On Redundancy in Constraint Satisfaction Problems

Clément Carbonnel
CNRS, LIRMM, University of Montpellier, France

Abstract

A constraint language Γ has non-redundancy $f(n)$ if every instance of CSP(Γ) with n variables contains at most $f(n)$ non-redundant constraints. If Γ has maximum arity r then it has non-redundancy $O(n^r)$, but there are notable examples for which this upper bound is far from the best possible. In general, the non-redundancy of constraint languages is poorly understood and little is known beyond the trivial bounds $\Omega(n)$ and $O(n^r)$.

In this paper, we introduce an elementary algebraic framework dedicated to the analysis of the non-redundancy of constraint languages. This framework relates redundancy-preserving reductions between constraint languages to closure operators known as pattern partial polymorphisms, which can be interpreted as generic mechanisms to generate redundant constraints in CSP instances. We illustrate the power of this framework by deriving a simple characterisation of all languages of arity r having non-redundancy $\Theta(n^r)$.

2012 ACM Subject Classification Theory of computation → Constraint and logic programming; Mathematics of computing → Discrete mathematics

Keywords and phrases Constraint satisfaction problem, redundancy, universal algebra, extremal combinatorics

Digital Object Identifier 10.4230/LIPIcs.CP.2022.11

Funding This work was supported by the AI Interdisciplinary Institute ANITI, funded by the French program "Investing for the Future – PIA3" under grant agreement no. ANR-19-PI3A-0004. The author also received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 952215.

1 Introduction

The constraint satisfaction problem (CSP) is a fundamental computer science problem with many applications in artificial intelligence and operational research. An instance of the CSP is a set of variables, a set of domain values, and a set of constraints, which are relations imposed upon certain sequences of variables. The goal is to decide whether it is possible to assign domain values to variables in such a way that all constraints are satisfied. The CSP is a natural common framework for a wide variety of well-studied combinatorial problems, such as satisfiability and graph homomorphism, and is in general intractable.

Following early work of Schaefer on the Boolean domain [25], Feder and Vardi initiated the systematic study of CSPs with fixed constraint languages and famously conjectured that all these "non-uniform" CSPs are either polynomial-time solvable or NP-complete [14]. This conjecture prompted a considerable research effort aimed at identifying generic sufficient conditions for the tractability of non-uniform CSPs, which eventually coalesced into a powerful, unified algebraic framework for analysing and classifying the complexity of constraint languages [2, 4]. After more than two decades of research, the Feder-Vardi conjecture was finally settled in the affirmative with two independent proofs by Bulatov [8] and Zhuk [26].

The success and flexibility of the algebraic framework motivated the study of constraint languages from a broader perspective. Beyond the classical "P versus NP-complete" question, classifications of constraint languages have been obtained for a wide variety of properties,

including solvability by specific classes of polynomial-time algorithms [3, 16], membership in fine complexity classes within P [12], learnability [10, 5], definability in certain logics [1, 22], and more.

In this paper we will study non-uniform CSPs from a different perspective. The central question we ask is the following: given a finite constraint language Γ of arity r, what is the maximum number of non-redundant constraints in a CSP instance over Γ? If we denote by n the number of variables, then this quantity (which we call the *non-redundancy* of Γ) is $O(n^r)$, and if Γ is non-trivial (i.e. at least one relation is neither empty nor complete) then it is $\Omega(n)$. As extreme examples, a set of affine relations over a finite field has non-redundancy $\Theta(n)$, while sets of r-clauses are easily seen to have non-redundancy $\Theta(n^r)$. Curiously, very little is known beyond these trivial bounds, especially outside the Boolean domain. The purpose of this paper is to describe an elementary algebraic framework for classifying the non-redundancy of constraint languages, which we illustrate by deriving a simple combinatorial characterisation of r-ary constraint languages with non-redundancy $\Theta(n^r)$.

We draw motivation for studying non-redundancy from two different lines of work. The task of learning a constraint network from answers to queries (sometimes called *constraint acquisition*) has attracted considerable interest in the past decades [7, 10], and a significant effort has been devoted to designing systems that can learn CSPs with as few queries as possible. In this context, it was observed in [5, 6] that the non-redundancy of a language Γ corresponds exactly to its *VC-dimension*, which is a lower bound on the number of yes/no queries (of any kind) that is necessary in order to learn exactly a constraint network over Γ. Therefore, any progress on lower bounds for non-redundancy immediately translates into unconditional, universal lower bounds for constraint acquisition. More generally, for applications where non-uniform CSPs are used to represent knowledge, the non-redundancy of a constraint language is a good estimate of its representational power: if Γ has non-redundancy $f(n)$ and arity r, then the number of n-variable CSP instances over Γ with pairwise distinct solution sets is $\Omega(2^{f(n)})$ and $O(2^{f(n)r\log n})$.

Our second motivation comes from a series of recent results on the sparsification of non-uniform Boolean CSPs [9, 20]. In these papers, the goal is to determine whether there exists a polynomial-time algorithm that takes as input an instance of CSP(Γ) (with up to roughly n^r constraints if Γ has arity r) and outputs an equisatisfiable instance of size $q(n)$, $q(n) = o(n^r)$. On the surface, this question looks quite different from estimating the non-redundancy of Γ: sparsification is in essence an algorithmic question, and sparsification algorithms are not limited to removing redundant constraints because they only have to maintain equisatisfiability. Nevertheless, all sparsification algorithms for NP-hard Boolean CSPs presented in [9, 20] operate purely by removing redundant constraints, and to the best of our knowledge all CSPs whose non-redundancy is known to be $O(n^q)$ also have an $O(n^q)$ sparsification algorithm. While non-redundancy and sparsifiability cannot be equivalent in general (for instance, all polynomial-time non-uniform CSPs have a sparsification algorithm that outputs an instance of size $O(1)$), this suggests that an improved understanding of non-redundancy in constraint languages would help design sparsification algorithms.

Our results

Our first contribution is a generic algebraic framework for the asymptotic study of non-redundancy in non-uniform CSPs. More precisely, we establish a tight connection between redundancy-preserving reductions for constraint languages and *pattern partial polymorphisms*, a type of closure operator that was recently introduced in the context of exponential algorithms for certain classes of non-uniform Boolean CSPs [21]. A key property of this algebraic duality

is that both sides are easily interpretable in terms of non-redundancy. We observe that each pattern partial polymorphism of a constraint language Γ describes a rule to identify (or produce) redundant constraints in CSP instances over Γ. In some cases, knowledge of a single non-trivial pattern partial polymorphism of Γ can be sufficient to establish an improved upper bound on its non-redundancy.

Then, we combine our framework with a theorem of Erdős on the maximum cardinality of K_2^r-free hypergraphs [13] to obtain an explicit characterisation of those constraint languages of arity r having non-redundancy $\Theta(n^r)$. Incidentally, we show the existence of a small gap: either a constraint language of arity r has non-redundancy $\Theta(n^r)$, or it has non-redundancy $O(n^{r-\epsilon})$ for $\epsilon = 2^{1-r}$. This (improperly) extends a result of Chen et al. [9] for Boolean languages, which was obtained using very different methods. Beyond non-redundancy, our main result has direct consequences for sparsification, which will be discussed towards the end of the paper.

Related work

A recent series of papers on the sparsification of Boolean languages have established a number of results on the non-redundancy of constraint languages as byproducts. In [9], Chen et al. show that every Boolean language of arity r that does not contain an r-clause can be expressed using multivariate polynomials of total degree at most $r-1$. Coupled with elementary arguments on Boolean clauses (see e.g. the proof of Lemma 15 in Section 3), this implies that the non-redundancy of any Boolean constraint language of arity at most r is either $\Theta(n^r)$ or $O(n^{r-1})$. Other results in the same paper imply a non-redundancy classification for Boolean constraint languages of arity at most 3, and a characterisation of symmetric Boolean constraint languages with linear non-redundancy. The framework presented in our paper is inspired from their methods, although it is extended to work with arbitrary domains and adapted to study specifically the non-redundancy of constraint languages.

Building upon these results, Lagerkvist and Wahlstrom [20] devised an $O(n)$ sparsification algorithm for the class of languages with a *Mal'tsev embedding*, which generalises linear equations over finite fields. Their algorithm operates by removing redundant constraints, and hence implies a similar bound on the non-redundancy of these languages. To the best of our knowledge, all languages known to have non-redundancy $O(n)$ belong to this class. The same paper also provides a sufficient condition for having non-redundancy $O(n^q), q > 1$ based on the closely related notion of *k-edge embedding*.

Bessiere et al. [5] initiated the direct study of non-redundancy of constraint languages, with a focus on applications in machine learning. They established the equivalence between non-redundancy and VC-dimension, classified the non-redundancy of constraint languages of arity at most 2, and identified a class of ternary constraint languages whose non-redundancy is $o(n^2)$ and cannot be fully determined using results based on algebraic embeddings.

2 Preliminaries

Relations, languages and constraint satisfaction problems

A *relation* R of arity $r = \mathrm{ar}(R)$ over a domain D is a subset of D^r. Given a tuple t of length r and $S \subseteq \{1, \dots, r\}$, we denote by $t[S]$ the tuple obtained from t by discarding elements whose index is not in S. Similarly, the projection on $S \subseteq \{1, \dots, r\}$ of a relation R of arity r is denoted by $R[S] = \{t[S] \mid t \in R\}$. A (finite) *constraint language* Γ is a finite set of

relations over a finite domain D, and the arity of a constraint language Γ is defined as the maximum arity of its relations. Given a constraint language Γ, a CSP instance over Γ is a pair (X, C), where X is a finite set of variables and C is a finite set of constraints, that is, pairs (R, S) with $R \in \Gamma$ and $S \in X^{\mathrm{ar}(R)}$. A *solution* to a CSP instance (X, C) is a mapping $\phi : X \to D$ such that for every $(R, S) \in C$, we have $\phi(S) \in R$. We will denote the set of all solutions to a CSP instance I by $\mathrm{sol}(I)$. The *constraint satisfaction problem over* Γ, denoted by $\mathrm{CSP}(\Gamma)$, takes as input a CSP instance I over Γ and asks whether $\mathrm{sol}(I)$ is non-empty.

Primitive-positive definitions and polymorphisms

Given a constraint language Γ, a relation R of arity r is *primitive-positive definable* (*pp-definable*) over Γ if there exists a first-order formula ψ with r free variables $x_1, \ldots, x_r$ that only uses existential quantification, conjunction, equality, and relations from Γ such that $R = \{(f(x_1), \ldots, f(x_r)) \mid \text{f is a model of } \psi\}$. In that case, we will often write $R(x_1, \ldots, x_r) \equiv \psi$. If ψ is quantifier-free, then R is *qfpp-definable* over Γ. We denote by $\langle\Gamma\rangle$ (resp. $\langle\Gamma\rangle_{\not=}$) the set of all relations that are pp-definable (resp. qfpp-definable) from Γ. It is well-known that $\mathrm{CSP}(\Gamma')$ is log-space reducible to $\mathrm{CSP}(\Gamma)$ for all $\Gamma' \subseteq \langle\Gamma\rangle$ [17]. If in addition we have $\Gamma' \subseteq \langle\Gamma\rangle_{\not=}$, then the reduction is tighter: if $\mathrm{CSP}(\Gamma)$ is solvable in time $O(c^n)$, then so is $\mathrm{CSP}(\Gamma')$ [18].

Given a set D, a *partial operation* over D of arity k is an operation $f : D_f \to D$ with $D_f \subseteq D^k$. Given a relation R of arity r over D, f is a *partial polymorphism* of R if for all tuples $t_1, \ldots, t_k \in R$ such that for all $1 \leq i \leq r$ we have $(t_1[i], \ldots, t_k[i]) \in D_f$, the tuple $f(t_1, \ldots, t_k) = (f(t_1[1], \ldots, t_k[1]), \ldots, f(t_1[r], \ldots, t_k[r]))$ belongs to R. By extension, an operation is a partial polymorphism of a language if it is a partial polymorphism of each of its relations. A *polymorphism* of a relation over D is a partial polymorphism f with $D_f = D^k$. Given a language Γ, we denote by $\mathrm{pol}(\Gamma)$ the set of polymorphisms of Γ.

Geiger's theorem [15] states that for any two languages Γ, Γ' over the same domain, we have $\Gamma' \subseteq \langle\Gamma\rangle$ if and only if $\mathrm{pol}(\Gamma) \subseteq \mathrm{pol}(\Gamma')$. A similar duality was observed between qfpp-definability and partial polymorphisms by Romov [24]. These results form the foundation of the algebraic approach to non-uniform CSPs, in which the complexity of constraint languages is studied through the lens of their (partial) polymorphisms. We refer the reader to recent surveys for a more in-depth treatment of the subject [4][11].

Redundancy

In a CSP instance (X, C), a constraint $c \in C$ is *non-redundant* if and only if (X, C) and $(X, C \backslash \{c\})$ have different solution sets. Given a constraint language Γ, the *non-redundancy* of Γ, denoted by NRD_Γ, is the function that maps each $n \in \mathbb{N}$ to the maximum number of non-redundant constraints in an instance of $\mathrm{CSP}(\Gamma)$ with n variables. It is easily seen that if Γ is a constraint language of arity r that does not contain only empty or complete relations, then $\mathrm{NRD}_\Gamma(n) = O(n^r)$ and $\mathrm{NRD}_\Gamma(n) = \Omega(n)$. It is also known that the asymptotic behaviour of the NRD_Γ function for a finite language Γ is governed by that of its individual relations, as witnessed by these two inequalities:

$$\mathrm{NRD}_\Gamma \leq \sum_{R \in \Gamma} \mathrm{NRD}_{\{R\}} \qquad\qquad \mathrm{NRD}_\Gamma \geq \max_{R \in \Gamma}(\mathrm{NRD}_{\{R\}})$$

The second inequality holds because each instance over $\{R\}$ is also over Γ, and the first holds because the property of being non-redundant is monotone. (If $c = (S, R)$ is non-redundant in I, then it is non-redundant in the subinstance of I consisting only of those

constraints with relation R. Repeating this reasoning with all $R \in \Gamma$ provides the desired upper bound.) Formal proofs can be found in [5]. In this paper we are only interested in the asymptotic behaviour of the NRD_Γ function; it follows from the inequalities above that classifying single-relation languages is sufficient to deduce a classification for all finite constraint languages.

3 Redundancy-preserving reductions

It is easily observed that primitive-positive definability does not preserve non-redundancy in general, in the sense that two constraint languages Γ_1 and Γ_2 with $\Gamma_1 \subseteq \langle\Gamma_2\rangle$ and $\Gamma_2 \subseteq \langle\Gamma_1\rangle$ may have very different non-redundancy asymptotics. (An extreme example is $\Gamma_1 = \{(0,0,1),(0,1,0),(1,0,0)\}$ and Γ_2 being the set of all ternary Boolean clauses. By the results of [9], $\mathrm{NRD}_{\Gamma_1}(n) = \Theta(n)$ but $\mathrm{NRD}_{\Gamma_2}(n) = \Theta(n^3)$. The pp-interdefinability of these languages is well known and can be verified by inspecting Post's lattice [23].) On the other hand, qfpp-definitions do preserve non-redundancy, but have limited expressive power. In this section, we attempt to construct an ideal notion of definability tailored for non-redundancy, with three goals in mind: the corresponding reductions between constraint languages must preserve non-redundancy bounds, a useful algebraic duality must exist, and the framework should be as general as possible.

We start by presenting our proposed notion of definability.

▶ **Definition 1.** *Let D be a set and Γ be a constraint language over D. We say that a relation R of arity r has an* fgpp-definition *over Γ if R has a pp-definition*

$$R(x_1,\dots,x_r) \equiv \exists y_1,\dots,y_q : \psi(x_1,\dots,x_r,y_1,\dots,y_q)$$

over $\Gamma \cup \{Q_g \mid g : D \to D\}$, where $Q_g = \{(d,g(d)) \mid d \in D\}$, and for each existentially quantified variable y_i there exists some x_j such that $Q_g(x_j,y_i)$ is an atom in ψ.

In Definition 1, "fgpp-definition" stands for *functionally guarded pp-definition*. Note that qfpp-definability implies fgpp-definability, but that fgpp-definability does not imply pp-definability in general. (This is due to the functional atoms Q_g, which may not belong to Γ.) On the Boolean domain, fgpp-definitions are equivalent to the *cone-definitions* of Chen et al. [9].

Given a constraint language Γ over D, let $\langle\Gamma\rangle_{\mathrm{fg}}$ denote the set of relations over D that are fgpp-definable over Γ. The next proposition is the first step towards proving that fgpp-definitions are suitable for studying the NRD function.

▶ **Proposition 2.** *Let Γ_1 and Γ_2 be two non-trivial languages over the same finite domain D. If $\Gamma_2 \subseteq \langle\Gamma_1\rangle_{fg}$, then $NRD_{\Gamma_2}(n) = O(NRD_{\Gamma_1}(n))$.*

Proof. Let I be an instance of $\mathrm{CSP}(\Gamma_2)$ with variable set X, $|X| = n$, and exactly $\mathrm{NRD}_{\Gamma_2}(n)$ non-redundant constraints. Without loss of generality, we assume that no constraint in I is redundant.

Let $R \in \Gamma_2$ be some relation and $R(x_1,\dots,x_r) \equiv \exists y_1,\dots,y_q : \psi(x_1,\dots,x_r,y_1,\dots,y_q)$ be an fgpp-definition of R over Γ_1. For each constraint $c_i = (R,(x_1^i,\dots,x_r^i))$ in I, we introduce a set Y^i of q fresh variables $y_1^i,\dots,y_q^i$ and replace c_i with the set of constraints

$$S^i = \{(P,(z_j^1,\dots,z_j^k)) \mid P(z_j^1,\dots,z_j^k) \text{ is an atom in } \psi(x_1^i,\dots,x_r^i,y_1^i,\dots,y_q^i)\}$$

Repeating this process for all $R \in \Gamma_2$ and constraint c_i yields a CSP instance I^* over $\Gamma_1 \cup \{Q_g \mid g : D \to D\}$ whose solution set, when projected onto X, is exactly $\mathrm{sol}(I)$.

By construction, for each $y \in Y = \cup_i Y^i$ there exist $g : D \to D$ and $x \in X$ such that for all $\phi \in \mathrm{sol}(I^*)$, we have $\phi(y) = g(\phi(x))$. In particular, if there exist $y_1, y_2 \in Y$, $x \in X$ and $g : D \to D$ such that $y_1 = g(x)$ and $y_2 = g(x)$ then we have $\phi(y_1) = \phi(y_2)$ for all $\phi \in \mathrm{sol}(I^*)$. It follows that y_1 and y_2 can be merged into a single variable without changing the number of non-redundant constraints in I^*. After exhaustive application of this rule, we have $|Y| \leq n \cdot |D|^{|D|}$.

Now, we greedily remove redundant constraints from I^* until all constraints are non-redundant. Observe that this process cannot remove all constraints from a set S^i, for any i. Indeed, by assumption, for each constraint c_i in I there exists an assignment $\phi : X \to D$ that only violates c_i in I. This assignment can be extended to an assignment $\phi^* : X \cup Y \to D$ that is not a solution to I^* and may only violate constraints in S^i. Therefore, removing all of S_i would increase the solution set of I^*, which cannot happen since only redundant constraints are removed.

In addition, the language $\{Q_g \mid g : D^c \to D\}$ contains only functional constraints and hence has linear non-redundancy. (This follows, for example, from [5, Theorem 13].) Since $|X| + |Y| \leq n \cdot (1 + |D|^{|D|})$, we deduce that I^* contains $O(n)$ constraints that are not from Γ_1.

By the three paragraphs above, I^* has at most $n \cdot (1 + |D|^{|D|})$ variables and at least $\mathrm{NRD}_{\Gamma_2}(n) - O(n)$ non-redundant constraints from Γ_1. By definition of NRD this implies $\mathrm{NRD}_{\Gamma_2}(n) = O(\mathrm{NRD}_{\Gamma_1}(n)) + O(n)$, and finally $\mathrm{NRD}_{\Gamma_2}(n) = O(\mathrm{NRD}_{\Gamma_1}(n))$ since Γ_1 is non-trivial. ◀

▶ **Example 3.** Let $p > 1$ be a prime number, $D = \{0, \ldots, p-1\}$ and consider the relation $R = \{(x, y, z) \mid x^3 + y^3 + z^2 = 1\}$, where sum and product are understood as operations over the finite field of order p. If we let $R_{\mathrm{lin}} = \{(x, y, z) \mid x + y + z = 1\}$ and $f, g : D \to D$ such that $f(d) = d^3$ and $g(d) = d^2$, we can equivalently define R as

$$R(x, y, z) \equiv \exists a, b, c : R_{\mathrm{lin}}(a, b, c) \wedge Q_f(x, a) \wedge Q_f(y, b) \wedge Q_g(z, c)$$

which implies that $R \in \langle \{R_{\mathrm{lin}}\} \rangle_{\mathrm{fg}}$. From Proposition 2 and the fact that linear equations over finite fields have linear non-redundancy, we deduce that $\{R\}$ has non-redundancy $O(n)$.

▶ **Example 4.** Following [20], a language Γ_1 over non-empty domain D_1 has an *embedding* over a language Γ_2 over domain $D_2 \supseteq D_1$ if there exists a bijective function $h : \Gamma_1 \to \Gamma_2$ such that for all $R \in \Gamma_1$, $\mathrm{ar}(R) = \mathrm{ar}(h(R))$ and $R = h(R) \cap D_1$. If we interpret both Γ_1 and Γ_2 as languages over D_2 and define $g : D_2 \to D_2$ such that $g(d) = d$ if $d \in D_1$ and $g(d) = d_1^*$ otherwise (where d_1^* is an arbitrary value in D_1), then each $R \in \Gamma_1$ can be written as

$$R(x_1, \ldots, x_r) \equiv h(R)(x_1, \ldots, x_r) \bigwedge_{1 \leq i \leq r} Q_g(x_i, x_i)$$

and hence $\Gamma_1 \subseteq \langle \Gamma_2 \rangle_{\mathrm{fg}}$. Therefore, by Proposition 2, embeddings preserve the non-redundancy asymptotics of constraint languages.

We will establish an algebraic duality for fgpp-definitions based on *pattern partial polymorphisms*, which were introduced by Lagerkvist and Wahlstrom [21] in a different context (the study of exponential algorithms for sign-symmetric Boolean languages).

A *polymorphism pattern* of arity k is a set of pairs (t, x), where t is a sequence of variables of length k and x occurs in t. A k-ary partial operation $f : D_f \to D$ *satisfies* a k-ary polymorphism pattern P if

$$D_f = \{(\phi(x_1), \ldots, \phi(x_k)) \mid ((x_1, \ldots, x_k), x) \in P, \phi : \{x_1, \ldots, x_k\} \to D\}$$

and $f(\phi(x_1), \ldots, \phi(x_k)) = \phi(x)$ for all $((x_1, \ldots, x_k), x) \in P$, $\phi : \{x_1, \ldots, x_k\} \to D$. It follows from definition that for any pattern P and finite set D, there is at most one partial operation on D that satisfies P. We denote this function by f_P^D and call it the *interpretation* of P on D.

We say that a partial operation f is a *pattern partial operation* if it satisfies some polymorphism pattern P. We will often use the following equivalent characterisation.

▶ **Observation 5.** *Let D be a finite set, k be a nonnegative integer and $D_f \subseteq D^k$. A partial operation $f : D_f \to D$ is a pattern partial operation if and only if for every $t \in D_f$ and $g : D \to D$, we have that $g(t) \in D_f$ and $f \circ g(t) = g \circ f(t)$.*

Proof. Suppose that f is a pattern partial operation because it satisfies a certain polymorphism pattern P. In particular, for every $t \in D_f$ there exists some $((x_1, \ldots, x_k), x) \in P$ and $\phi : \{x_1, \ldots, x_k\} \to D$ such that $t = (\phi(x_1), \ldots, \phi(x_k))$. Then, for any mapping $g : D \to D$ we have $g(t) = (g(\phi(x_1)), \ldots, g(\phi(x_k)))$, which must belong to D_f as witnessed by the mapping $\phi' = g \circ \phi$. Furthermore, by definition we have $f(\phi'(x_1), \ldots, \phi'(x_k)) = \phi'(x)$, or equivalently $f \circ g(t) = g \circ f(t)$.

Conversely, suppose that for every $t \in D_f$ and $g : D \to D$, we have that $g(t) \in D_f$ and $f \circ g(t) = g \circ f(t)$. Let $D^P = \{x_1, \ldots, x_q\}$ be a set of variables in bijection with $D = \{d_1, \ldots, d_q\}$, and let P denote the pattern

$$\{((x_{i_1}, \ldots, x_{i_k}), x_j) \mid (d_{i_1}, \ldots, d_{i_k}) \in D_f, f(d_{i_1}, \ldots, d_{i_k}) = d_j\}$$

Then, we must have $x_j \in \{x_{i_1}, \ldots, x_{i_k}\}$ for any $((x_{i_1}, \ldots, x_{i_k}), x_j) \in P$. Indeed, if it were not the case then there would exist a tuple $t = (d_{i_1}, \ldots, d_{i_k}) \in D_f$ such that $f(t) \notin \{d_{i_1}, \ldots, d_{i_k}\}$, and we would have $f \circ g(t) \neq g \circ f(t)$ for the mapping $g : D \to D$ such that $g(d) = d$ if $d \in \{d_{i_1}, \ldots, d_{i_k}\}$ and $g(d) = d_{i_1}$ otherwise.

Furthermore, mappings ϕ from D^P to D can be identified with mappings from D to D, so with a slight abuse of notation we have

$$f(\phi(x_{i_1}), \ldots, \phi(x_{i_k})) = \phi(f(x_{i_1}, \ldots, x_{i_k})) = \phi(x_j)$$

for all $\phi : D^P$, $((x_{i_1}, \ldots, x_{i_k}), x_j) \in P$ and f satisfies P. ◀

On the Boolean domain, pattern partial operations are called *pSDI operations* [21] (for *partial self-dual idempotent operations*). Beyond the Boolean domain, notable examples of pattern partial operations are the *first Pixley partial operation* of [5] and the *universal Mal'tsev partial operations* of [20], the simplest of which is presented in Example 6.

▶ **Example 6.** Let P_2^M denote the polymorphism pattern

$$((x, x, y), y)$$
$$((y, x, x), y)$$

and consider the partial operation $f_{P_2^M}^D$ over some set D, which is an example of a pattern partial operation with domain $\{(d_1, d_2, d_3) \in D^3 \mid (d_1 = d_2) \text{ or } (d_2 = d_3)\}$. By definition, a binary relation R admits $f_{P_2^M}^D$ as a partial polymorphism if and only if it is *rectangular*, that is, R does not contain three tuples $(a, b), (a, c), (d, c)$ such that $(d, b) \notin R$. It can be further observed (although it is not immediately obvious) that a binary relation admits $f_{P_2^M}^D$ as a partial polymorphism if and only if it is fgpp-definable from the empty constraint language. This polymorphism pattern plays a critical role in the characterisation of the non-redundancy of binary constraint languages obtained in [5], and we will revisit it in the next section.

Throughout this note we will use $\mathrm{p}^2\mathrm{pol}(\Gamma)$ to denote the set of all pattern partial polymorphisms of Γ. The following proposition shows that $\mathrm{p}^2\mathrm{pol}(\Gamma)$ determines precisely the set of relations that are fgpp-definable over Γ.

▶ **Proposition 7.** *Let Γ_1 and Γ_2 be two constraint languages over the same finite domain D. Then, $p^2pol(\Gamma_1) \subseteq p^2pol(\Gamma_2)$ if and only if $\Gamma_2 \subseteq \langle\Gamma_1\rangle_{fg}$.*

Proof. We first prove the backward implication. Suppose that $\Gamma_2 \subseteq \langle\Gamma_1\rangle_{\mathrm{fg}}$ but there exists some pattern partial operation $f \in \mathrm{p}^2\mathrm{pol}(\Gamma_1)$ of arity k that is not a partial polymorphism of some relation $R \in \Gamma_2$. Let $R(x_1,\ldots,x_r) \equiv \exists y_1,\ldots,y_q : \psi(x_1,\ldots,x_r,y_1,\ldots,y_q)$ be an fgpp-definition of R over Γ_1 and define $R_{\not\exists}(x_1,\ldots,x_r,y_1,\ldots,y_q) \equiv \psi(x_1,\ldots,x_r,y_1,\ldots,y_q)$. First, observe that for all $g : D \to D$ and k tuples $t_1 = (d_1, g(d_1)),\ldots,t_k = (d_k, g(d_k))$ of Q_g such that $f(t_1,\ldots,t_k)$ is defined, it holds that

$$f(t_1,\ldots,t_k) = (f(d_1,\ldots,d_k), f(g(d_1),\ldots,g(d_k))) = (f(d_1,\ldots,d_k), g(f(d_1,\ldots,d_k))) \in Q_g$$

so f is a partial polymorphism of $\Gamma_1 \cup \{Q_g \mid g : D \to D\}$. Since $R_{\not\exists}$ is qfpp-definable over $\Gamma_1 \cup \{Q_g \mid g : D \to D\}$, this implies that f is a partial polymorphism of $R_{\not\exists}$. However, f is not a partial polymorphism of R, so there exist $k = \mathrm{ar}(f)$ tuples $t_1,\ldots,t_k \in R$ such that $f(t_1,\ldots,t_k)$ is defined and does not belong to R. Let $t'_1,\ldots,t'_k \in R_{\not\exists}$ be such that $t'_l[1,\ldots,r] = t_l$ for all $l \leq k$. By Definition 1, there exists for each $r < i \leq r+q$ an index $j \leq r$ and a mapping $g : D \to D$ such that $t'_l[i] = g(t'_l[j])$ for all $l \leq k$. Since the domain of f is closed under all unary operations from D to D, $t_f = f(t'_1,\ldots,t'_k)$ is defined and belongs to $R^{\not\exists}$, a contradiction since $t_f[1,\ldots,r] = f(t_1,\ldots,t_k) \notin R = R^{\not\exists}[1,\ldots,r]$.

The forward implication is a bit more difficult. Let $\mathcal{R}$ denote the set of all relations R over D such that $R \notin \langle\Gamma_1\rangle_{\mathrm{fg}}$ and every pattern partial polymorphism of Γ_1 is a partial polymorphism of R. Towards a contradiction, suppose that $\mathcal{R}$ is non-empty. Let R be a relation in $\mathcal{R}$ with minimum arity r. Note that $\langle\Gamma_1\rangle_{\mathrm{fg}}$ contains all unary relations over D, so we may assume that $r \geq 2$. Now, we define

$$\hat{R} = \bigcap_{\substack{Q \in \langle\Gamma_1\rangle_{\mathrm{fg}} \\ R \subseteq Q}} Q$$

and observe that $\hat{R}$ is well defined (because $D^r \in \langle\Gamma_1\rangle_{\mathrm{fg}}$) and strictly contains R. In particular, there exists a certain tuple $t \in \hat{R} \backslash R$. We pick an arbitrary ordering $t_1,\ldots,t_m$ of the tuples of R, and for all $l \leq r$ we define the lth column of R as $c_l = (t_1[l],\ldots,t_m[l])$. Then, we define

$$D_f = \{g(c_l) \mid 1 \leq l \leq r,\, g : D \to D\}$$

and let $p = |D_f|$, as well as $\sigma : D_f \to \{1,\ldots,p\}$ be an arbitrary bijection such that $\sigma^{-1}(i) = c_i$ for $i \leq r$. Now, consider the relation $R_f(y_1,\ldots,y_r) \equiv \exists y_{r+1},\ldots,y_p : \psi(y_1,\ldots,y_p)$, where $\psi(y_1,\ldots,y_p)$ is given by

$$\bigwedge_{\substack{Q \in \Gamma_1 \\ (t^q_1,\ldots,t^q_m) \in Q}} Q(y_{\sigma(t^q_1[1],\ldots,t^q_m[1])},\ldots,y_{\sigma(t^q_1[\mathrm{ar}(Q)],\ldots,t^q_m[\mathrm{ar}(Q)])}) \bigwedge_{\substack{i,j \leq p,\, g : D \to D: \\ \sigma^{-1}(i) = g(\sigma^{-1}(j))}} Q_g(y_i, y_j)$$

and the first conjunction is restricted to tuples of variables that are well-defined with respect to σ. By construction, the tuples of R_f are in one-to-one correspondance with the pattern partial polymorphisms of Γ_1 of arity m whose domain is the closure of $c_1,\ldots,c_r$ under all unary operations $D \to D$. In particular, R_f contains the tuples corresponding to the m partial projection operations on D_f and hence R_f contains R. Then, since R_f is fgpp-definable over Γ_1, it follows that $t \in R_f$. This particular tuple t corresponds to a certain pattern partial polymorphism f_t of Γ_1, of arity m, domain D_f and such that $f(c_l) = t[l]$ for all $l \leq r$. Since $t \notin R$, f_t is not a partial polymorphism of R, which concludes the proof. ◀

4 Pattern partial polymorphisms and redundancy

Recall from Section 2 that in order to study the function NRD_Γ, we can assume without loss of generality that Γ contains a single relation R. Then, it will be convenient to rephrase $\mathrm{CSP}(\Gamma)$ as a homomorphism problem: given a relation R_X over some finite set X of the same arity as R, is there a homomorphism from R_X to R? Here, a homomorphism is a mapping ϕ from X to D such that $\phi(t) \in R$ for all $t \in R_X$. We will use $\hom(R_X, R)$ to denote the set of all homomomorphisms from R_X to R. In this formulation, the constraint scopes are given by the tuples of R_X and a constraint (R, t), $t \in R_X$, is redundant if and only if $\hom(R_X, R) = \hom(R_X \backslash \{t\}, R)$.

▶ **Lemma 8.** *Let R_X, R be relations with respective domains X, D and let f_P^D be a k-ary partial polymorphism of R that satisfies a pattern P. If $t, t_1, \ldots, t_k$ are tuples of R_X such that $t = f_P^X(t_1, \ldots, t_k)$, then $hom(R_X, R) = hom(R_X \backslash \{t\}, R)$.*

Proof. For the sake of contradiction, suppose that there exists a homomorphism $h : X \to D$ such that $h(t) \notin R$ but $h(t_1), \ldots, h(t_k) \in R$. Observe that $f_P^{X \cup D}$ is a partial polymorphism of R (when interpreted as a relation over $X \cup D$) and define $g : X \cup D \to X \cup D$ such that $g(u) = h(u)$ if $u \in X$ and $g(u) = u$ otherwise. Since $f_P^{X \cup D}$ is a pattern partial operation, we have that

$$f_P^{X \cup D}(g(t_1), \ldots, g(t_k)) = g(f_P^{X \cup D}(t_1, \ldots, t_k)) = g(f_P^X(t_1, \ldots, t_k)) = g(t) = h(t) \notin R$$

which contradicts the fact that $f_P^{X \cup D}$ is a partial polymorphism of R. ◀

In essence, a (partial) polymorphism is an operator that combines solutions (tuples of values) to produce new ones. What this lemma says is that *pattern* partial polymorphisms can also be used to combine *constraints* and produce new ones that are valid for the instance, i.e. redundant. The is particularly interesting in light of the algebraic duality uncovered in Proposition 7: if Γ can fgpp-define a relation R with high non-redundancy, then Γ has high non-redundancy by Proposition 2, and if it cannot then Proposition 7 and Lemma 8 provide a non-trivial mechanism to identify redundant constraints that is valid for $\mathrm{CSP}(\Gamma)$ but not for $\mathrm{CSP}(\{R\})$.

▶ **Example 9.** Let R be a relation with the operation $f_{P_2^M}^D$ of Example 6 as partial polymorphism. Consider a CSP instance (R_X, R) and suppose that there exist four variables $x_1, x_2, y_1, y_2 \in X$ such that $(x_1, y_1), (x_1, y_2), (x_2, y_2), (x_2, y_1)$ are tuples of R_X (i.e. are scopes of constraints with relation R). Then, the pattern partial polymorphism $f_{P_2^M}^D$ combined with Lemma 8 implies that the constraint $(R, (x_2, y_1))$ is redundant, as it is the image through $f_{P_2^M}^X$ of the first three constraints.

Given a relation R over a set X and a set $\mathcal{F}$ of partial operations on X, we denote by $\mathcal{F}(R)$ the transitive closure of R under operations from $\mathcal{F}$. If no tuple t of R can be generated from tuples in $R \backslash \{t\}$ via an operation in $\mathcal{F}$, we say that R is *$\mathcal{F}$-independent*. The following two propositions are natural consequences of Lemma 8 regarding upper bounds on the NRD function.

▶ **Proposition 10.** *Let R be a relation over a set D, P_R be the set of polymorphism patterns that are satisfied by partial polymorphisms of R, and P_R^S denote the set of interpretations of P_R on set S. If for every relation R_X over a set X of n elements such that $ar(R_X) = ar(R)$ there exists a relation R_X^* of cardinality at most $f(n)$ such that $R_X^* \subseteq R_X \subseteq P_R^X(R_X^*)$, then $NRD_{\{R\}}(n) \leq f(n)$.*

Proof. Suppose that such a relation R^*_X exists for every relation R_X. Let (R_X, R) be an instance of CSP($\{R\}$) and $R_1 \subset R_2 \subset \ldots \subset R_q$ be the sequence of distinct relations obtained by transitive closure of R^*_X under P^X_R, with $R_1 = R^*_X$ and $R_j = R_X$. For every $1 \leq i < q$, there exists a pattern $P \in P_R$ and tuples $t_1, \ldots, t_k$ in R_i such that $R_{i+1} = R_i \cup \{t\}$, $t = f^X_P(t_1, \ldots, t_k)$. By Lemma 8, we have $\hom(R_i, R) = \hom(R_{i+1}, R)$. This is true for all i, so in particular we have $\hom(R^*_X, R) = \hom(R_X, R)$. Therefore, every non-redundant constraint in (R_X, R) must be of the form (R, t) with $t \in R^*_X$, and their total number is at most $f(n)$. ◀

▶ **Example 11.** Consider a relation R of arity r over a set D, and suppose that R has the pattern partial polymorphism $f^D_{P^M_2}$ of Examples 6 and 9. We will use Proposition 10 to show that $\mathrm{NRD}_{\{R\}}(n) \leq 2n^q$, where $q = \lceil r/2 \rceil$.

Let (R_X, R) be an instance of CSP($\{R\}$) with n variables and no redundant constraint. Let R^1_X denote the projection of R_X onto its first q indices and R^2_X be its projection onto the remainder. For simplicity, we will interpret R_X as a binary relation over disjoint domains R^1_X and R^2_X. Let G_X be the bipartite graph with domain $R^1_X \cup R^2_X$ and edge relation R_X. Observe that for any path $(v_1, v_2), (v_2, v_3), \ldots, (v_{k-1}, v_k)$ in G_X with an odd number of edges, repeated application of $f^X_{P^M_2}$ on the corresponding tuples of R_X eventually produces the edge (v_1, v_k). Therefore, the smallest subrelation R^*_X of R_X that contains R_X in its transitive closure via $f^X_{P^M_2}$ corresponds to a forest subgraph F_X of G_X. In particular, $|R^*_X| = |E(F_X)| \leq |R^1_X| + |R^2_X| \leq 2n^q$, and by Proposition 10, $\mathrm{NRD}_{\{R\}}(n) \leq 2n^q$.

▶ **Proposition 12.** *Let R be a relation over a set D, P_R be the set of polymorphism patterns that are satisfied by partial polymorphisms of R, and P^S_R denote the set of interpretations of P_R on set S. If every relation R_X over a set X of n elements that is P^X_R-independent and such that $ar(R_X) = ar(R)$ has cardinality at most $f(n)$, then $NRD_{\{R\}}(n) \leq f(n)$.*

Proof. Follows immediately from Proposition 10 as every minimal relation R^*_X with $R^*_X \subseteq R_X \subseteq P^X_R(R^*_X)$ is P^X_R-independent. ◀

We conclude this section with a straightforward lower bound on the non-redundancy of constraint languages that do not admit a certain polymorphism pattern related to the fgpp-definability of k-clauses, whose properties are well known on the Boolean domain.

▶ **Definition 13.** *Let $k \geq 2$ and $c_1, \ldots, c_{2^k-1}$ be the lexicographic ordering of the relation $\{x, y\}^k \backslash \{(y, \ldots, y)\}$ with respect to $y > x$. The* k-universal polymorphism pattern P^u_k *is the set of all pairs (t_i, y) with $t_i = (c_1[i], \ldots, c_{2^k-1}[i])$, $i \leq k$.*

The interpretation of P^u_k on the Boolean domain is called the *Boolean k-universal partial operation* [21]. In the definition above, the ordering of $\{x, y\}^k \backslash \{(y, \ldots, y)\}$ is not important: a different ordering would produce a different pattern, but it would be equivalent in the sense that it would have the same interpretation on all sets, up to a permutation of the arguments. (For instance, the pattern P^M_2 of Example 6 is equivalent to P^u_2.)

▶ **Example 14.** P^u_3 is the pattern given by the following pairs:

$((x, x, x, x, y, y, y), y)$
$((x, x, y, y, x, x, y), y)$
$((x, y, x, y, x, y, x), y)$

Observe that the left-hand side of these pairs corresponds to the three columns of the relation corresponding to a 3-clause with no negated literals, modulo the renaming $x \leftarrow 1$, $y \leftarrow 0$. The right-hand side is the missing tuple $(0, 0, 0)$ in the clause.

▶ **Lemma 15.** *Let R be a relation of arity r over a domain D and $r \geq k \geq 2$. If $f^D_{P^u_k} \notin p^2pol(\{R\})$, then $NRD_{\{R\}}(n) = \Omega(n^k)$.*

Proof. Suppose that $f^D_{P^u_k} \notin \mathrm{p}^2\mathrm{pol}(\{R\})$. For simplicity of notation we write $f = f^D_{P^u_k}$ and assume that $\{0,1\} \subseteq D$. (Note that $|D| > 1$ since otherwise we would have $f \in \mathrm{p}^2\mathrm{pol}(R)$.) We claim that $C_k \in \langle\{R\}\rangle_{\mathrm{fg}}$, where $C_k(x_1, \ldots, x_k) \equiv x_1 \vee \ldots \vee x_k$. Let $p = |D_f|$ and $\sigma : D_f \to \{1, \ldots, p\}$ be a bijection such that $\sigma^{-1}(i) = \phi(t_i)$ for all $i \leq k$, where $(t_i, y) \in P^u_k$ is as in Definition 13 and $\phi : \{x, y\} \to D$ is such that $\phi(x) = 1$ and $\phi(y) = 0$. We define

$$\psi(y_1, \ldots, y_p) \equiv \bigwedge_{\substack{(t^*_1,\ldots,t^*_k)\in R \\ \forall i,\, (t^*_1[i],\ldots,t^*_k[i])\in D_f}} R(y_{\sigma((t^*_1[1],\ldots,t^*_k[1]))}, \ldots, y_{\sigma((t^*_1[r],\ldots,t^*_k[r]))})$$

and note that the set of models of this formula are in one-to-one correspondance with the partial polymorphisms of R with domain D_f. Then, the formula

$$\phi(y_1, \ldots, y_k) \equiv \exists y_{k+1}, \ldots, y_p \,:\, \psi(y_1, \ldots, y_p) \bigwedge_{\substack{i,j\leq p: \\ \exists g: D\to D\,:\, \sigma^{-1}(i)=g(\sigma^{-1}(j))}} Q_g(y_j, y_i)$$

is an fgpp-definition of a relation of arity k that contains every tuple of C_k (as projections with domain D_f are pattern partial operations) and cannot contain the tuple $(0, \ldots, 0)$ (otherwise this tuple would extend to the model of ψ that corresponds to f, and by assumption $f \notin \mathrm{p}^2\mathrm{pol}(\{R\})$). Since the unary relation $\{(0), (1)\}$ is fgpp-definable from any language, we have $C_k \in \langle\{R\}\rangle_{\mathrm{fg}}$, as claimed.

Now, by Proposition 2 we need only prove that the language $\{C_k\}$ has non-redundancy $\Omega(n^k)$. A simple argument is to define for any n a CSP instance $I_n = (X, C)$ over $\{C_k\}$ with n variables and such that C contains one constraint $(C_k, (x_1, \ldots, x_k))$ for all distinct $x_1, \ldots, x_k$ in X. This instance has $\Omega(n^k)$ constraints, and none is redundant: for any constraint $c = (C_k, (x_1, \ldots, x_k)) \in C$, the assignment that maps every variable to 1 except $x_1, \ldots, x_k$ satisfies every constraint except c. ◀

▶ **Example 16.** Let $p > 2$, $D = \{1, \ldots, p\}$ and consider the relation $R = \{(x, y, z) \in D^3 \mid \max(x, y) > z\}$. Observe that the set of tuples $S = (\{1, 2\}^2 \times \{2, 3\})\backslash\{(2, 2, 2)\}$ is a subset of R, while the missing tuple $(2, 2, 2)$ does not belong to R. It follows that there exist some ordering $t_1, \ldots, t_8$ of S such that $f^D_{P^u_3}(t_1, \ldots, t_8)$ is defined and is equal to $(2, 2, 2)$, and hence $f^D_{P^u_3} \notin \mathrm{p}^2\mathrm{pol}(\{R\})$. By Lemma 15, $\{R\}$ has non-redundancy $\Omega(n^3)$, which is tight since R has arity 3.

In general, the largest value k for which Lemma 15 applies on a relation R gives a simple lower bound on its non-redundancy. This bound is unlikely to be tight in general, although we do not know any counter-examples.

5 A classification for languages with maximum non-redundancy

In this section, we will combine the lower bound of Lemma 15 with an upper bound derived from Proposition 12 and a well-known theorem in extremal hypergraph theory to prove our main result: a characterisation of constraint languages of arity r whose non-redundancy has the fastest possible asymptotic growth $\Theta(n^r)$. Our approach suggests a simple connection between the non-redundancy of constraint languages and hypergraph Turán numbers.

▶ **Definition 17** ([19]). *Let $\mathcal{H}$ be a family of r-uniform hypergraphs. The nth Turán number of $\mathcal{H}$, denoted by $ex(n, \mathcal{H})$, is the maximum number of edges in an r-uniform hypergraph with n vertices that does not contain any hypergraph in $\mathcal{H}$ as a subgraph.*

The following theorem of Erdős is a fundamental result on this topic.

▶ **Theorem 18** ([13]). *If K_2^r be the complete r-uniform r-partite hypergraph with vertex classes of size two and $K_2^r \in \mathcal{H}$, then $ex(n, \mathcal{H}) = O(n^{r-\epsilon})$, where $\epsilon = 2^{1-r}$.*

We will link relations and hypergraphs in a slightly unusual way. If R is a relation over X, then we define $\mathcal{H}^M(R)$ as the r-partite r-uniform hypergraph over vertex set $X_1, \dots, X_r$, where each $X_i = \{x^i \mid x \in X\}$ is a copy of X, and edge set $\{\{x_1^1, \dots, x_r^r\} \mid (x_1, \dots, x_r) \in R\}$. Note that $\mathcal{H}^M(R)$ has cardinality exactly $|R|$, and its vertex set is of size $r \cdot |X| = O(|X|)$.

▶ **Lemma 19.** *Let R be a relation of arity $r \geq 2$ over a domain D with partial polymorphism $f^D_{P_r^u}$. If $I = (R_X, R)$ is an instance of CSP($\{R\}$) and $\mathcal{H}^M(R_X)$ contains K_2^r as a subgraph, then I contains a redundant constraint.*

Proof. Suppose that K_2^r occurs in $\mathcal{H}^M(R_X)$ as a subgraph. Let $t_1, \dots, t_{2^r}$ be 2^r tuples of R_X whose images in $\mathcal{H}^M(R_X)$ are the edges of a subgraph H isomorphic to K_2^r. Because both $\mathcal{H}^M(R_X)$ and K_2^r are r-uniform r-partite hypergraphs and K_2^r contains for each vertex class $\{x, y\}$ two edges e_1, e_2 with $e_1 = e_2 \backslash \{x\} \cup \{y\}$, the vertex classes of H are subsets of the vertex classes of $\mathcal{H}^M(R_X)$. This implies that for each $j \leq r$, there exist two elements x_j, y_j such that $t_i[j] \in \{x_j, y_j\}$ for all $i \leq 2^r$. Furthermore, all tuples t_i are distinct, so $\{t_i \mid i \leq 2^r\} = \Pi_{j \leq r}\{x_j, y_j\}$ and some tuple, say t_1, is exactly $(y_1, \dots, y_r)$. After reordering lexicographically the other tuples $t_2, \dots, t_2^r$ with respect to $y_j > x_j$, we obtain that $f^X_{P_r^u}(t_2, \dots, t_{2^r}) = t_1$, so by Lemma 8 the constraint (R, t_1) is redundant in I and the claim follows. ◀

▶ **Corollary 20.** *Let R be a relation of arity $r \geq 2$ over a domain D. If $f^D_{P_r^u} \in p^2pol(\{R\})$, then $NRD_{\{R\}}(n) = O(n^{r-\epsilon})$, where $\epsilon = 2^{1-r} > 0$.*

Proof. Let $I = (R_X, R)$ be an instance of CSP($\{R\}$) with exactly $\mathrm{NRD}_{\{R\}}(n)$ constraint, all of which are non-redundant. By Lemma 19, $\mathcal{H}^M(R_X)$ does not contain K_2^r as a subgraph. Since $\mathcal{H}^M(R_X)$ has $O(n)$ vertices, by Theorem 18 it has $O(n^{r-\epsilon})$ edges. By construction we have $|R_X| = |\mathcal{H}^M(R_X)|$, so $|R_X| = O(n^{r-\epsilon})$ and finally $\mathrm{NRD}_{\{R\}}(n) = O(n^{r-\epsilon})$. ◀

Combining Corollary 20 with Lemma 15, we can fully characterise constraint languages with worst-case non-redundancy $\Theta(n^r)$.

▶ **Theorem 21.** *Let Γ be a constraint language with domain D and maximum arity $r \geq 2$. If $f^D_{P_r^u} \notin p^2pol(\Gamma)$ then $NRD_\Gamma(n) = \Theta(n^r)$, and otherwise $NRD_\Gamma(n) = O(n^{r-\epsilon})$, where $\epsilon = 2^{1-r} > 0$.*

Proof. Recall from Section 2 that the non-redundancy of a constraint language is asymptotically determined by the non-redundancy of its individual relations, i.e. $\mathrm{NRD}_\Gamma(n) = \Theta(\max_{R \in \Gamma} \mathrm{NRD}_{\{R\}}(n))$. If $f^D_{P_r^u} \notin \mathrm{p}^2\mathrm{pol}(\Gamma)$ then there exists $R \in \Gamma$ such that $f^D_{P_r^u} \notin \mathrm{p}^2\mathrm{pol}(\{R\})$, and by Lemma 15 we have $\mathrm{NRD}_\Gamma(n) = \Theta(n^r)$. If instead $f^D_{P_r^u} \in \mathrm{p}^2\mathrm{pol}(\Gamma)$, then by Corollary 20 we obtain $\mathrm{NRD}_\Gamma(n) = O(n^{r-\epsilon})$. ◀

It is unlikely that the literature on Turán numbers can be used to derive tight upper bounds. Most results on this topic focus on forbidding a single fixed subhypergraph, while in our case the list of forbidden structures in irredundant instances is typically infinite and

equipped with an algebraic structure; this discrepancy makes any bound obtained this way quite loose. For instance, on the elementary case $r = 2$, Corollary 20 only produces an upper bound of $O(n^{3/2})$ for binary rectangular relations while more direct arguments (Example 11) easily establish the tight bound $\Theta(n)$. Similarly, on Boolean languages the same result holds for $\epsilon = 1$, but proving such a bound using Lemma 8 (rather than polynomials, as in [9]) would necessitate a much deeper analysis of the pattern partial polymorphisms of constraint languages preserved by $f^D_{P^u_r}$.

Finally, we remark that the proof of Corollary 20 implies a simple polynomial-time sparsification algorithm for all languages Γ of arity r with $\mathrm{NRD}_\Gamma(n) = o(n^r)$.

▶ **Theorem 22.** *Let Γ be a constraint language with domain D and maximum arity $r \geq 2$. If $f^D_{P^u_r} \in p^2pol(\Gamma)$, then there exists a polynomial time algorithm that takes an instance of CSP(Γ) as input and outputs an equisatisfiable instance of CSP(Γ) with $O(n^{r-\epsilon})$ constraints, where $\epsilon = 2^{1-r} > 0$.*

Proof. Let $I = (X, C)$ be an instance of CSP(Γ). For each relation $R \in \Gamma$, the algorithm constructs the relation $R_X = \{(x_1, \ldots, x_r) \mid (R, (x_1, \ldots, x_r)) \in C\}$ and enumerates all sequences $t_1, \ldots, t_{2^r}$ of tuples of R_X. For each sequence, it tests whether $t_1 = f^X_{P^u_r}(t_2, \ldots, t_{2^r})$ and discards the constraint (R, t_1) from I when the test succeeds. By Lemma 8, this process only removes redundant constraints. The algorithm then outputs the residual instance.

After this algorithm has terminated, for each relation R the corresponding relation R_X contains at most $O(n^{r-\epsilon})$ tuples because the r-uniform r-partite hypergraph $\mathcal{H}^M(R_X)$ has cardinality $|R_X|$ and does not contain K^r_2 as a subhypergraph. There are $O(1)$ distinct relations in Γ, so the total number of remaining constraints is $O(n^{r-\epsilon})$. ◀

6 Conclusion

We have presented an algebraic framework based on fgpp-definitions and pattern partial polymorphisms dedicated to the study of non-redundancy of constraint languages, extending earlier work on Boolean languages [9, 21]. Based on this framework, we have established a loose connection with extremal hypergraph theory and deduced a characterisation of constraint languages of arity r with non-redundancy $\Theta(n^r)$. The progress we have made in this paper is modest, and much is still unknown on this topic. We believe that the following challenges are the natural next steps towards a better understanding of non-redundancy.

Find a characterisation of constraint languages with non-redundancy $O(n)$. In this paper we have characterised constraint languages whose non-redundancy is the highest possible with respect to their arity, so it would be interesting to do the same for languages whose non-redundancy is the *lowest* possible. It is conceivable that this class coincides with that of languages with a finite Mal'tsev embedding [21] since no counter-example is known. However, proving that it is the case will likely require a better understanding of the pattern partial polymorphisms of these languages and lower bounds more sophisticated than those based on Boolean clauses.

Determine whether all r-ary constraint languages with non-redundancy $o(n^r)$ have non-redundancy $O(n^{r-1})$. This is known to be true for the Boolean domain by the results of Chen et al. [9], but for larger domains we are only able to prove the existence of a considerably smaller gap which vanishes as r grows. Both our approach and that of Chen et al. have intrinsic limitations when dealing simultaneously with large domains and large arities, so it would be interesting to see how they could be combined.

Determine the non-redundancy of all ternary constraint languages. A classification is known for binary languages (see [5], although a more direct proof follows from Example 11 and Lemma 15) and ternary Boolean languages [9], but not on ternary languages with arbitrary domains.

Clarify the relationship between non-redundancy, sparsification, and learnability. In particular, it would be interesting to determine whether non-redundancy $O(n^q)$ implies sparsification algorithms with output size $O(n^q)$ and whether non-redundancy is asymptotically equivalent to chain length, a closely related measure that characterises the efficiency of a class of learning algorithms for constraint acquisition [5].

References

1 Albert Atserias, Andrei A. Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Compututer Science*, 410(18):1666–1683, 2009. doi:10.1016/j.tcs.2008.12.049.

2 Libor Barto, Zarathustra Brady, Andrei Bulatov, Marcin Kozik, and Dmitriy Zhuk. Minimal taylor algebras as a common framework for the three algebraic approaches to the CSP. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'21)*, pages 1–13. IEEE, 2021. doi:10.1109/LICS52264.2021.9470557.

3 Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *Journal of the ACM*, 61(1):3:1–3:19, 2014. doi:10.1145/2556646.

4 Libor Barto, Andrei Krokhin, and Ross Willard. Polymorphisms, and how to use them. In *Dagstuhl Follow-Ups*, volume 7. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

5 Christian Bessiere, Clément Carbonnel, and George Katsirelos. Chain length and csps learnable with few queries. In *Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI'20)*, pages 1420–1427, 2020. doi:10.1609/aaai.v34i02.5499.

6 Christian Bessiere, Remi Coletta, Emmanuel Hebrard, George Katsirelos, Nadjib Lazaar, Nina Narodytska, Claude-Guy Quimper, and Toby Walsh. Constraint acquisition via partial queries. In Francesca Rossi, editor, *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, pages 475–481, 2013.

7 Christian Bessiere, Frédéric Koriche, Nadjib Lazaar, and Barry O'Sullivan. Constraint acquisition. *Artificial Intelligence*, 244:315–342, 2017. doi:10.1016/j.artint.2015.08.001.

8 Andrei A. Bulatov. A dichotomy theorem for nonuniform csps. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science (FOCS'17)*, pages 319–330. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.37.

9 Hubie Chen, Bart M. P. Jansen, and Astrid Pieterse. Best-case and worst-case sparsifiability of Boolean csps. *Algorithmica*, 82(8):2200–2242, 2020. doi:10.1007/s00453-019-00660-y.

10 Hubie Chen and Matthew Valeriote. Learnability of solutions to conjunctive queries. *Journal of Machine Learing Research*, 20:67:1–67:28, 2019. URL: http://jmlr.org/papers/v20/17-734.html.

11 Miguel Couceiro, Lucien Haddad, and Victor Lagerkvist. A survey on the fine-grained complexity of constraint satisfaction problems based on partial polymorphisms. *J. Multiple Valued Log. Soft Comput.*, 38(1-2):115–136, 2022.

12 László Egri, Pavol Hell, Benoît Larose, and Arash Rafiey. Space complexity of list H-colouring: a dichotomy. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'14)*, pages 349–365. SIAM, 2014. doi:10.1137/1.9781611973402.26.

13 P Erdös. On extremal problems of graphs and generalized graphs. *Israel Journal of Mathematics*, 2(3):183–190, 1964.

14 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.

15 David Geiger. Closed systems of functions and predicates. *Pacific journal of mathematics*, 27(1):95–100, 1968.

16 Pawel M. Idziak, Petar Markovic, Ralph McKenzie, Matthew Valeriote, and Ross Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM Journal on Computing*, 39(7):3023–3037, 2010. doi:10.1137/090775646.

17 Peter Jeavons, David Cohen, and Marc Gyssens. Closure properties of constraints. *J. ACM*, 44(4):527–548, July 1997. doi:10.1145/263867.263489.

18 Peter Jonsson, Victor Lagerkvist, Gustav Nordh, and Bruno Zanuttini. Strong partial clones and the time complexity of SAT problems. *J. Comput. Syst. Sci.*, 84:52–78, 2017. doi:10.1016/j.jcss.2016.07.008.

19 Peter Keevash. Hypergraph turán problems. *Surveys in combinatorics*, 392:83–140, 2011.

20 Victor Lagerkvist and Magnus Wahlström. Kernelization of constraint satisfaction problems: A study through universal algebra. In *Proceedings of the 23rd Conference on Principles and Practice of Constraint Programming (CP'17)*, pages 157–171, 2017. doi:10.1007/978-3-319-66158-2_11.

21 Victor Lagerkvist and Magnus Wahlström. Which np-hard SAT and CSP problems admit exponentially improved algorithms? *CoRR*, abs/1801.09488, 2018. arXiv:1801.09488.

22 Benoît Larose, Cynthia Loten, and Claude Tardif. A characterisation of first-order constraint satisfaction problems. *Logical Methods in Computer Science*, 3(4), 2007. doi:10.2168/LMCS-3(4:6)2007.

23 Emil L. Post. The two-valued iterative systems of mathematical logic. *Annals of Mathematics studies*, 1941. doi:10.2307/2268608.

24 Boris A Romov. The algebras of partial functions and their invariants. *Cybernetics*, 17(2):157–167, 1981.

25 Thomas J. Schaefer. The complexity of satisfiability problems. In *STOC '78: Proceedings of the tenth annual ACM Symposium on Theory of Computing (STOC'78)*, pages 216–226. Association for Computing Machinery, 1978. doi:10.1145/800133.804350.

26 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *Journal of the ACM*, 67(5):30:1–30:78, 2020. doi:10.1145/3402029.