization $(f_2, \ldots, f_{l-1})$ of $(x_2, C_2, \ldots, x_l)$ that connects $a' = h'(x_2)$ to $b = h'(x_l)$. Hence, $(f_1, f_2, \ldots, f_{l-1})$ connects a to b in q. □

Lemma 4.1.7. *Every $(2,3)$-minimal instance is a Prague instance.*

Proof. Any $(2,3)$-minimal instance is 1-minimal by definition.

In order to prove the second property of Prague instances, choose any variable $x \in V$, an arbitrary closed pattern $p = (x_1 = x, C_1, \dots, x_k = x)$ from x to x, and $a, b \in \mathrm{P}_x$ that are connected via a realisation $(f_1, \dots, f_{m-1})$ of a pattern $q = (y_1 = x, C'_1, y_2, \dots, y_m = x)$ such that $[\![q]\!] \subseteq [\![p]\!]$. Clearly, $(f_i)_{|_{\{y_i, y_{i+1}\}}} \in \mathrm{P}_{\{y_i, y_{i+1}\}}$ holds for every $i \in \{1, \dots, m-1\}$. Hence, by Lemma 4.1.6, the sub-patterns p_i of $p + p$, where

$$\begin{aligned} p_1 &= (x = y_1, C_1, \dots, y_2), \\ p_i &= (y_i, \dots, C_{k-1}, x) + (x, C_1, \dots, y_{i+1}) \text{ for } i \in \{2, \dots, m-2\}, \\ \text{and} \quad p_{m-1} &= (y_{m-1}, \dots, C_{k-1}, x = y_m), \end{aligned}$$

respectively, connect $f_i(y_i)$ to $f_i(y_{i+1})$ for every $i \in \{1, \dots, m-1\}$. Thus,

$$(m-2) \times p = p_1 + p_2 + \cdots + p_{m-1}$$

connects a to b. □

Lemma 4.1.8. *Let $I = (V, D, \mathcal{C})$ be a Prague instance, let $x \in V$, and let p be a closed pattern from x to x. Then there exists a positive integer m such that for every integer $k \geq m$ and every $a, b \in \mathrm{P}_x$, if a and b are connected in $[\![p]\!]$, then $k \times p$ connects a to b.*

Proof. Since D is finite, we may use (P2) in the definition of Prague instances and restrict ourselves to the case $a = b$. It even suffices to prove the existence of such m for fixed but arbitrary $a = b$, again, due to the finiteness of D. Now, by (P2), there exists an integer l such that $l \times p$ connects a to a. Let $c \in \mathrm{P}_x$ be such that p connects a to c and $(l-1) \times p$ connects c to a. Since a and c are connected in $[\![l \times p]\!]$, there exists an integer l' such that $l' \times (l \times p) = (ll') \times p$ connects a to c. Now, $l \times p$ and $(l'l + l - 1) \times p$ both connect a to a. Since l and $l'l + l - 1$ are coprime, it follows that $k \times p$ connects a to a for every $k \geq l(l'l + l - 1)$, as required. □

Lemma 4.1.9. *Let $I = (V, D, \mathcal{C})$ be a Prague instance, let $x, y \in V$, let p be a pattern from x to y, let q be a pattern from y to x, let $A \subseteq \mathrm{P}_x$, $B \subseteq \mathrm{P}_y$, and let r be a pattern from x to y such that $[\![r]\!] \subseteq [\![p + q]\!]$. If $A + p = B$ and $B + q = A$, then $A + r = B$.*

Proof. Let $a \in A + r$. Then there exists an element $b \in B$ such that a and b are connected in $[\![p + q]\!]$, e.g., via the path $-r + p$. Hence, b and a are connected via the path $k(q + p)$ for some k by the definition of Prague instance. $B + q + p = B$ yields $a \in B$ and, furthermore, $A + r \subseteq B$. Analogously, we obtain $B - r \subseteq A$. Now, $B \subseteq B + (-r) - (-r) \subseteq A + r$ and $A + r = B$ follows. □

4.2 Proof of Theorem 4.0.3

The strategy of this proof is to successively make the Prague instance smaller until P_x contains only one element for every $x \in V$. Sending x to that left-over element clearly yields a solution of the original Prague instance.

Throughout the proof, unless stated otherwise, $\mathbf{D}$ denotes a finite, idempotent SD($\wedge$) algebra and $I = (V, D, \mathcal{C})$ is a Prague instance with constraint relations that are invariant under $\mathbf{D}$ such that $|\,\mathrm{P}_x\,| > 1$ for at least one $x \in V$. Furthermore, note that each set P_x is the projection of a subpower of $\mathbf{D}$ and hence a subuniverse of $\mathbf{D}$. We write $\mathbf{P}_x$ for the subalgebra of $\mathbf{D}$ with domain P_x.

The proof of Theorem 4.0.3 is split into three parts. The first two parts are dedicated to the construction of a system

$$(t, X, (\mathrm{P}_x^i)_{i=0,\, x\in V}^n),$$

consisting of an n-ary operation t of $\mathbf{D}$, a nonempty subset $X \subseteq V$, and subsets $\mathrm{P}_x^i \subseteq \mathrm{P}_x$, $x \in V$, $i \in \{0, \dots, n\}$ such that

(D1) P_x^0 is a proper subset of P_x for every $x \in X$,
$\mathrm{P}_x^i = \mathrm{P}_x$ for every $x \in V \setminus X$, $i \in \{0, \dots, n\}$,

(D2) P_x^i is a subuniverse of $\mathbf{P}_x$ for every $x \in V$, $i \in \{0, \dots, n\}$,

(D3) $\mathrm{P}_x^i + (x, C, y) = \mathrm{P}_y^i$ for every $x \in X$, $y \in V$, $i \in \{0, \dots, n\}$ and every constraint $C \in \mathcal{C}$ whose scope contains $\{x, y\}$,

(D4) $t(a_1, \dots, a_n) \in \mathrm{P}_x^0$ for every $x \in V$, $a_1, \dots, a_n \in \mathrm{P}_x$ such that $a_i \in \mathrm{P}_x^i$ holds for all but at most one $i \in \{1, \dots, n\}$.

We distinguish whether there exists a $z \in V$ such that $\mathbf{P}_z$ has a *proper absorbing subuniverse* or not. Afterwards, in Theorem 4.2.9, a smaller Prague instance shall be constructed from the system. Repeating this construction finitely many times yields a Prague instance with $|\,\mathrm{P}_x\,| = 1$ for every $x \in V$.

Definition 4.2.1. A subuniverse A of an idempotent algebra $\mathbf{D}$ is *absorbing* if there exists an operation $t \in \mathrm{Clo}(\mathbf{D})$ of arity $n > 1$ such that $t(a_1, \dots, a_n) \in A$ whenever $a_i \in D$ and $a_i \in A$ for all but at most one $i \in \{1, \dots, n\}$. An absorbing subuniverse A of D is called *proper* if $\emptyset \subsetneq A \subsetneq D$.

Lemma 4.2.2. *Let $\mathbf{D}$ be a finite algebra and let $I = (V, D, \mathcal{C})$ be a Prague instance which has only constraint relations that are subpowers of $\mathbf{D}$. Let $x, y \in V$ and let p be a pattern from x to y.*

(1) The set $S = \{(a, b) \in \mathrm{P}_x \times \mathrm{P}_y : p$ connects a to $b\}$ is invariant under $\mathbf{D}$ and its projection to the first (the second, resp.) coordinate is P_x (P_y, resp.).

(2) If s is a k-ary operation of $\mathbf{D}$, $A_1, \dots, A_k, B \subseteq \mathrm{P}_x$ and $s(a_1, \dots, a_k) \in B$ for any $a_i \in A_i$, $i \in \{1, \dots, k\}$, then for any $a'_1, \dots, a'_k \in \mathrm{P}_y$ such that $a'_i \in A_i + p$, $i \in \{1, \dots, k\}$, we have $s(a'_1, \dots, a'_k) \in B + p$.

(3) If A is a subuniverse (an absorbing subuniverse, resp.) of $\mathbf{P}_x$, then $A + p$ is a subuniverse (an absorbing subuniverse, resp.) of $\mathbf{P}_y$.

Proof. To prove (1), let s be a k-ary operation of $\mathbf{D}$, let $(a_i, b_i) \in S$ and let $(f_1^{(i)}, \dots, f_n^{(i)})$ be realizations of p connecting a_i to b_i for $i \in \{1, \dots, k\}$. Then

$$(s(f_1^{(1)}, \dots, f_1^{(k)}), \dots, s(f_n^{(1)}, \dots, f_n^{(k)}))$$

is a realization of p connecting $s(a_1, \dots, a_k)$ to $s(b_1, \dots, b_k)$ because the constraint relations of I are preserved by s. Since every Prague instance is 1-minimal by definition, the second part follows.

(2) follows similarly as the first part of (1).

For (3), apply (2) with $A_1 = \cdots = A_k = B = A$ (and $A_i = \mathrm{P}_x$ for one $i \in \{1, \dots, k\}$ in the case of an absorbing subuniverse A). □

Lemma 4.2.3. *If R, S are subuniverses of $\mathbf{D}^2$, then the relational composition*

$$R \circ S = \{(a, c) \in D \times D : (\exists b \in D)\, (a, b) \in R \text{ and } (b, c) \in S\}$$

is also a subuniverse of $\mathbf{D}^2$.

Proof. Straightforward. □

4.2.1 Absorption

Theorem 4.2.4. *Let $z \in V$ be such that $\mathbf{P}_z$ has a proper absorbing subuniverse E. Let t be an operation of $\mathbf{P}_z$ witnessing the absorption and let n be its arity.*

Define a quasiorder $\preceq$ on the set of all pairs (A, x) such that $x \in V$ and $A \subsetneq \mathrm{P}_x$ by

$$(A, x) \preceq (B, y) \text{ iff } B = A + p \text{ for some pattern } p \text{ from } x \text{ to } y.$$

Let $\mathcal{M}$ be a maximal class of the partial order induced by $\preceq$, greater or equal to the class of (E, z). Define

$$X := \{x \in V : \exists A \subsetneq \mathrm{P}_x,\ (A, x) \in \mathcal{M}\}.$$

Then for every $x \in X$, the set $M_x \subseteq \mathrm{P}_x$ such that $(M_x, x) \in \mathcal{M}$ is unique. Furthermore, the system $(t, X, (\mathrm{P}_x^i)_{i=0, x \in V}^n)$, where for all $i \in \{0, \dots, n\}$, we define

$$\mathrm{P}_x^i = \begin{cases} M_x, & \text{for } x \in X, \\ \mathrm{P}_x, & \text{for } x \in V \setminus X, \end{cases}$$

satisfies properties (D1) - (D4) introduced in the beginning of this section.

Proof. Let $(A, x), (B, x) \in \mathcal{M}$. By definition, we have $(A, x) \preceq (B, x) \preceq (A, x)$, i.e., there exist patterns p, q from x to x with $A + p = B$ and $B + q = A$. This implies $A + (p + q) = A$ and with $[\![p]\!] \subseteq [\![(p + q) + (p + q)]\!]$, Lemma 4.1.9 yields $A + p = A$. Hence, $A = A + p = B$ holds for $(A, x), (B, x) \in \mathcal{M}$

(D1) is satisfied by construction.

(D2) and (D4) are obvious for $x \in V \setminus X$. For $x \in X$, we apply item (3) of Lemma 4.2.2: since E is an absorbing subuniverse of $\mathbf{P}_z$ and $(E, z) \preceq (\mathrm{P}_x^i, x)$ holds for every $i \in \{0, \dots, n\}$, there exists a pattern p from z to x such that $E + p = \mathrm{P}_x^0 = \cdots = \mathrm{P}_x^n$ and hence P_x^0 is an

absorbing subuniverse of $\mathbf{P}_x$, where the absorption is witnessed by t. This yields (D2) and (D4). We are left to prove (D3).

To that end, let $x \in X$, $y \in V$, $i \in \{0, \dots, n\}$ and let $C \in \mathcal{C}$ be a constraint such that its scope contains $\{x, y\}$. If $y \in X$, we have $(\mathrm{P}^i_y, y) \in \mathcal{M}$ by definition and hence there exist patterns p from x to y and q from y to x such that $\mathrm{P}^i_x + p = \mathrm{P}^i_y$ and $\mathrm{P}^i_y + q = \mathrm{P}^i_x$ holds. Thus, Lemma 4.1.9 yields $\mathrm{P}^i_x + (x, C, y) = \mathrm{P}^i_y$. If $y \in V \setminus X$ and $\mathrm{P}^i_x + (x, C, y) \subsetneq \mathrm{P}^i_y = \mathrm{P}_y$, the pair $(\mathrm{P}^i_x + (x, C, y), y)$ would belong to the maximal class $\mathcal{M}$ which would yield $y \in X$, a contradiction. □

4.2.2 No absorption

In the case that $\mathbf{P}_x$ does not have a proper absorbing subalgebra for every $x \in V$, the following theorem yields the term t for our system satisfying (D1) to (D4).

Theorem 4.2.5. *Let* $\mathbf{D}$ *be a finite, idempotent* SD(∧) *algebra. Then there exists a term* $t \in \mathrm{Clo}(\mathbf{D})$ *and elements* $c_1, \dots, c_n, b \in D$ *such that* $t(a_1, \dots, a_n) = b$ *whenever* $a_1, \dots, a_n \in D$ *for all* $i \in \{1, \dots, n\}$ *and* $a_i = c_i$ *for all but at most one* $i \in \{1, \dots, n\}$.

Proof. [BKS15, Lemma 2.0.12]. □

The proof of Theorem 4.2.5 relies on a theorem called Absorption Theorem. We will state and use a slightly altered version of this theorem.

Definition 4.2.6. Let P and Q be nonempty sets. A subset R of $P \times Q$ is called *linked* if the bipartite graph $(P \dot\cup Q, R)$ with the partite sets P and Q, where every tuple in R is regarded as an undirected edge, is connected, i.e., any two vertices in $P \dot\cup Q$ are connected via a path only using edges of R.

Equivalently, the projection of R to the first (the second, resp.) coordinate yields P (Q, resp.) and the transitive closure of

$$R \circ R^{-1} = \{(a, b) \in P^2 : (\exists c \in Q)\, (a, c), (b, c) \in R\}$$

is equal to P^2.

Theorem 4.2.7. *Let* $\mathbf{D}$ *be a finite, idempotent* SD(∧) *algebra, let* R *be a subuniverse of* $\mathbf{D}^2$, *let* P, Q *be the projections of* R *to the first and the second coordinate, respectively, and let* $\mathbf{P}, \mathbf{Q}$ *be the corresponding subalgebras of* $\mathbf{D}$. *If neither* $\mathbf{P}$ *nor* $\mathbf{Q}$ *has a proper absorbing subuniverse and* R *is linked (as subset of* $P \times Q$*), then* $R = P \times Q$.

Proof. There is no clone homomorphism from $\mathrm{Clo}(\mathbf{D})$ to an affine clone by Lemma 4.0.2. Hence, there is also no clone homomorphism from $\mathrm{Clo}(\mathbf{D})$ to the clone of all projections. This is equivalent to $\mathbf{D}$ being Taylor, see e.g., [Bod21, Theorem 6.6.4]. Under that weaker assumption, the theorem is known as the Absorption Theorem. A proof can be found in [KB12]. □

Theorem 4.2.8. *Assume that for every* $x \in V$ *the algebra* $\mathbf{P}_x$ *does not have a proper absorbing subuniverse. Then there exists a system* $(t, X, (\mathrm{P}^i_x)^n_{i=0,\, x \in V})$, *consisting of an* n*-ary operation* t *of* $\mathbf{D}$, *a nonempty subset* $X \subseteq V$, *and subsets* $\mathrm{P}^i_x \subseteq \mathrm{P}_x$, $x \in V$, $i \in \{0, \dots, n\}$, *which satisfies the properties (D1) - (D4).*

Proof. Let $z \in V$ be a variable such that $|\,\mathrm{P}_z\,| > 1$ and let $\sim$ be a maximal congruence of $\mathbf{P}_z$, see Remark 2.1.12. Theorem 4.2.5 applied to $\mathbf{P}_z$ yields an n-ary operation $t \in \mathrm{Clo}(\mathbf{P}_z)$ (we can also interpret t as term of $\mathrm{Clo}(\mathbf{D})$) and elements $c_0, c_1, \ldots, c_n \in \mathrm{P}_z$ such that $t(a_1, \ldots, a_n) = c_0$ whenever $a_i \in \mathrm{P}_z$ for all $i \in \{1, \ldots, n\}$ and $a_i = c_i$ for all but at most one $i \in \{1, \ldots, n\}$. We set

$$\mathrm{P}_z^i = [c_i]_\sim$$

for all $i \in \{0, 1, \ldots, n\}$ and define

$$X := \{x \in V : \text{ there exists a pattern } p_x \text{ from } z \text{ to } x \text{ such that } \mathrm{P}_z^0 + p \subsetneq \mathrm{P}_x\}.$$

Choose a pattern p_x from z to x with $\mathrm{P}_z^0 + p_x \subsetneq \mathrm{P}_x$ for every $x \in X$. Now, for every $i \in \{0, \ldots, n\}$, set

$$\mathrm{P}_x^i = \begin{cases} \mathrm{P}_z^i + p_x, & \text{for } x \in X \setminus \{z\}, \text{ and} \\ \mathrm{P}_x, & \text{for } x \in V \setminus X. \end{cases}$$

Property (D1) is satisfied by construction. Properties (D2) and (D4) are trivial for $x \in V \setminus X$. We now show that (D2) and (D4) hold for $x \in X$ separately for the cases $x = z$ and $x \neq z$.

Let $x = z$. The idempotency of $\mathbf{D}$ yields (D2). Let $i \in \{0, 1, \ldots, n\}$. If s is a k-ary operation of $\mathbf{P}_z$ and $a_1, \ldots, a_k \in \mathrm{P}_z^i = [c_i]_\sim$, then

$$s(a_1, \ldots, a_k) \sim s(c_i, \ldots, c_i) = c_i$$

and hence $s(a_1, \ldots, a_k) \in [c_i]_\sim = \mathrm{P}_z^i$. For (D4), we utilize the specific attribute of our t. If $a_i \in \mathrm{P}_z$ for all $i \in \{1, \ldots, n\}$ and $a_i \in \mathrm{P}_z^i$, i.e., $a_i \sim c_i$ for all but at most one $i \in \{1, \ldots, n\}$, say $i = 1$, then

$$t(a_1, \ldots, a_n) \sim t(a_1, c_2, \ldots, c_n) = b$$

and hence $t(a_1, \ldots, a_n) \in [b]_\sim = \mathrm{P}_z^0$.

Let $x \in X$, $x \neq z$. To prove (D4), let $i' \in \{1, \ldots, n\}$, and let $a'_1, \ldots, a'_n \in \mathrm{P}_x$ be such that $a'_i \in \mathrm{P}_x^i$ for all $i \in \{1, \ldots, n\} \setminus \{i'\}$. By construction, $\mathrm{P}_x^i = \mathrm{P}_z^i + p_x$ holds for all $i \in \{0, \ldots, n\}$. Recall, that $\mathrm{P}_x = \mathrm{P}_z + p_x$. Hence, we can apply item (2) of Lemma 4.2.2 with $s = t$, $p = p_x$, $B = \mathrm{P}_z^0$, and $A_i = \mathrm{P}_z^i$ for all $i \in \{1, \ldots, n\} \setminus \{i'\}$, as well as $A_{i'} = \mathrm{P}_z$, and $t(a'_1, \ldots, a'_n) \in \mathrm{P}_x^0$ follows.

Item (3) of the same lemma transfers property (D2) from $x = z$ to arbitrary $x \in X$.

We are left to prove (D3). To that end, we use the following consequence of Theorem 4.2.7.

Claim 1. For every $x \in V$ and every pattern p from z to x, either

(1) for any $i, j \in \{0, \ldots, n\}$ with $\mathrm{P}_z^i \neq \mathrm{P}_z^j$, the sets $\mathrm{P}_z^i + p$ and $\mathrm{P}_z^j + p$ are disjoint, or

(2) $\mathrm{P}_z^i + p = \mathrm{P}_x$ for all $i \in \{0, \ldots, n\}$.

Proof of Claim 1. Define

$$\begin{aligned} S &= \{(a,b) \in \mathrm{P}_z \times \mathrm{P}_x : p \text{ connects } a \text{ to } b\} \\ R &= \sim \circ S = \{(a,b) \in \mathrm{P}_z \times \mathrm{P}_x : p \text{ connects some } a' \sim a \text{ to } b\}, \text{ and} \\ R^{-1} &= \{(b,a) : (a,b) \in R\}. \end{aligned}$$

By item (1) of Lemma 4.2.2, the relation S and, as congruence of $\mathbf{P}_z$, also $\sim$ are subpowers of $\mathbf{D}$. By Lemma 4.2.3, the relational composition $R = \sim \circ S$ is also a subpower of $\mathbf{D}$ and $R \circ R^{-1}$ is a subpower of $\mathbf{P}_z$. Furthermore, notice that the projection of R to its first coordinate equals P_z, its projection to the second coordinate equals P_x, and

$$\mathrm{P}_z^i + p = \{b : (c_i, b) \in R\}$$

for every $i \in \{0, \dots, n\}$.

Let β be the transitive closure of $R \circ R^{-1}$. Since β is finite, we can write β as a relational product of finitely many copies of $R \circ R^{-1}$, i.e., $\beta = R \circ R^{-1} \circ \cdots \circ R \circ R^{-1}$. Clearly, β is an equivalence relation and due to Lemma 4.2.3 also a congruence of $\mathbf{P}_z$.

As $\sim \subseteq \beta$ and $\sim$ is a maximal congruence, either $\beta = \sim$, or $\beta = \mathrm{P}_z \times \mathrm{P}_z$. The first case translates to (1), since $\mathrm{P}_z^i \neq \mathrm{P}_z^j$ is equivalent to $(c_i, c_j) \notin \sim$ by definition, but

$$\emptyset \neq (\mathrm{P}_z^i + p) \cap (\mathrm{P}_z^j + p) = \{b : (c_i, b) \in R\} \cap \{b : (c_j, b) \in R\}$$

would yield the contradiction $(c_i, c_j) \in \beta$. In the second case R is linked and Theorem 4.2.7 implies $R = \mathrm{P}_z \times \mathrm{P}_x$, hence $\mathrm{P}_z^i + p = \mathrm{P}_x$ holds for every $i \in \{0, \dots, n\}$. ■

For any $x \in X$, case (1) takes place for the pattern p_x since $\mathrm{P}_z^0 + p_x \subsetneq \mathrm{P}_x$ by definition of X. Hence, $\beta = \sim$ and we see

$$\mathrm{P}_z^i \subseteq \mathrm{P}_z^i + p_x - p_x \subseteq \{a \in \mathrm{P}_z : \exists a' \in \mathrm{P}_z^i,\ (a', a) \in \beta\} = \mathrm{P}_z^i$$

and thus $\mathrm{P}_x^i - p_x = (\mathrm{P}_z^i + p_x) - p_x = \mathrm{P}_z^i$ for every $i \in \{0, \dots, n\}$.

We can now prove (D3). Let $x \in X$, $y \in V$, and let $C \in \mathcal{C}$ be a constraint whose scope contains $\{x, y\}$.

If $y \in V \setminus X$, we have $\mathrm{P}_z^0 + p_x + (x, C, y) = \mathrm{P}_y$ by definition of X. Hence, case (2) of the claim must take place for the pattern $p_x + (x, C, y)$ from z to y, i.e., for every $i \in \{0, \dots, n\}$, we have $\mathrm{P}_z^i + p_x + (x, C, y) = \mathrm{P}_y$ and further

$$\mathrm{P}_x^i + (x, C, y) = (\mathrm{P}_z^i + p_x) + (x, C, y) = \mathrm{P}_y = \mathrm{P}_y^i .$$

If $y \in X$, we have $\mathrm{P}_x^i - p_x + p_y = \mathrm{P}_z^i + p_y = \mathrm{P}_y^i$ and $\mathrm{P}_y^i - p_y + p_x = \mathrm{P}_z^i + p_x = \mathrm{P}_x^i$ for every $i \in \{0, \dots, n\}$. Hence, Lemma 4.1.9 yields $\mathrm{P}_x^i + (x, C, y) = \mathrm{P}_y^i$. □

4.2.3 Smaller instance

Theorem 4.2.9. *Let t be an n-ary operation of $\mathrm{Clo}(\mathbf{D})$, let X be a nonempty subset of V and let $\mathrm{P}_x^i \subseteq \mathrm{P}_x$ for $x \in V$, $i \in \{0, \dots, n\}$ be sets such that the system $(t, X, (\mathrm{P}_x^i)_{i=0,\, x \in V}^n)$*

satisfies (D1) to (D4). Define instances $I^i = (V, D, \mathcal{C}^i)$ *for every* $i \in \{0, \dots, n\}$*, where for every constraint* $C = (\{x_1, \dots, x_k\}, R)$ *of* I*, we set*

$$C^i := (\{x_1, \dots, x_k\}, R \cap (\mathrm{P}^i_{x_1} \times \cdots \times \mathrm{P}^i_{x_k})) \quad \text{and} \quad \mathcal{C}^i = \{C^i : C \in \mathcal{C}\}.$$

Then

(1) for every $i \in \{0, \dots, n\}$*, the instance* I^i *is 1-minimal with* $\mathrm{P}^{(I^i)}_x = \mathrm{P}^i_x$*, its constraint relations are invariant under* $\mathbf{D}$*, and*

(2) the instance I^0 *is a Prague instance with* $\mathrm{P}^{(I^0)}_x \subseteq \mathrm{P}^{(I)}_x$ *for every* $x \in V$ *and* $\mathrm{P}^{(I^0)}_x \subsetneq \mathrm{P}^{(I)}_x$ *for at least one* $x \in V$*.*

Proof. For the proof of item (1), let $i \in \{0, \dots, n\}$. The constraint relations of I^i are invariant under $\mathbf{D}$ because those of I were and because of (D2). We have to prove $\mathrm{Sol}_{\{x\}}(C^i) = \{x\} \times \mathrm{P}^i_x$ for every constraint $C^i \in \mathcal{C}^i$ and every x in the scope s of C^i. To that end, fix one variable x in s. The inclusion $\mathrm{Sol}_{\{x\}}(C^i) \subseteq \{x\} \times \mathrm{P}^i_x$ is clear by construction. Let $a \in \mathrm{P}^i_x$. We want to prove that C^i permits the assignment $\{(x, a)\}$.

Assume $x \in X$. Let C be a constraint of I that corresponds to C^i and let f be a solution of C with $f(x) = a$. By (D3), we have $\mathrm{P}^i_x + (x, C, y) = \mathrm{P}^i_y$ for every y in the scope s of C. Hence, $f(y) \in \mathrm{P}^i_y$ for every y in s and f also solves C^i. Thus, its restriction to $\{x\}$ is in $\mathrm{Sol}_{\{x\}}(C^i)$.

Now, assume $x \notin X$ and that there exists a y in s with $y \in X$. Then (D3) and (D1) yield $\mathrm{P}^i_y + (y, C, x) = \mathrm{P}_x$. Hence, there exists some solution f of C with $f(x) = a$ and $f(y) \in \mathrm{P}^i_y$. The same argument as above applied on $y \in X$ instead of x implies that f solves C^i and its restriction to $\{x\}$ is in $\mathrm{Sol}_{\{x\}}(C^i)$.

Lastly, assume that s does not contain variables of X. Then by (D1), we have $C^i = C$ and $\mathrm{Sol}_{\{x\}}(C^i) = \{x\} \times \mathrm{P}^i_x = \{x\} \times \mathrm{P}_x$. Altogether, the the instance I^i is 1-minimal for every $i \in \{0, \dots, n\}$.

Property (D1) and item (1) immediately yield that I_0 is properly contained in I as described in item (2). We are left to prove that $J = I^0$ satisfies the property (P2) in the definition of a Prague instance.

To that end, we introduce notation to distinguish patterns (and later realizations of these) in I and I^i for $i \in \{0, \dots, n\}$. Whenever $p = (x_1, C_1, x_2, C_2, \dots, x_k)$ is a pattern in I and $i \in \{0, \dots, n\}$, we write

$$p^i = (x_1, C^i_1, x_2, C^i_2, \dots, x_k)$$

for the corresponding pattern in I^i. Note that any pattern in I^i can be written as p^i for a suitable pattern p in I.

Let $x \in V$, $a, b \in \mathrm{P}^{(J)}_x = \mathrm{P}^0_x$. Assume that p is a closed pattern in I from x to x such that a and b are connected in $[\![p]\!] = [\![p^0]\!]$ in J. We have to prove that $k \times p^0$ connects a to b for some integer k. In fact, we can even weaken the assumption and suppose that a and b are merely connected in $[\![p]\!]$ in the instance I.

Lemma 4.1.8 yields an integer m such that for every $m' \geq m$, the pattern $m' \times p$ connects any $a', b' \in \mathrm{P}_x$ which are connected in $[\![p]\!]$.

Assume $x \in V \setminus X$ for now. We will show that $nm \times p^0$ connects a to b in J by applying the operation t to a specific n-tuple of realizations

$$\mathbf{f}^i = (f^i_{1,1}, \dots, f^i_{1,ml}, f^i_{2,1}, \dots, f^i_{2,ml}, \dots, \dots, f^i_{n,ml}), \quad i \in \{1, \dots, n\}$$

of the pattern $nm \times p$, where l denotes the length of p. These realizations will be built such that

- $\mathbf{f}^i$ connects a to b for every $i \in \{1, \dots, n\}$, i.e., $f^i_{1,1}(x) = a$ and $f^i_{n,ml}(x) = b$, and
- for every $i, j \in \{1, \dots, n\}$ with $i \neq j$, the j-th subpattern $(f^i_{j,1}, \dots, f^i_{j,ml})$ of $\mathbf{f}^i$ is a realization of $m \times p^i$.

Let $i \in \{1, \dots, n\}$. We will now construct $\mathbf{f}^i$. Since $x \in V \setminus X$, we have $a \in \mathrm{P}_x = \mathrm{P}^i_x$ and hence there exists a realization $(f^i_{1,1}, \dots, f^i_{i-1,ml})$ of $(i-1)m \times p^i$ with $f^i_{1,1}(x) = a$. Similarly, there exists a realization $(f^i_{i+1,1}, \dots, f^i_{n,ml})$ of $(n-i)m \times p^i$ with $f^i_{n,ml}(x) = b$. It remains to find a realization of the i-th subpattern $m \times p$ of $\mathbf{f}^i$ that connects $a' := f^i_{i-1,ml}(x)$ to $b' := f^i_{i+1,1}(x)$. Since a' and a, a and b, as well as b and b' are connected in $[\![p]\!]$, also a' and b' are connected in $[\![p]\!]$. By our choice of m, the pattern $m \times p$ connects a' to b'. Choose a realization $(f^i_{i,1}, \dots, f^i_{i,ml})$ of $m \times p$ witnessing this.

As promised, we now define functions $f_{i,j} = t(f^1_{i,j}, \dots, f^n_{i,j})$ for $i \in \{1, \dots, n\}$, $j \in \{1, \dots, ml\}$, think of applying t to the columns of the matrix with rows $\mathbf{f}^1, \dots, \mathbf{f}^n$. We want to prove that $\mathbf{f} = (f_{1,1}, \dots, f_{n,ml})$ is a realization of $nm \times p^0$ connecting a to b. The latter is an immediate consequence of the idempotency of t. Indeed,

$$f_{1,1}(x) = t(f^1_{1,1}(x), \dots, f^n_{1,1}(x)) = t(a, \dots, a) = a$$

and analogously follows $f_{n,ml}(x) = b$. Now, we use our assumption on the subpatterns of $\mathbf{f}^i$. Let $i \in \{1, \dots, n\}$, $j \in \{1, \dots, ml\}$. For every y in the domain of $f_{i,j}$ and for every $i' \in \{1, \dots, n\}$ with $i' \neq i$, we have $f^{i'}_{i,j}(y) \in \mathrm{P}^i_y$. Hence, (D4) yields $f_{i,j}(y) \in \mathrm{P}^0_y$. Furthermore, the constraint relations appearing in p are subpowers of $\mathbf{D}$. This guarantees that $\mathbf{f}$ is a realization of $nm \times p$ and thus even of $nm \times p^0$.

If $x \in X$ and $[\![p]\!] \subseteq X$, then, by (D3) and similar reasoning as in the proof of the 1-minimality, every realization of $m \times p$ witnessing that this pattern connects $a \in \mathrm{P}^0_x$ to $b \in \mathrm{P}^0_x$ is already a realization of $m \times p^0$.

If $x \in X$ and there exists some $y \in [\![p]\!]$ with $y \notin X$, we can write $p = p_1 + p_2$ such that p_1 is the subpattern of p from x to the first appearance of y and p_2 is the subpattern of p starting with y. There exist $a', b' \in \mathrm{P}^0_y$ such that p^0_1 connects a to a' and such that p^0_2 connects b' to b. Hence, $p_2 + p_1$ is a closed pattern from y to y and a', b' are connected in $[\![p_2 + p_1]\!]$. Since $y \notin X$, there exists an integer k such that $k \times (p_2 + p_1)^0$ connects a' to b' by the previous paragraphs. Thus, $(k+1) \times p^0 = p^0_1 + k(p_2 + p_1)^0 + p^0_2$ connects a to b. □

5 Collapse of the Relational Width Hierarchy

Corollary 4.0.6 shows that a constraint language with bounded relational width has relational width $(2,3)$. Strengthening this result, we can prove that any constraint language with bounded relational width has exactly relational width $(2,3)$ or relational width $(1,1)$, excluding the cases of relational width $(2,2)$ or $(1,l)$ for an integer l.

5.1 Relational Width (1,l)

Theorem 5.1.1. *Every finite constraint language with relational width* $(1,l)$ *for an integer* $l \geq 1$ *has relational width* 1.

Proof. Let $\mathbb{D}$ be a finite constraint language with relational width $(1,l)$. Let I be an instance of CSP($\mathbb{D}$) such that $I' = (1,1)$-minimality(I) is non-trivial. By Lemma 2.3.24, it suffices to prove that the instance $(1,l)$-minimality(I') is still non-trivial.

For every set $\{x_1,\dots,x_l\} \subseteq V$ of l variables which is not contained in the scope of a constraint of I', a constraint $C = ((x_1,\dots,x_l), D^l)$ is added at the beginning of the $(1,l)$-minimality algorithm with input I'. Since the order in which constraints are compared in line 12 of the algorithm does not matter, see Lemma 2.3.26, we may assume that every new constraint C will be compared to all constraints of I' at the beginning of the computation and we call the resulting instance I_{min}. The instance I' is $(1,1)$-minimal and hence, the constraint relation of every constraint $C = ((x_1,\dots,x_l), R)$ of I_{min} that is not in I' is exactly

$$R = \mathrm{P}_{x_1}^{(I')} \times \cdots \times \mathrm{P}_{x_l}^{(I')}.$$

None of these relations is empty because I' is non-trivial. Thus, also I_{min} is non-trivial. Moreover, we have $\mathrm{Sol}_{\{x\}}(C_1) = \mathrm{Sol}_{\{x\}}(C_2)$ for all constraints C_1, C_2 of I_{min}. Hence, (M2') holds for I_{min} and I_{min} is $(1,l)$-minimal. □

5.2 Relational Width (2,2)

For the proof of the analogous statement for relational width $(2,2)$, we follow the idea in [Dal09].

Theorem 5.2.1. *Every finite constraint language with relational width* $(2,2)$ *has relational width* $(1,1)$.

The strategy of the proof can be outlined by two steps. Theorem 5.2.11 yields a characterization of relational width (k,k) which then enables the use of the Sparse Incomparability Lemma, Theorem 5.2.4.

Definition 5.2.2. Let $\mathbb{E}$ be a relational structure. If $s_i = (x_1^{(i)}, \dots, x_{k_i}^{(i)})$ are tuples of relations R_i from $\mathbb{E}$ for $i = 0, \dots, m-1$, the pairs (s_i, R_i) are pairwise distinct and the set $\{x_j^{(i)} : 0 \leq i \leq m-1,\ 1, \leq j \leq k_i\}$ has at most $\sum_{i=0}^{m-1}(k_i - 1)$ elements, then $s_0, \dots, s_{m-1}$ is called a *circle of length* m. The *girth* of $\mathbb{E}$ denotes the length of its shortest circle.

Remark 5.2.3. For every integer $m \geq 2$, the above definition for girth m of a relational structure $\mathbb{E}$ coincides with the definition of girth m for $\mathbb{E}$, interpreted as an instance, given in Definition 3.1.3. The case $m = 1$ is impossible in Definition 3.1.3 due to the requirement that variables that are contained in scopes must be pairwise distinct.

Theorem 5.2.4 (Sparse Incomparability Lemma)**.** *Let k, l be positive integers and let $\mathbb{E}$ be a finite constraint language. Then there exists a finite constraint language $\mathbb{G}$ such that*

1. *$\mathbb{G} \xrightarrow{hom} \mathbb{E}$,*
2. *for every constraint language $\mathbb{D}$ with at most k elements, $\mathbb{E} \xrightarrow{hom} \mathbb{D}$ iff $\mathbb{G} \xrightarrow{hom} \mathbb{D}$, and*
3. *$\mathbb{G}$ has girth at least l.*

Proof. [NR89] □

Definition 5.2.5. Let $\mathbb{A}$ be a relational τ-structure. A *tree-decomposition* of $\mathbb{A}$ is a pair $(\mathcal{T}, \phi)$ where the graph $\mathcal{T} = (T, E)$ is a tree and $\phi : T \to 2^A$ assigns a subset of A to every node of $\mathcal{T}$ such that

(td1) for every $a \in A$, the subgraph of $\mathcal{T}$ generated by $\phi^{-1}(a)$ is connected and hence a tree itself,

(td2) for every relation symbol R of τ and every tuple $s \in R^{\mathbb{A}}$, there exists a node $t_{(s,R)}$ of $\mathcal{T}$ such that $\phi(t_{(s,R)})$ contains the elements of s. The assignment $(s, R) \mapsto t_{(s,R)}$ has to be injective on $\{(s, R) : R \in \tau,\ s \in R^{\mathbb{A}}\}$.

Notation 5.2.6. We will be a bit imprecise and say that t is a *node of a tree-decomposition* $(\mathcal{T}, \phi)$ if t is a node of $\mathcal{T}$. Furthermore, we will say that a node t of a tree-decomposition *contains* a set of elements or just an element of A to abbreviate that $\phi(t)$ contains the respective set or element.

Definition 5.2.7. Let $\mathbb{A}$ be a finite constraint language and let τ be its signature. $\mathbb{A}$ is a *k-relational tree* or *k-reltree* if there exists a tree-decomposition $(\mathcal{T}, \phi)$ of $\mathbb{A}$ such that

(k-rt1) for all nodes $t_1 \neq t_2$ of $\mathcal{T}$, we have $|\phi(t_1) \cap \phi(t_2)| \leq k$,

(k-rt2) for every node t of $\mathcal{T}$, either $|\phi(t)| \leq k$ or there exists a tuple s of some relation of $\mathbb{A}$ such that s contains $\phi(t)$.

For the proof of Theorem 5.2.11, we introduce a recursive definition of k-reltrees.

Definition 5.2.8. Let $\mathbb{A}$ be a finite constraint language and let $L \subseteq A$ with $|L| \leq k$. The pair $(\mathbb{A}, L)$ is called a *recursive k-reltree* if

(rk-rt1) there is at most one tuple in at most one relation of $\mathbb{A}$,

(rk-rt2) there exist a finite index set J, k-reltrees $(\mathbb{A}_j, L_j)$, $j \in J$ and (not necessarily distinct) $a_1, \ldots, a_n \in A$, $n \geq 0$ such that for all $j \in J$, $A_j \cap \{a_1, \ldots, a_n\} \subseteq L_j$ and for all $i, j \in J$, $T_i \cap T_j \subseteq \{a_1, \ldots, a_n\}$, and either

(a) $\mathbb{A} = \bigcup_{j \in J} \mathbb{A}_j$ but one n-ary relation of $\mathbb{A}$ additionally contains the tuple $(a_1, \ldots, a_n)$, and $L \subseteq \{a_1, \ldots, a_n\}$, or

(b) $\mathbb{A} = \bigcup_{j \in J} \mathbb{A}_j$ and $L = \{a_1, \ldots, a_n\}$, or

(rk-rt3) there is a k-reltree $(\mathbb{A}, L')$ with $L \subseteq L'$.

Remark 5.2.9. Let $\mathbb{A}$ be a k-reltree. Then adding or removing elements from the domain of $\mathbb{A}$ that do not occur in any tuple of a relation of $\mathbb{A}$ yields another k-reltree. Similarly, if $(\mathbb{A}, L)$ is a recursive k-reltree, we can add or remove elements that do not occur in L or in any tuple of a relation of $\mathbb{A}$ and obtain another recursive k-reltree.

Lemma 5.2.10. *A finite constraint language $\mathbb{A}$ is a k-reltree iff $(\mathbb{A}, \emptyset)$ is a recursive k-reltree.*

Proof. We will prove that for every recursive k-reltree $(\mathbb{A}, L)$, there exists a tree-decomposition $(\mathcal{T}, \phi)$ of $\mathbb{A}$ such that the set L is contained in some node of $\mathcal{T}$ by induction on the construction of $(\mathbb{A}, L)$ via the rules (rk-rt1),(rk-rt2), and (rk-rt3).

If $\mathbb{A}$ contains only one tuple s, the graph $\mathcal{T}$ shall consist of only two connected nodes t_1, t_2, where t_1 contains exactly the elements of s and t_2 the elements of L.

Assume that J is a finite index set and that $(\mathbb{A}, L)$ is built from finitely many recursive k-reltrees $(\mathbb{A}_j, L_j)$, $j \in J$ using rule (rk-rt2) part (a) or (b) with the elements $a_1, \ldots, a_n \in A$. Let $(\mathcal{T}_j, \phi_j)$ be an appropriate tree-decomposition for $(\mathbb{A}_j, L_j)$ for every $j \in J$. Then $(\mathcal{T}, \phi)$ is gained by adding a new node t, defining $\phi(t) = \{a_1, \ldots, a_n\}$ and connecting t to the respective nodes of $\mathcal{T}_j$ containing L_j for $j \in J$.

The third case is easy, just take the same tree-decomposition as for $(\mathbb{A}, L')$.

For the converse implication, let $(\mathcal{T}, \phi)$ be a suitable tree-decomposition of $\mathbb{A}$ with a minimum number of nodes. Let τ be the signature of $\mathbb{A}$. For each relation symbol $R \in \tau$ and every tuple s of $R^{\mathbb{A}}$, there exists a distinguished node $t_{(s,R)}$ of $\mathcal{T}$ containing the elements of s by (td2). Denote the set of such vertices by S. By the definition of k-reltrees, every node $t_{(s,R)}$ in S is either of size at most k or contained in s. Thus, if a node $t_{(s,R)}$ in S has more than k elements, $t_{(s,R)}$ and s share the same elements.

Choose a node t_0 of $\mathcal{T}$ and think of $\mathcal{T}$ as a rooted tree with root t_0. Let M be the length of the longest path in $\mathcal{T}$ starting from t_0. For every $m \in \{0, \ldots, M\}$, denote the set of nodes that can be reached from t_0 by a path of length m by V_m. Furthermore, for every $t_m \in V_m$, we write $\mathcal{T}_{t_m}$ for the rooted subtree of $\mathcal{T}$ with root t_m, where $\mathcal{T}_{t_m}$ contains the nodes in $V_{m+1} \cup \cdots \cup V_M$ that can only be reached from t_0 by a path through t_m. We show by induction on $m \in \{M, M-1, \ldots, 0\}$ that for every $t_m \in V_m$, the graph $\mathcal{T}_{t_m}$ corresponds to a recursive k-reltree $(\mathbb{A}_{t_m}, L_{t_m})$ such that

- $L_{t_m} = \emptyset$ if $m = 0$ and $L_{t_m} = \phi(t_m) \cap \phi(t'_m)$, where t'_m denotes the first node on the path from t_m to t_0, otherwise, and

- the τ-structure $\mathbb{A}_{t_m}$, roughly said, includes all the tuples s that occur in $\mathcal{T}_{t_m}$.

V_M only contains leaves of $\mathcal{T}$, not necessarily all of them. Due to the minimality of $\mathcal{T}$, each node t_M of V_M is in S. Fix a node $t_M \in V_M$ and let s be the tuple and R be the relational symbol such that $t_M = t_{(s,R)}$ holds with the notation in the definition of a tree-decomposition. We construct a recursive k-reltree $(\mathbb{A}_{t_M}, L_{t_M})$. The constraint language $\mathbb{A}_{t_M}$ has the domain $\phi(t_M)$ and empty relations except $R^{\mathbb{A}_{t_M}} = \{s\}$. We set $L_{t_M} = \emptyset$ if $M = 0$ and we set $L_{t_M} = \phi(t_M) \cap \phi(t_M{}')$, where $t_M{}' \in V_{M-1}$ denotes the first node on the path from t_M to t_0, otherwise.

Let $m \leq M-1$, let $t_m \in V_m$, and let J be the set of its adjacent vertices in V_{m+1}. If J is empty, t_m is a leave and we construct the recursive k-reltree as above. If J is nonempty, let $(\mathbb{A}_j, L_j)$, $j \in J$ be appropriate recursive k-reltrees provided by induction hypothesis. We set $\{a_1, \ldots, a_n\} = \phi(t_m)$.

If t_m is not in S, we apply item (b) of (rk-rt2) to obtain the recursive k-reltree $(\mathbb{A}_{t_m}, L_{t_m})$.

If $t_m \in S$ and $t_m = t_{(s,R)}$ for a tuple s and a relation symbol R, there are two cases. If $\phi(t_m)$ is at most of size k and contains elements that are not in s, we enlarge J by one element j' and define a recursive k-reltree $(\mathbb{A}_{j'}, L_{j'})$, where $A_{j'} = L_{j'} = \phi(t_m)$ and $\mathbb{A}_{j'}$ contains only the tuple s in the relation corresponding to R. Then apply item (b) of (rk-rt2). If s contains all the elements of $\phi(t_m)$, we apply item (a) of (rk-rt2). For both cases, $t_m \in S$ and $t_m \notin S$, set $L_{t_m} = \emptyset$ if $m = 0$ and $L_{t_m} = \phi(t_m) \cap \phi(t'_m)$, where t'_m denotes the first node on the path from t_m to t_0, otherwise.

By (td2) and by construction, $\mathbb{A}_{t_0}$ contains exactly the tuples of $\mathbb{A}$ in corresponding relations. Remark 5.2.9 yields that $(\mathbb{A}, \emptyset)$ is a recursive k-reltree. □

Theorem 5.2.11. *Let $\mathbb{D}$ be a finite constraint language with signature τ and let I be an instance of* CSP($\mathbb{D}$). *Write $\mathbb{E}_I$ for I interpreted as a τ-structure. Then the following are equivalent for every integer $k > 0$.*

(i) (k,k)-decision(I) *returns YES.*

(ii) For every k-reltree $\mathbb{A}$, if $\mathbb{A} \overset{hom}{\longrightarrow} \mathbb{E}_I$, then also $\mathbb{A} \overset{hom}{\longrightarrow} \mathbb{D}$.

Proof. By Lemma 2.3.34, (k,k)-decision(I) returns YES iff there exists a (k,k)-minimal family $\mathcal{P}$ for I. We prove by structural induction that if $(\mathbb{A}, L)$ is a recursive k-reltree, f a homomorphism from $\mathbb{A}$ to $\mathbb{E}_I$ and $h \in \mathcal{P}$ a map with domain $f(L)$, then there exists a homomorphism g from $\mathbb{A}$ to $\mathbb{D}$ with $g_{|_L} = h \circ f_{|_L}$. With Lemma 5.2.10 this yields the implication from (i) to (ii).

(rk-rt1) $\mathbb{A}$ has only one tuple $s = (a_1, \ldots, a_n)$ in a relation R and L is a subset of A of size at most k. Let $(x_1, \ldots, x_n) = (f(a_1), \ldots, f(a_n))$ and let $(d_1, \ldots, d_n)$ be the tuple witnessing the first condition of (k,k)-minimal families for h and the constraint $((x_1, \ldots, x_n), R)$. Now, define $g : A \to D$ by $g(a_i) = d_i$ for $i \in \{1, \ldots, n\}$, $g(a) = h \circ f(a)$ for $a \in L$ and arbitrary assignments otherwise.

(rk-rt2) (a) Let $(a_1, \ldots, a_n)$ be the tuple that was added to a relation R. Define $x_i = f(a_i)$ for $i \in \{1, \ldots, n\}$. Let $(d_1, \ldots, d_n)$ be the tuple witnessing the first condition of (k,k)-minimal families for h and the constraint $((x_1, \ldots, x_n), R)$. Now, define

$g(a_i) = d_i$ for $i \in \{1, \dots, n\}$. For the rest of $\mathbb{A}$, consider a $j \in J$. By the second and third property of (k,k)-minimal families, there exists a map $h_j \in \mathcal{P}$ with domain $f(L_j)$ such that $h_j(x_i) = d_i$ for every $i \in \{1, \dots, n\}$ with $x_i \in f(L_j)$. Clearly, $f_{|A_j}$ is a homomorphism from $\mathbb{A}_j$ to $\mathbb{E}_I$ and by induction hypothesis, there exists a homomorphism g_j from $\mathbb{A}_j$ to $\mathbb{D}$ with $(g_j)_{|L_j} = h_j \circ f_{|L_j}$. For every $a \in A_j$, we define $g(a) = g_j(a)$. The map g is a homomorphism, g satisfies $g_{|L} = h \circ f_{|L}$ by construction and g is well-defined by the assumptions $A_j \cap \{a_1, \dots, a_n\} \subseteq L_j$ and $A_i \cap A_j \subseteq \{a_1, \dots, a_n\}$ for every $i \neq j \in J$.

(b) Define $g(a) = h \circ f(a)$ for all $a \in L$ and extend g to all of A as in the previous case.

(rk-rt3) Let $(\mathbb{A}, L')$ be a k-reltree such that $L \subseteq L'$. By the third property of (k,k)-minimal families, there exists an extension h' of h defined on all of $f(L') \supseteq f(L)$. Induction hypothesis yields a homomorphism g from $\mathbb{A}$ to $\mathbb{D}$ with $g_{|L'} = h' \circ f_{|L'}$, hence, also $g_{|L} = h \circ f_{|L}$.

For the proof of the implication from (ii) to (i), we will use the following claim.

Claim 1. Let $W \subseteq V$ be of size at most k and let $h : W \to D$ be a map that is not in $\mathrm{Sol}_W(C)$ at some point during the (k,k)-minimality algorithm for any constraint C. Then there exists a recursive k-reltree $(\mathbb{A}, L)$ as well as a homomorphism f from $\mathbb{A}$ to $\mathbb{E}_I$ which is bijective on L and with $f(L) = W$, and any homomorphism g from $\mathbb{A}$ to $\mathbb{D}$ satisfies

$$g_{|L} \neq h \circ f_{|L}.$$

Let us explain how to make use of the above before proving it. The map $h = \emptyset$ on the empty domain $W = \emptyset$ is in $\mathrm{Sol}_\emptyset(C)$ iff C has a solution. If (k,k)-decision(I) returns NO, the constraint relation of some constraint C will become empty during the computation. Then, we have $h \notin \mathrm{Sol}_\emptyset(C)$ and the above claim yields a recursive k-reltree $(\mathbb{A}, \emptyset)$ and a homomorphism f from $\mathbb{A}$ to $\mathbb{E}_I$ such that for every homomorphism g from $\mathbb{A}$ to $\mathbb{D}$, we have

$$\emptyset = g_{|\emptyset} \neq h \circ f_{|\emptyset} = \emptyset.$$

Hence, there exists no homomorphism g from $\mathbb{A}$ to $\mathbb{D}$ and we proved the contraposition of the missing implication since $\mathbb{A}$ is a k-reltree by Lemma 5.2.10.

Proof of Claim 1. If $W \subseteq V$ is a set of at most k variables and $h : W \to D$ is not an element of $\mathrm{Sol}_W(C)$ for some constraint $C = ((x_1, \dots, x_n), R)$ of I containing W before even starting the algorithm, then set $\mathbb{A}$ to be the relational structure that has domain $\{x_1, \dots, x_n\}$ and only one tuple $(x_1, \dots, x_n)$ in a relation $R^{\mathbb{A}}$ that has the same relation symbol as R, set f to be the identity, and $L = W$. Any homomorphism g from $\mathbb{A}$ to $\mathbb{D}$ with $g_{|W} = h$ would imply $h \in \mathrm{Sol}_W(C)$ by definition.

Now, assume that for fixed W and $h : W \to D$ the constraint $C = (s, R)$, where $s = (x_1, \dots, x_n)$, is the first constraint which is altered during the execution of (k,k)-minimality such that afterwards $h \notin \mathrm{Sol}_W(C)$. Let h_j, $j \in J$ be the set of partial mappings from sets W_j of size at most k contained in s to D which the algorithm already sorted out, i.e., for every $j \in J$, there exists a constraint C_j containing W_j in some earlier stage of the execution of (k,k)-minimality such that $h_j \notin \mathrm{Sol}_{W_j}(C_j)$.

For each $j \in J$, the induction hypothesis provides a recursive k-reltree $(\mathbb{A}_j, L_j)$ and a homomorphism f_j from $\mathbb{A}_j$ to $\mathbb{E}_I$ which is bijective on L_j. Hence, by possibly renaming the elements of A_j, we may assume $L_j = W_j \subseteq \{x_1, \dots, x_n\}$, that $(f_j)_{|_{L_j}} = id_{L_j}$ and that the elements of $A_j \setminus L_j$ do not occur in any other $A_{j'}$ with $j' \neq j$ or in $\{x_1, \dots, x_n\}$. Then $L_j = A_j \cap \{x_1, \dots, x_n\}$ and $A_j \cap A_{j'} \subseteq \{x_1, \dots, x_n\}$. Furthermore, for any homomorphism g_j from $\mathbb{A}_j$ to $\mathbb{D}$, we have $(g_j)_{|_{L_j}} \neq h_j \circ (f_j)_{|_{L_j}} = h_j$.

We construct $(\mathbb{A}, L)$ via item (a) of (rk-rt2), i.e., $\mathbb{A} = \bigcup_{j \in J} \mathbb{A}_j$ and $\mathbb{A}$ additionally contains the tuple $(x_1, \dots, x_n)$ in the relation corresponding to the constraint (s, R) but we set $L = W \subseteq \{x_1, \dots, x_n\}$ using (rk-rt3). Define $f(x) = x$ for $x \in \{x_1, \dots, x_n\}$ and $f(x) = f_j(x)$ whenever $x \in A_j$. Clearly, f is a homomorphism from $\mathbb{A}$ to $\mathbb{E}_I$.

Let g be a homomorphism from $\mathbb{A}$ to $\mathbb{D}$. If $g_{|_L} = h \circ f_{|_L} = h$, then the tuple $a = (g(x_1), \dots, g(x_n))$ was not removed from the constraint (s, R) during the computation of the (k, k)-minimality algorithm since for every $j \in J$, the homomorphisms $g_j := g_{|_{A_j}}$ from $\mathbb{A}_j$ to $\mathbb{D}$ satisfies $(g_j)_{|_{L_j}} \neq h_j$ and hence, for any constraint C_2 containing W_j, we still have $(g_j)_{|_{L_j}} \in \mathrm{Sol}_{W_j}(C_2)$ and $(s, \{a\})$ clearly permits $(g_j)_{|_{L_j}}$, compare line 12 of Algorithm 2. Hence, h would still lie in $\mathrm{Sol}_W(C)$, which contradicts our assumption, or $g_{|_L} \neq h \circ f_{|_L}$. □

Lemma 5.2.12. *A finite constraint language without circles is a 1-reltree*

Proof. Let $\mathbb{A}$ be a constraint language without circles and let $\mathcal{T}$ be its incidence graph. For every node t of $\mathcal{T}$ that is a variable, define $\phi(t)$ to be the set containing only this variable. For every node t of $\mathcal{T}$ that is a scope $s = (x_1, \dots, x_n)$, define $\phi(t) = \{x_1, \dots, x_n\}$. Then $(\mathcal{T}, \phi)$ is a tree-decomposition of $\mathbb{A}$ witnessing that $\mathbb{A}$ is a 1-reltree. □

Lemma 5.2.13. *A 2-reltree with girth 3 or higher does not contain cycles.*

Proof. Let $\mathbb{A}$ be a 2-reltree of girth $m \geq 3$ and $(\mathcal{T}, \phi)$ be a tree-decomposition witnessing this. Assume $s_i = (a^i_1, \dots, a^i_{r_i})$ are tuples of relations of $\mathbb{A}$ for $i \in \{0, \dots, m-1\}$ such that $s_0, \dots, s_{m-1}$ is a circle of minimal length m. Hence, $r_i \geq 2$ for every $i \in \{0, \dots, m-1\}$ and we may order the tuples in a way such that for $i, j \in \{0, \dots, m-1\}$, $i \neq j$, the tuples s_i and s_j share exactly one element iff $j = i + 1$, we will call this element a_i, and share no elements otherwise.

By definition of tree-decomposition, there exists a vertex t_i of $\mathcal{T}$ containing $(a^i_1, \dots, a^i_{r_i})$ for every $i \in \{0, \dots, m-1\}$. For arbitrary but fixed $i \in \{0, \dots, m-1\}$, the node t_i either contains at most 2 elements or is contained in a tuple s of $\mathbb{D}$. If s is not equal to s_i or in a different relation than s_i, the tuples s_i, s would constitute a circle of length 2, contradicting our assumptions. Hence, t_i contains exactly the r_i-many pairwise distinct elements of s_i. We will now utilize the fact that exactly two of the elements of t_i, namely a_i and a_{i-1}, where $a_{-1} := a_{m-1}$, are from $\{a_0, \dots, a_{m-1}\}$.

Consider the unique path in $\mathcal{T}$ from t_0 to t_1 continued by the unique path from t_1 to t_2 and so forth until reaching t_0 again using the path from t_{m-1} to t_0. We start by showing that the path has to reverse after reaching t_1 for the first time. In order to avoid a circle, the path has to pass through the last edge used to originally reach t_1 at some later point, say during the walk from t_i to t_{i+1} for some $i \geq 1$. Every node in the path from t_i to t_{i+1} contains a_i and hence t_1 contains a_i. Thus, $i = 1$ and the path reverses after reaching t_1.

Every vertex in the proceeding path from t_1 to t_2 contains a_1 and hence, this path cannot pass through t_0. The path branches away from the path from t_1 to t_0 at a vertex u. This node u is used in the path from t_0 to t_1 and in the path from t_1 to t_2 and must therefore contain a_0 and a_1. At some later point, say in the path from t_j to t_{j+1} for a $j \geq 2$, u will be passed again. Hence, u also contains a_j. The node u has cardinality at least 3 and must be contained in some tuple s of $\mathbb{A}$ which differs from $s_0, \ldots, s_{m-1}$. Now, s_1 and s constitute a circle of length 2. Hence, $\mathbb{A}$ must not contain circles at all. □

Proof of Theorem 5.2.1. Let $\mathbb{D}$ be a finite constraint language with relational width 2 and let I be an unsolvable instance of CSP($\mathbb{D}$). We will show that I is already refuted by the $(1,1)$-decision algorithm.

Write $\mathbb{E}_I$ for I interpreted as relational structure. Since I does not have a solution, there is no homomorphism from $\mathbb{E}_I$ to $\mathbb{D}$. By the Sparse Incomparability Lemma, Theorem 5.2.4, there exists a finite relational structure $\mathbb{G}$ with girth at least 3 such that $\mathbb{G} \overset{hom}{\longrightarrow} \mathbb{E}_I$ but there is no homomorphism from $\mathbb{G}$ to $\mathbb{D}$. Hence, $\mathbb{G}$, interpreted as an instance of CSP($\mathbb{D}$), is refuted by the $(2,2)$-decision algorithm and by Theorem 5.2.11, there exists a (finite) 2-reltree $\mathbb{A}$ which maps homomorphically to $\mathbb{G}$ but not to $\mathbb{D}$. Choose such $\mathbb{A}$ with a minimal number of elements. We will prove that the girth of $\mathbb{A}$ is at least 3. In that case, $\mathbb{A}$ is cycle-free and thus a 1-reltree by Lemma 5.2.13 and Lemma 5.2.12. Since $\mathbb{A} \overset{hom}{\longrightarrow} \mathbb{E}_I$, by composition of homomorphisms, Theorem 5.2.11 then implies that $(1,1)$-decision(I) returns NO.

We are left to show that $\mathbb{A}$ does not contain cycles of length 1 or 2. Let h be a homomorphism from $\mathbb{A}$ to $\mathbb{G}$. If $\mathbb{A}$ had a cycle of length 1, i.e., a tuple s containing the same element twice, the image of s in $\mathbb{G}$ would also be a cycle of length 1. That is impossible.

Similar reasoning works for a cycle of length 2 consisting of two tuples $s_1 = (a_1, \ldots, a_{r_1})$ and $s_2 = (b_1, \ldots, b_{r_2})$, if those tuples are either from different relations or their images in $\mathbb{G}$ are not equal. If s_1, s_2 both only occur in the same relation R of $\mathbb{A}$, $r_1 = r_2 = r$, and $(h(a_1), \ldots, h(a_r)) = (h(a_1), \ldots, h(a_r))$, we have to find a different reasoning. In order to rule out this case, we will construct a constraint language $\mathbb{A}'$ with less elements than $\mathbb{A}$ which still is a 1-reltree and maps homomorphically to $\mathbb{E}_I$ but not to $\mathbb{D}$, contradicting the minimality of $\mathbb{A}$.

Define $f : A \to A$ by $f(b_i) = a_i$ for $i \in \{1, \ldots, r\}$ and $f(x) = x$ for $x \in A \setminus \{b_1, \ldots, b_r\}$. Since $s_1 \neq s_2$, the map f cannot be surjective. Hence, $f(A)$ has less elements than A. Define the constraint language $\mathbb{A}' := f(\mathbb{A})$. Since $h(a_i) = h(b_i)$ for $i \in \{1, \ldots, r\}$, we have $\mathbb{A}' \overset{hom}{\longrightarrow} \mathbb{G}$. Any homomorphism from $\mathbb{A}'$ to $\mathbb{D}$ would immediately yield one from $\mathbb{A}$ to $\mathbb{D}$ via composition. We are left to show that $f(\mathbb{A})$ is a 2-reltree. The tuples s_1 and s_2 build a cycle. Hence, they share at least 2 elements, in fact, they share exactly 2 elements because $\mathbb{A}$ is a 2-reltree. Call those elements a_1 and a_2. Let $(\mathcal{T}, \phi)$ be a tree-decomposition witnessing that $\mathbb{A}$ is a 2-reltree. Denote v_1 and v_2 the vertices of $\mathcal{T}$ containing s_1 and s_2, respectively. Define V_1 to be the set of nodes of $\mathcal{T}$ reachable from v_1 in $\mathcal{T}$ without crossing v_2 and V_2 shall contain all the other nodes of $\mathcal{T}$. Then

- V_1 and V_2 are both connected in $\mathcal{T}$,
- $\bigcup_{v \in V_1} \phi(v) \cap \bigcup_{v \in V_2} \phi(v) = \{a_1, a_2\}$, and
- $\{a_1, \ldots, a_r\} \subseteq \bigcup_{v \in V_1} \phi(v)$ and $\{b_1, \ldots, b_r\} \subseteq \bigcup_{v \in V_2} \phi(v)$.

Let $\mathbb{A}_i$ be the substructure of $\mathbb{A}$ induced by $\bigcup_{v \in V_i} \phi(v)$ for $i \in \{1, 2\}$. Then $f(\mathbb{A}) = \mathbb{A}_1 \cup f(\mathbb{A}_2)$. The structure $\mathbb{A}_1$ is a 2-reltree and since f is injective on A_2, also $f(\mathbb{A}_2)$ is. Let $(\mathcal{T}_1, \phi_1)$ and $(\mathcal{T}_2, \phi_2)$ be tree-decompositions witnessing this. Let u_1 and u_2 be the nodes of $\mathcal{T}_1$ and $\mathcal{T}_2$ containing the elements of $s_1 = f(s_2)$, where f is applied component-wise, provided by (td2). Hence, we have $\phi_1(u_1) = \{a_1, \ldots, a_r\} = \phi_2(u_2)$. Define $\mathcal{T}'$ to be the tree gained from $\mathcal{T}_1 = (T_1, E_1)$ and $\mathcal{T}_2 = (T_2, E_2)$ by glueing together the nodes u_1 and u_2, i.e., the vertex set of $\mathcal{T}'$ is the disjoint union of T_1 and $T_2 \setminus \{u_2\}$. The edges of $\mathcal{T}'$ are the edges of $\mathcal{T}_1$ and $\mathcal{T}_2$ but every occurrence of u_2 in an edge of $\mathcal{T}_2$ is substituted by u_1. We define $\phi'(v) = \phi_1(v)$ for every node v of $\mathcal{T}_1$ and $\phi'(v) = \phi_2$ for every node v of $\mathcal{T}_2$. Then $(\mathcal{T}', \phi')$ is a suitable tree-decomposition of $f(\mathbb{A})$. □

Theorem 5.2.14. *For every finite constraint language $\mathbb{D}$, precisely one of the following statements are true.*

1. *$\mathbb{D}$ has relational width 1.*
2. *$\mathbb{D}$ has relational width $(2, 3)$ and does not have relational width 2, nor $(1, l)$ for any $l \geq 1$.*
3. *$\mathbb{D}$ does not have relational width.*

Proof. Theorem 5.1.1, Theorem 5.2.1 and Corollary 4.0.6. □

6 Summary

In this final chapter, we add one last characterization of finite constraint languages with bounded relational width via the existence of *weak near unanimity* polymorphisms. Equivalent properties characterizing bounded relational width that are covered by this thesis will be summarized in Theorem 6.1.7 and Corollary 6.1.8. We describe an algorithm that decides bounded relational width for finite core constraint languages in polynomial time in the last subsection and show that the assumption that the constraint language is a core cannot be omitted.

6.1 Existence of WNU Operations

Definition 6.1.1. A k-ary *weak near unanimity* operation, *WNU* operation or just *WNU* on a set D is an operation w that satisfies the equations

$$w(y, x, \dots, x) = w(x, y, \dots, x) = \cdots = w(x, x, \dots, x, y)$$

for all $x, y \in D$.

Remark 6.1.2. Note that above definition of WNU operations deviates from the terminology of some literature in that we do not require a WNU operation to be idempotent. That way, WNU operations are defined solely by height 1 identities which are preserved by h1 clone homomorphisms. Our notion of WNU operation is also known as *quasi-WNU* operation in literature. An idempotent relational structure has a WNU polymorphism iff it has an idempotent WNU polymorphism.

Definition 6.1.3. Let v be a 3-ary and let w be a 4-ary WNU operation on a domain D. We say that v and w are *compatible* if $v(y, x, x) = w(y, x, x, x)$ holds for every $x, y \in D$.

The idea of the proof for the next two results is attributed to E. Kiss in [KKVW15].

Lemma 6.1.4. *Let* $\mathbf{D}$ *be a finite algebra with bounded relational width and let* $a \neq b$ *be elements of* D*. Then*

(1) *for every* $k \geq 3$*, there exists a* k*-ary term* $t \in \mathrm{Clo}(\mathbf{D})$ *such that*

$$t(b, a, \dots, a) = t(a, b, \dots, a) = \cdots = t(a, a, \dots, a, b)$$

holds, and

(2) *there exist a* 3*-ary term* s_3 *and* 4*-ary term* s_4 *in* $\mathrm{Clo}(\mathbf{D})$ *such that*

$$s_3(b, a, a) = s_3(a, b, a) = s_3(a, a, b) = s_4(b, a, a, a) = \cdots = s_4(a, a, a, b).$$

Proof. Let $a \neq b$ be elements of D and let $k \geq 3$ be an integer. To prove item (1), define a k-ary relation $R_k \in \mathrm{Inv}(\mathbf{D})$ by

$$R_k = \left\{ \begin{pmatrix} t(b,a,\ldots,a) \\ t(a,b,\ldots,a) \\ \vdots \\ t(a,\ldots,a,b) \end{pmatrix} \mid t \in \mathrm{Clo}_k(\mathbf{D}) \right\}.$$

Hence, R_k is the subuniverse of $\mathbf{D}^k$ generated by the elements

$$\mathbf{b}_1 = \begin{pmatrix} b \\ a \\ a \\ \vdots \\ a \end{pmatrix}, \mathbf{b}_2 = \begin{pmatrix} a \\ b \\ a \\ \vdots \\ a \end{pmatrix}, \ldots, \mathbf{b}_k = \begin{pmatrix} a \\ a \\ \vdots \\ a \\ b \end{pmatrix}.$$

It is sufficient to prove that there exists a $c \in D$ such that $(c,c,\ldots,c)^T \in R_k$. To that end, we define a specific instance $I = (V, D, \mathcal{C})$ of $\mathrm{CSP}((D, \{R_k\}))$ and aim to apply the pigeonhole principle on a solution of this instance. We set $N = (k-1)|D| + 1$, $V = \{x_1 \ldots, x_N\}$, and $\mathcal{C}$ shall contain a constraint (s, R_k) for every scope $s \in V^k$ consisting of k pairwise distinct variables.

The instance I is non-trivial and we continue by proving that I is (k', l)-minimal for all integers $0 < k' \leq l \leq k$. Every subset of at most l variables is contained in a constraint of I. The relation R_k is closed under permutations. To show this, let $(a_1, \ldots, a_n) \in R_k$, let $h \in \mathrm{Clo}_k(\mathbf{D})$ be such that

$$\begin{aligned} a_1 &= h(b,a,a,\ldots,a), \\ a_2 &= h(a,b,a,\ldots,a), \\ &\vdots \\ a_k &= h(a,a,\ldots,a,b) \end{aligned}$$

holds, and let σ be a permutation of $\{1,\ldots,n\}$. Then

$$(a_{\sigma(1)}, a_{\sigma(2)}, \ldots, a_{\sigma(k)})^T = h((\mathbf{b}_{\sigma^{-1}(1)}, \mathbf{b}_{\sigma^{-1}(2)}, \ldots, \mathbf{b}_{\sigma^{-1}(k)}))$$

where h is applied row-wise. Hence, the projection of R_k onto at most k' of its coordinates does not depend on the particular choice of coordinates. For any set W of at most k' variables and for constraints C_1, C_2 of I with scopes containing W, this implies $\mathrm{Sol}_W(C_1) = \mathrm{Sol}_W(C_2)$, hence (M2) holds. The property (M3) is satisfied since constraints of I have scopes with pairwise distinct elements. In particular, I is $(2,3)$-minimal and the projection of R_k onto any two of its variables can be written as

$$R_k^{(1,2)} = \left\{ \begin{pmatrix} t(b,a,a,\ldots,a) \\ t(a,b,a,\ldots,a) \end{pmatrix} \mid t \in \mathrm{Clo}_k(\mathbf{D}) \right\}. \tag{6.1}$$

R_k is invariant under $\mathbf{D}$. Thus, the constraint language $(D, \{R_k\})$ has bounded relational width by the assumption on $\mathbf{D}$. Note, however, that I is clearly not (k', l)-minimal for $l > k$ since sets of l variables are no longer contained in the scope of a constraint of I. Even after adding such constraints, these might not have constraint relations that are closed under permutations throughout the algorithm and the above argument cannot be used. Here, our proof relies on Corollary 4.0.6. The language $(D, \{R_k\})$ even has relational width $(2, 3)$. Hence, I has a solution $f : V \to D$. The pigeonhole principle yields the existence of a $c \in D$ and at least k variables $x_1, \ldots, x_k$ with $f(x_1) = \cdots = f(x_k) = c$. Now, $(c, c, \ldots, c) \in R_k$ holds since f solves the constraint $((x_1, \ldots, x_k), R_k)$.

The proof of item (2) can be done in a very similar way using the relations R_3 and R_4 and the instance $J = (V, D, \mathcal{C})$ with $3(|D|) + 1$ variables, a constraint (s_3, R_3) for every 3-ary scope $s_3 \in V^3$ with pairwise distinct entries and a constraint (s_4, R_4) for every 4-ary scope $s_4 \in V^4$ with pairwise distinct entries. The instance is again $(2, 3)$-minimal. Indeed, property (M1) is clearly satisfied and we have already argued (M2) for constraints of the form $C_1 = (s_1, R_3)$ and $C_2 = (s_2, R_3)$ of I, and of the form $C_1 = (s_1, R_4)$ and $C_2 = (s_2, R_4)$. For the case $C_1 = (s_1, R_3)$ and $C_2 = (s_2, R_4)$ or vice versa, notice that for every $t \in \mathrm{Clo}_k(\mathbf{D})$, the operation $t_3 = t(\pi^3_1, \pi^3_2, \pi^3_3, \pi^3_3)$, where π^n_m denotes the n-ary projection onto the m-th coordinate, is in $\mathrm{Clo}_3(\mathbf{D})$. Similarly, for every $h \in \mathrm{Clo}_4(\mathbf{D})$, the operation $h_4 = h(\pi^4_1, \pi^4_2, \pi^4_3)$ is in $\mathrm{Clo}_3(\mathbf{D})$. By construction, we have

$$t_3(b, a, a) = t(b, a, a, a),$$
$$t_3(a, b, a) = t(a, b, a, a)$$

for every $t \in \mathrm{Clo}_3(\mathbf{D})$, as well as

$$h_4(b, a, a, a) = h(b, a, a),$$
$$h_4(a, b, a, a) = h(a, b, a)$$

for every $h \in \mathrm{Clo}_4(\mathbf{D})$. With the representation from (6.1), we see $R_3^{(1,2)} = R_4^{(1,2)}$ and property (M2) follows. The instance J has a solution since the relations of (D, R_3, R_4) are invariant under $\mathbf{D}$. Hence, it has bounded relational width and, furthermore, relational width $(2, 3)$, just as in the previous case. By the pigeonhole principle, there exists a $c \in D$ with $(c, c, c) \in R_3$ and $(c, c, c, c) \in R_4$. Hence, suitable terms s_3 and s_4 exist. □

Theorem 6.1.5. *Let $\mathbb{D}$ be a finite constraint language with bounded relational width such that* $\mathrm{Pol}(\mathbb{D})$ *is idempotent. Then* $\mathrm{Pol}(\mathbb{D})$ *has k-ary WNU operations for every integer $k \geq 3$, as well as compatible 3-ary and 4-ary WNUs v and w.*

Proof. The subpowers of $\mathbf{D} = (D, \mathrm{Pol}(\mathbb{D}))$ are pp-definable in $\mathbb{D}$ and hence $\mathbf{D}$ has bounded relational width. Consider the algebra $\mathbf{D'} = (\mathrm{Pol}_2(\mathbb{D}), \mathrm{Pol}(\mathbb{D}))$, where $\mathrm{Pol}(\mathbb{D})$ acts on $\mathrm{Pol}_2(\mathbb{D})$ via generalized composition of polymorphisms. Note that $\mathrm{Pol}_2(\mathbb{D}) \subseteq D^{D^2}$ and that for every k-ary $f \in \mathrm{Pol}(\mathbb{D})$ and $g_1, \ldots, g_k \in \mathrm{Pol}_2(\mathbb{D})$, the generalized composition $f(g_1, \ldots, g_k)$ is exactly the operation sending $(x_1, x_2) \in D^2$ to $f(g_1(x_1, x_2), \ldots, g_k(x_1, x_2))$. This is exactly how f applies to $g_1, \ldots, g_k \in D^{D^2}$ on the power $\mathbf{D}^{D^2}$ of $\mathbf{D}$. Hence, $\mathbf{D'}$ is a subalgebra of $\mathbf{D}^{D^2}$ and has bounded relational width by Corollary 2.4.11.

Now, for an arbitrary integer $k \geq 3$, item (1) of Lemma 6.1.4 applied to $\mathbf{D}'$ with the projections $a = \pi^2_1$ and $b = \pi^2_1$, where $\pi^2_i(x_1, x_2) = x_i$ for all $x_1, x_2 \in D$ and $i \in \{1, 2\}$, yields a k-ary operation $t \in \mathrm{Pol}(\mathbb{D})$ such that

$$t(\pi^2_2, \pi^2_1, \ldots, \pi^2_1) = t(\pi^2_1, \pi^2_2, \ldots, \pi^2_1) = \cdots = t(\pi^2_1, \ldots, \pi^2_1, \pi^2_2)$$

holds. Hence, t is a k-ary WNU operation.

Item (2) of Lemma 6.1.4 with $a = \pi^2_1$ and $b = \pi^2_2$ yields compatible 3-ary and 4-ary WNUs. □

Lemma 6.1.6. *Let p be a prime. Then $\mathrm{Clo}(\mathbb{Z}_p; +)$ does not contain k-ary idempotent WNU operations for any multiple k of p, or compatible 3-ary and 4-ary idempotent WNUs.*

Proof. Any idempotent operation w of $\mathrm{Clo}(\mathbb{Z}_p; +)$ is of the form $w(x_1, \ldots, x_k) = \sum_{i=1}^{k} d_i x_i$. If w is a WNU operation, the equations

$$w(1, 0, 0, \ldots, 0) = w(0, 1, 0, \ldots, 0) = \cdots = w(0, 0, \ldots, 0, 1)$$

and hence $d_1 = d_2 = \cdots = d_k$ hold. This contradicts the idempotency of w if k is a multiple of p since $w(1, 1, \ldots, 1) = kd_1 = 0$ holds in $\mathbb{Z}_p$.

If $v = c_1x_1 + c_2x_2 \cdots + c_3x_3$ and $w = d_1x_1 + d_2x_2 + d_3x_3 + d_4x_4$ are compatible WNUs,

$$v(1, 0, 0) = v(0, 1, 0) = v(0, 0, 1) = w(1, 0, 0, 0) = \cdots = w(0, 0, 0, 1)$$

yields $c_1 = c_2 = c_3 = d_1 = \cdots = d_4$. Now, $v(1, 1, 1) = w(1, 1, 1, 1)$ yields $d_4 = 0$ and hence $v = 0 = w$, contradicting the idempotency. □

Theorem 6.1.7. *For a finite constraint language $\mathbb{D}$ such that $\mathrm{Pol}(\mathbb{D})$ is idempotent, the following are equivalent.*

(I) *$\mathbb{D}$ has bounded relational width.*

(II) *$\mathbb{D}$ has relational width $(2, 3)$.*

(III) *$\mathrm{Pol}(\mathbb{D})$, interpreted as an algebra on D, is $\mathrm{SD}(\wedge)$*

(IV) *$\mathrm{Pol}(\mathbb{D})$ does not have a clone homomorphism to an affine clone.*

(V) *$\mathrm{Pol}(\mathbb{D})$ does not have a minion homomorphism to an affine clone.*

(VI) *$\mathrm{Pol}(\mathbb{D})$ does not have a minion homomorphism to the idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$ for any prime p,*

(VII) *$\mathbb{D}$ does not pp-construct $(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ for any prime p.*

(VIII) *$\mathrm{Pol}(\mathbb{D})$ has a k-ary WNU operation for every arity $k \geq 3$.*

(IX) *$\mathrm{Pol}(\mathbb{D})$ has compatible 3-ary and 4-ary WNU operations.*

Proof. The equivalence of (I), (II) and (III) is given by Corollary 4.0.5.

The equivalence of (V), (VI) and (VII) is given by Theorem 3.2.5. The same theorem yields the implication from (I) to either one of them. Clearly, (V) implies (IV). By Lemma 4.0.2, (III) and (IV) are equivalent. Hence, (I) - (VII) are equivalent.

Theorem 6.1.5 shows that (I) implies (VIII) and (IX). If (VIII) or (IX) holds, (VI) must also hold, since any minion homomorphism would map the WNU operations to WNUs in the idempotent reduct of $\mathrm{Clo}(\mathbb{Z}_p; +)$, contradicting Lemma 6.1.6. □

Corollary 6.1.8. *For a finite constraint language $\mathbb{D}$, the following are equivalent.*

(I) $\mathbb{D}$ *has bounded relational width.*

(II) $\mathbb{D}$ *has relational width* $(2, 3)$.

(III) $\mathrm{Pol}(\mathbb{D})$ *does not have a minion homomorphism to an affine clone.*

(IV) $\mathrm{Pol}(\mathbb{D})$ *does not have a minion homomorphism to the idempotent reduct of* $\mathrm{Clo}(\mathbb{Z}_p; +)$ *for any prime p,*

(V) $\mathbb{D}$ *does not pp-construct* $(\mathbb{Z}_p; \mathrm{Aff}(\mathbb{Z}_p))$ *for any prime p.*

(VI) $\mathrm{Pol}(\mathbb{D})$ *has a k-ary WNU operation for every arity $k \geq 3$.*

(VII) $\mathrm{Pol}(\mathbb{D})$ *has compatible* 3*-ary and* 4*-ary WNU operations.*

Proof. The equivalence of (I) and (II) is given by Corollary 4.0.5. Let $\mathbb{D}$ be a finite constraint language and $\mathbb{E}$ the singleton expansion of its core. Then $\mathbb{D}$ pp-constructs $\mathbb{E}$ and vice versa. Hence, $\mathrm{Pol}(\mathbb{D}) \leq_{h1} \mathrm{Pol}(\mathbb{E})$ and $\mathrm{Pol}(\mathbb{E}) \leq_{h1} \mathrm{Pol}(\mathbb{D})$ hold by Theorem 2.1.44. Note that minion homomorphisms map WNU operations to WNU operations. This observation concludes the proof since the items (I) to (VII) are equivalent for the idempotent constraint language $\mathbb{E}$ by Theorem 6.1.7. □

6.2 Deciding Bounded Relational Width

We conclude this chapter by stating an algorithm that decides whether a finite core constraint language has bounded relational width in polynomial time and proving that this decision problem is NP-hard for arbitrary finite constraint languages.

Theorem 6.2.1. *There exists an algorithm that decides whether a given finite core constraint language $\mathbb{D}$ has bounded relational width in polynomial time. If $\mathbb{D}$ has bounded relational width, the algorithm returns compatible* 3*-ary and* 4*-ary WNU polymorphisms of $\mathbb{D}$.*

Proof. Let $\mathbb{D}$ be a finite core constraint language. We can add the equality relation and, since $\mathbb{D}$ is a core, all singleton relations to $\mathbb{D}$ without changing whether $\mathbb{D}$ has bounded relational width or not, by Theorem 2.4.9. The equality relation will be used to define constraints, the singleton relations enable us to predefine values for variables.

By Theorem 6.1.7, $\mathbb{D}$ has bounded relational width iff $\mathrm{Pol}(\mathbb{D})$ contains compatible 3-ary and 4-ary WNU operations. We now construct an instance $I = (V, D, \mathcal{C})$ of $\mathrm{CSP}(\mathbb{D})$ such

that any solution of I corresponds to compatible 3-ary and 4-ary WNU polymorphisms v and w of $\mathbb{D}$. To that end, we set

$$V = \{v_{a,b,c} : a, b, c \in D\} \cup \{w_{a,b,c,d} : a, b, c, d \in D\}.$$

If $f : V \to D$ is a solution of I, the operations

$$v : (a, b, c) \mapsto f(v_{a,b,c}) \quad \text{and} \quad w : (a, b, c, d) \mapsto f(w_{a,b,c,d})$$

will be the proclaimed WNU polymorphisms. We use two kinds of constraints to ensure that v and w are polymorphisms and that they are compatible WNU operations.

For every integer k, every k-ary relation R of $\mathbb{D}$, and tuples $\mathbf{a} = (a_1, \dots, a_k)$, $\mathbf{b} = (b_1, \dots, b_k)$, $\mathbf{c} = (c_1, \dots, c_k) \in R$, we add a constraint

$$C_{R,\mathbf{a},\mathbf{b},\mathbf{c}} = ((v_{a_1,b_1,c_1}, \dots, v_{a_k,b_k,c_k}), R)$$

to $\mathcal{C}$ which ensures that v preserves R. Likewise, we add constraints $C_{R,\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d}}$ to ensure that w is a polymorphism.

We add constraints to enforce the equations that characterize WNUs. For example, for every $a, b \in D$, we add

$$C_{v,b,a,a} = ((v_{b,a,a}, v_{a,b,a}), =),\ C_{v,a,b,a} = ((v_{a,b,a}, v_{a,a,b}), =) \text{ and } C_{a,b} = ((v_{a,a,b}, w_{a,a,a,b}), =)$$

and similar constraints $C_{w,b,a,a,a}, C_{w,a,b,a,a}, C_{w,a,a,b,a}$ to $\mathcal{C}$. By construction, I has a solution iff $\mathbb{D}$ has bounded relational width.

If $\mathbb{D}$ has bounded relational width, it has relational width $(2, 3)$ by Corollary 4.0.6. In this case, the $(2, 3)$-decision algorithm correctly decides the satisfiability of I. If $\mathbb{D}$ does not have bounded relational width, the $(2, 3)$-decision algorithm alone might not decide satisfiability of the instance I correctly as it might return YES even though there is no solution of I. Here comes into play that $\mathbb{D}$ contains the singleton relations. Let $f : V \to D$ be an assignment. The $(2, 3)$-decision algorithm is capable of checking whether f is a solution of I if we add a constraint $((x), \{f(x)\})$ for every variable $x \in V$ to I and run the $(2, 3)$-decision algorithm on this altered instance. This observation motivates the following procedure.

Denote the variables of I by $V = \{x_1, x_2, \dots, x_n\}$. For every $d_1 \in D$, we run the $(2, 3)$-decision algorithm on the instance I_{d_1} gained by adding an additional constraint $x_1 = d_1$ to I until it halts with YES for some d_1 or no elements of D are left to try. In the first case, we continue by running the $(2, 3)$-decision algorithm on I_{d_1} with an additional constraint $x_2 = d_2$ for every $d_2 \in D$, until we find a d_2 such that the algorithm halts with YES, or every possible assignment for x_2 was refuted. This process is repeated for every variable x_i, $i \in \{1, \dots, n\}$ as long as the algorithm returns YES for at least one new fixed value d_i. If the full n steps can be completed, the resulting map $x_i \mapsto d_i$ for $i \in \{1, \dots, n\}$ is a solution of I and hence corresponds to compatible 3-ary and 4-ary WNU polymorphisms of $\mathbb{D}$. The algorithm then returns the answer YES and the WNU polymorphisms. Otherwise, the algorithm stops and terminates with NO as a final answer.

We are left to prove the correctness of this procedure. Assume that $\mathbb{D}$ has bounded relational width. Then the instance I has at least one solution and $\mathbb{D}$ has relational width

$(2,3)$ by Corollary 4.0.6. Hence, for given elements $d_1, d_2, \ldots$ of D, if the $(2,3)$-decision algorithm halts with YES when run on I with the additional constraint $x_1 = d_1$ during the first step, additional constraints $x_1 = d_1$ and $x_2 = d_2$ during the second step and so on, then there really exists a solution f of I with $f(x_1) = d_1$, $f(x_2) = d_2$ and so on. Thus, the procedure described in the previous paragraph necessarily completes all n steps and yields a full solution f of I. If $\mathbb{D}$ does not have bounded relational width, the instance I does not have a solution. While the $(2,3)$-decision algorithm alone might not falsify the instance I, it will eventually stop after $r < n$ steps with additional constraints $x_1 = d_1$, $x_2 = d_2, \ldots, x_r = d_r$ in I because the $(2,3)$-decision algorithm terminates with NO upon adding any further constraint $x_{r+1} = d_{r+1}$ for $d_{r+1} \in D$ to I.

Since the size of I is polynomial in the size of $\mathbb{D}$, the procedure described above correctly decides whether a given finite core constraint language has bounded relational width in polynomial time. □

The assumption that $\mathbb{D}$ is a core played an important role in the proof and cannot be omitted. In fact, deciding bounded relational width is NP-complete for finite constraint languages as we will show now, following [CL17].

Definition 6.2.2. A *strong Maltsev condition* $\mathcal{M}$ is a finite set of identities. A relational structure $\mathbb{D}$ *satisfies* $\mathcal{M}$ if it has polymorphisms that satisfy the equations in $\mathcal{M}$. A strong Maltsev condition $\mathcal{M}$ is

- *non-trivial* if $\mathcal{M}$ cannot be satisfied by projections on a domain with at least 2 elements;
- of *height 1* if all identities in $\mathcal{M}$ are of height 1;
- *consistent* if for every non-empty finite set D, there exist idempotent operations on D that satisfy the identities in $\mathcal{M}$.

Example 6.2.3. The strong Maltsev condition $\mathcal{M}_{BW}$ given by the identities

$$\forall x, y : v(y, x, x) = w(y, x, x, x)$$
$$\forall x, y : v(y, x, x) = v(x, y, x) = v(y, x, x)$$
$$\forall x, y : w(y, x, x, x) = w(x, y, x, x) = w(x, x, y, x) = w(x, x, x, y)$$

is of height 1 and non-trivial. On a given finite set $D = \{d_1, \ldots, d_n\}$, we can define a strict linear order by $d_i <_D d_j$ iff $i < j$ for all $i, j \in \{1, \ldots n\}$. Then the operations $v(x_1, x_2, x_3) = \max\{x_1, x_2, x_3\}$ and $w(x_1, x_2, x_3, x_4) = \max\{x_1, x_2, x_3, x_4\}$ on D are idempotent and satisfy $\mathcal{M}_{BW}$. Hence, $\mathcal{M}_{BW}$ is consistent. By Corollary 6.1.8, a finite constraint language has bounded relational width iff it satisfies $\mathcal{M}_{BW}$.

Remark 6.2.4. 3-colourability, see Example 2.2.10, is NP-complete, even when restricted to connected, undirected graphs without loops and with at least 2 vertices, see, e.g., [Pap94].

Lemma 6.2.5. *Let $\mathbb{D}$ be a finite constraint language and let $\mathcal{M}$ be a strong Maltsev condition of height 1. Then $\mathbb{D}$ satisfies $\mathcal{M}$ iff its core satisfies $\mathcal{M}$ by idempotent polymorphisms.*

Proof. Let $\mathbb{E}$ be the singleton expansion of the core $\mathbb{C}$ of $\mathbb{D}$. Then $\mathbb{D}$ and $\mathbb{E}$ are pp-constructible from each other. By Theorem 2.1.44, we have $\mathrm{Pol}(\mathbb{D}) \leq_{h1} \mathrm{Pol}(\mathbb{E})$ and $\mathrm{Pol}(\mathbb{E}) \leq_{h1} \mathrm{Pol}(\mathbb{D})$. Hence, $\mathbb{D}$ satisfies $\mathcal{M}$ iff $\mathbb{E}$ satisfies $\mathcal{M}$ since $\mathcal{M}$ is of height 1. To conclude the proof, note, that $\mathrm{Pol}(\mathbb{E})$ is exactly the idempotent reduct of $\mathrm{Pol}(\mathbb{C})$. □

Lemma 6.2.6. *Let $\mathcal{G} = (G, E)$ be a finite, connected, undirected graph without loops and with at least 2 vertices. Then there exists a finite constraint language $\mathbb{D}$ containing only binary relations such that $\mathbb{D}$ is computable in polynomial time from $\mathcal{G}$ and the following properties hold.*

(i) *$\mathcal{G}$ is 3-colourable iff $\mathbb{D}$ is not a core.*

(ii) *Denote the core of $\mathbb{D}$ by $\mathbb{C}$. If $\mathcal{G}$ is 3-colourable, every idempotent operation on C is a polymorphism of $\mathbb{C}$.*

(iii) *There exists a 3-element subset S of D such that the restriction of any idempotent polymorphism of $\mathbb{D}$ to S is a projection on S.*

Proof. The domain of $\mathbb{D}$ is $D = E \times \{1, 2, 3\}$. For an element $(u, i) \in D$, we will write u_i since this is more convenient. Choose an arbitrary orientation of the edges E of $\mathcal{G}$, and define a binary relation R_e on D for every edge $e = (u, v) \in E$, by

$$R_e = \{(u_1, v_2), (u_1, v_3), (u_2, v_1), (u_2, v_3), (u_3, v_1), (u_3, v_2)\}.$$

Then set $\mathbb{D} = (D, (R_e)_{e \in E})$. This language is computable in polynomial time from $\mathcal{G}$.

(i) Assume $\mathcal{G}$ is 3-colourable. We want to show that $\mathbb{D}$ is not a core. To that end, let $\phi : G \to \{1, 2, 3\}$ be a 3-colouring. Then the map $r : D \to D$ defined by $r(u_i) = u_{\phi(u)}$ for $i \in \{1, 2, 3\}$ is an endomorphism of $\mathbb{D}$. Indeed, for each edge $e = (u, v)$ of $\mathcal{G}$ we have $\phi(u) \neq \phi(v)$ and $(r(u_i), r(u_j)) = (u_{\phi(u)}, v_{\phi(v)}) \in R_e$. Hence, r preserves R_e.

In particular, r is a homomorphism from $\mathbb{D}$ to the proper, induced substructure $\mathbb{C}$ of $\mathbb{D}$ with domain $C = \{u_{\phi(u)} : u \in G\}$. Thus $\mathbb{D}$ is not a core. In fact, $\mathbb{C}$ is the core of $\mathbb{D}$: for every edge $e = (u, v)$ of $\mathcal{G}$, the restriction of R_e to C^2 is the singleton relation $\{(u_{\phi(u)}, v_{\phi(v)})\}$. Since $\mathcal{G}$ is connected and contains at least 2 vertices, C contains at least 2 elements and every element of C occurs in at least one singleton relation of $\mathbb{C}$. Hence, endomorphisms of $\mathbb{D}$ must fix every element of C.

For the converse implication, assume that $\mathbb{D}$ is not a core. We want to prove that $\mathcal{G}$ is 3-colourable. Since $\mathbb{D}$ is not a core, there exists an endomorphism f of $\mathbb{D}$ that is not injective. For every $u \in G$, write

$$F(u) = \{f(u_1), f(u_2), f(u_3)\}$$

for the image of f on $\{u_1, u_2, u_3\}$. Note, that for every vertex $u \in G$, there exists an edge e of $\mathcal{G}$ such that u is incident to e. The set $\{u_1, u_2, u_3\}$ is a projection of R_e and hence invariant under $\mathrm{Pol}(\mathbb{D})$. In particular, $F(u) \subseteq \{u_1, u_2, u_3\}$ holds. Since the sets $\{u_1, u_2, u_3\}$, $u \in G$ are pairwise disjoint and since f is not injective, there exists a vertex $w \in G$ such that $F(w)$ has size at most 2.

Claim 1. Let $e = (u, v)$ be an edge of $\mathcal{G}$, and let $\{i, j, k\} = \{1, 2, 3\}$. Then both of the following implications hold, even for reversed roles of u and v.

(1) If $u_i, u_j \in F(u)$, then $v_k \in F(v)$.

(2) If $u_i \in F(u)$, then $v_j \in F(v)$ or $v_k \in F(v)$.

Proof of Claim 1. To prove the first part, let $s, t \in \{1, 2, 3\}$ be such that $f(u_s) = u_i$ and $f(u_t) = u_j$. For $r \in \{1, 2, 3\} \setminus \{s, t\}$, we have (u_s, v_r), $(u_t, v_r) \in R_e$ and hence $(u_i, f(v_r))$, $(u_j, f(v_r)) \in R_e$ which yields $f(v_r) = v_k$. The second part follows since $f(v_s) = v_i$ and $r \in \{1, 2, 3\} \setminus \{i\}$ yields $(u_i, f(v_r)) \in R_e$ and hence $f(v_r) = v_j$ or $f(v_r) = v_k$ holds. ■

If there exists a vertex u of $\mathcal{G}$ such that $F(u)$ has size 3 and (u, v) is an edge of $\mathcal{G}$, then $F(v)$ also has size 3 by item (1) of Claim 1. By the connectedness of $\mathcal{G}$ and since $F(w)$ contains at most 2 elements, this yields $|F(u)| \leq 2$ for all $u \in G$.

Define $\phi : G \to \{1, 2, 3\}$ by

$$\phi(u) = \begin{cases} 1, & \text{for } F(u) \in \{\{u_1\}, \{u_1, u_3\}\}, \\ 2, & \text{for } F(u) \in \{\{u_2\}, \{u_1, u_2\}\}, \\ 3, & \text{for } F(u) \in \{\{u_3\}, \{u_2, u_3\}\}. \end{cases}$$

Claim 2. ϕ is a 3-colouring of $\mathcal{G}$.

Proof of Claim 2. Let $e = (u, v)$ be an edge of $\mathcal{G}$ and let $\phi(u) = i$. If $F(u) = \{u_i\}$, then by item (2) of Claim 1, $F(v)$ does not contain v_i. By the definition of ϕ, we have $\phi(v) \neq i = \phi(u)$. If $F(u) = \{u_i, u_j\}$ has size 2, then by item (1) of Claim 1, $F(v)$ must contain v_k for the unique $k \in \{1, 2, 3\} \setminus \{i, j\}$. The definition of ϕ then yields $\phi(u) \neq \phi(v)$. Hence, ϕ is a 3-colouring of $\mathcal{G}$. ■

(ii) If $\mathcal{G}$ is 3-colourable, then every idempotent operation on C is a polymorphism of $\mathbb{C}$ since every relation of $\mathbb{C}$ contains only one tuple.

(iii) Let $u \in G$. We will prove that the restriction of any idempotent polymorphism of $\mathbb{D}$ to $\{u_1, u_2, u_3\}$ is a projection on $\{u_1, u_2, u_3\}$. The following construction is attributed to Feder, [Fed01] in [CL17].

Suppose that u has an outgoing edge (u, v), otherwise switch the direction of all arrows in the gadget in Figure 6.2. This gadget visualizes how one can pp-define the relation $\theta = \{(u_i, v_i) : i \in \{1, 2, 3\}\}$ by a formula $\psi(x, y)$. The 6 outer nodes that are depicted by rectangles correspond to the constants $u_1, u_2, u_3, v_1, v_2, v_3$. The 12 inner nodes that are depicted by a circle correspond to variables. Every arrow stands for a formula $R_e(\alpha, \beta)$ where α is the variable or constant at the shaft of the arrow and β the variable or constant at the head of the arrow. Denote the set of all formulas built this way by Ψ. Then $\psi(x, y)$ is the conjunction of the formulas in Ψ, where x, y are free variables and $u_{12}, u_{13}, v_{12}, v_{13}, u_{21}, u_{22}, u_{23}, v_{21}, v_{22}, v_{23}$ are existentially quantified.

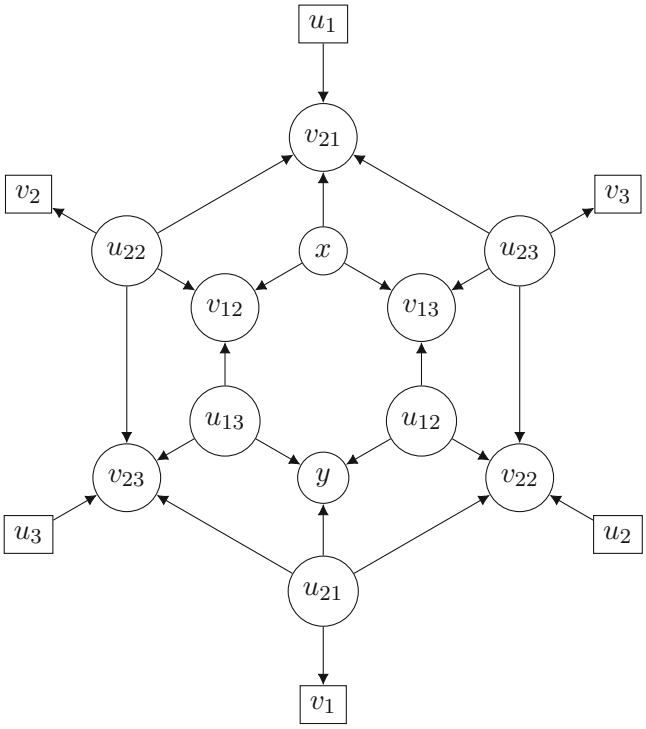

Figure 6.1: The gadget used to define the relation θ in the proof of item (iii) of Lemma 6.2.6.

In order to satisfy ψ, for every $i \in \{1,2,3\}$, the variables u_{2i}, v_{2i} must be assigned the values u_j, v_j for a $j \in \{1,2,3\} \setminus \{i\}$, respectively. Case distinction then yields that ψ indeed defines θ. If u_{21} takes the value u_2, then x and y either take the values u_1 and v_1, or u_3 and v_3. If u_{21} takes the value u_3 then x and y can either take the values u_1 and v_1, or u_2 and v_2.

The pp-formula ψ uses only constants and the relation R_e to define θ. Hence, idempotent polymorphisms of $\mathbb{D}$ preserve θ and the relation $\{(s,t) : \exists z\,(s,z) \in \theta, (t,z) \in R_e\}$ which is the complete graph K_3 on $\{u_1, u_2, u_3\}$. The only idempotent polymorphisms of K_3 are projections, see, e.g., [Bod21, Proposition 6.1.43]. □

Theorem 6.2.7. *Let $\mathcal{M}$ be a non-trivial, consistent, strong Maltsev condition of height 1. The problem of deciding if a finite constraint language satisfies $\mathcal{M}$ is NP-complete.*

Proof. The problem is in NP since we can simply guess polymorphisms and then check whether they satisfy the equations of $\mathcal{M}$ in polynomial time.

We will show that a given graph $\mathcal{G}$ that meets the requirements of Lemma 6.2.6 is 3-colourable iff the language $\mathbb{D}$ associated to $\mathcal{G}$ in Lemma 6.2.6 satisfies $\mathcal{M}$. Since $\mathbb{D}$ is computable in polynomial time from $\mathcal{G}$ and since 3-colourability, even restricted to the considered graphs, is NP-complete, the problem of deciding whether a given finite constraint language satisfies $\mathcal{M}$ must then be NP-hard.

Suppose that $\mathbb{D}$ satisfies $\mathcal{M}$. By Lemma 6.2.5, the core $\mathbb{C}$ of $\mathbb{D}$ satisfies $\mathcal{M}$ with idempotent polymorphisms. If $\mathbb{D}$ would be a core, said idempotent polymorphisms that satisfy

$\mathcal{M}$ would restrict to projections on a 3-element subset S of D by Lemma 6.2.6 (iii). This contradicts that $\mathcal{M}$ is non-trivial. Hence, $\mathbb{D}$ cannot be a core and by Lemma 6.2.6 (i), the graph $\mathcal{G}$ is 3-colourable.

If $\mathbb{D}$ does not satisfy $\mathcal{M}$, then neither does the core $\mathbb{C}$ of $\mathbb{D}$ by Lemma 6.2.5. Since $\mathcal{M}$ is consistent, it is satisfied by idempotent operations on C. If $\mathcal{G}$ was 3-colourable, these idempotent operations would be polymorphisms of $\mathbb{C}$ by Lemma 6.2.6(ii), contradicting that $\mathbb{C}$ does not satisfy $\mathcal{M}$. Hence, $\mathcal{G}$ is not 3-colourable. □

Corollary 6.2.8. *Deciding whether a given finite constraint language has bounded relational width is NP-complete.*

Proof. Finite constraint languages with bounded relational width are exactly those that satisfy the non-trivial, consistent, strong Maltsev condition $\mathcal{M}_{BW}$ of height 1 from Example 6.2.3. The theorem follows from Theorem 6.2.7. □

Bibliography

[Bar14] Libor Barto. The collapse of the bounded width hierarchy. *Journal of Logic and Computation*, 26, 2014.

[BG08] Manuel Bodirsky and Martin Grohe. Non-dichotomies in constraint satisfaction complexity. In *International Colloquium on Automata, Languages, and Programming*, pages 184–196. Springer, 2008.

[Big98] Norman Biggs. Constructions for cubic graphs with large girth. *Electronic Journal of Combinatorics*, 5, 1998.

[Bir35] Garrett Birkhoff. On the structure of abstract algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4):433–454, 1935.

[BJK05] Andrei Bulatov, Peter Jeavons, and Andrei Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM journal on computing*, 34(3):720–742, 2005.

[BK14] Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *Journal of the ACM (JACM)*, 61(1), 2014.

[BKS15] Libor Barto, Marcin Kozik, and David Stanovskỳ. Mal'tsev conditions, lack of absorption, and solvability. *Algebra universalis*, 74(1):185–206, 2015.

[Bod21] Manuel Bodirsky. *Complexity of Infinite-Domain Constraint Satisfaction*. Lecture Notes in Logic. Cambridge University Press, 2021.

[BOP15] Libor Barto, Jakub Opršal, and Michael Pinsker. The wonderland of reflections. *Israel Journal of Mathematics*, 223(1), 2015.

[BP15] Manuel Bodirsky and Michael Pinsker. Topological birkhoff. *Transactions of the American Mathematical Society*, 367(4):2527–2549, 2015.

[Bul17] Andrei A Bulatov. A dichotomy theorem for nonuniform csps. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 319–330. IEEE, 2017.

[CL17] Hubie Chen and Benoit Larose. Asking the metaquestions in constraint tractability. *ACM Transactions on Computation Theory (TOCT)*, 9(3):1–27, 2017.

[Dal09] Víctor Dalmau. There are no pure relational width 2 constraint satisfaction problems. *Information Processing Letters*, 109:213–218, 2009.

[Fed01] Tomás Feder. Classification of homomorphisms to oriented cycles and of k-partite satisfiability. *SIAM Journal on Discrete Mathematics*, 14(4):471–480, 2001.

[FV98] Tomás Feder and Moshe Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28:57–104, 1998.

[HM88] David Hobby and Ralph McKenzie. *The structure of finite algebras*, volume 76. American Mathematical Society, 1988.

[KB12] Marcin Kozik and Libor Barto. Absorbing subalgebras, cyclic terms, and the constraint satisfaction problem. *Logical Methods in Computer Science*, 8(1), 2012.

[KKVW15] Marcin Kozik, Andrei Krokhin, Matt Valeriote, and Ross Willard. Characterizations of several maltsev conditions. *Algebra universalis*, 73(3):205–224, 2015.

[Lad75] Richard E Ladner. On the structure of polynomial time reducibility. *Journal of the ACM (JACM)*, 22(1):155–171, 1975.

[LZ07] Benoit Larose and László Zádori. Bounded width problems and algebras. *Algebra Universalis*, 56:439–466, 2007.

[MM08] Miklós Maróti and Ralph McKenzie. Existence theorems for weakly symmetric operations. *Algebra universalis*, 59(3-4):463–489, 2008.

[NR89] Jaroslav Nešetřil and Vojtěch Rödl. Chromatically optimal rigid graphs. *Journal of Combinatorial Theory, Series B*, 46(2):133–141, 1989.

[Pap94] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[Rob12] Derek JS Robinson. *A Course in the Theory of Groups*, volume 80. Springer Science & Business Media, 2012.

[Sze86] Ágnes Szendrei. Clones in universal algebra. *Les presses de L'universite de Montreal*, 1986.

[Ull88] Jeffrey D Ullman. *Principles of Database and Knowledge base systems, volume I*. Computer Science Press, 1988.

[Zhu20] Dmitriy Zhuk. A proof of the csp dichotomy conjecture. *Journal of the ACM (JACM)*, 67(5):1–78, 2020.