Constraints (2023) 28:13–44
https://doi.org/10.1007/s10601-022-09340-1

# The algebraic structure of the densification and the sparsification tasks for CSPs

**Rustem Takhanov**[1]

Accepted: 12 November 2022 / Published online: 8 December 2022


**Abstract**

The tractability of certain CSPs for dense or sparse instances is known from the 90s. Recently, the densification and the sparsification of CSPs were formulated as computational tasks and the systematical study of their computational complexity was initiated. We approach this problem by introducing the densification operator, i.e. the closure operator that, given an instance of a CSP, outputs all constraints that are satisfied by all of its solutions. According to the Galois theory of closure operators, any such operator is related to a certain implicational system (or, a functional dependency) $\Sigma$. We are specifically interested in those classes of fixed-template CSPs, parameterized by constraint languages $\Gamma$, for which there is an implicational system $\Sigma$ whose size is a polynomial in the number of variables $n$. We show that in the Boolean case, such implicational systems exist if and only if $\Gamma$ is of bounded width. For such languages, $\Sigma$ can be computed in log-space or in a logarithmic time with a polynomial number of processors. Given an implicational system $\Sigma$, the densification task is equivalent to the computation of the closure of input constraints. The sparsification task is equivalent to the computation of the minimal key.

**Keywords** Horn formula minimization · Sparsification of CSP · Densification of CSP · Polynomial densification operator · Implicational system · Bounded width · Datalog

## 1 Introduction

In the constraint satisfaction problem (CSP) [1–3] we are given a set of variables with prescribed domains and a set of constraints. The task's goal is to assign each variable a value such that all the constraints are satisfied. Given an instance of CSP, besides the classical formulation, one can formulate many other tasks, such as maximum/minimum CSPs (Max/Min-CSPs) [4], valued CSP (VCSPs) [5, 6], counting CSPs [7, 8], promise CSPs [9, 10], quantified CSPs [11–13], and others. Thus, the computational task of finding a single solution is not the only aspect that is of interest from the perspective of applications of CSPs.

✉ Rustem Takhanov
rustem.takhanov@nu.edu.kz

[1] Nazarbayev University, School of Sciences and Humanities, Mathematics Department, 53 Kabanbay Batyr Ave, Nur-Sultan, 010000, Kazakhstan

Sometimes in applications we have a CSP instance that defines a set of solutions, and we need to preprocess the instance by making it denser (i.e. adding new constraints) or, vice versa, sparser (removing as many constraints as we can) without changing the set of solutions. Let us give an example of such an application. The paper by Jia Deng et al. [14] is dedicated to the Conditional Random Field (CRF) based on the so-called HEX graphs. The algorithm for the inference in CRFs presented there is based on the standard junction tree algorithm [15], but with one additional trick — before constructing the junction tree of the factor graph, the factor tree is sparsified. This step aims to make the factor graph as close to the tree structure as possible. After that step, cliques of the junction tree are expected to have fewer nodes. The sparsification of the HEX graph done in this approach is equivalent to the sparsification of a CSP instance, i.e. the deletion of as many constraints as possible while maintaining the set of solutions. The term "sparsification" is also used in a related line of work in which the goal is, given a CSP instance, to reduce the number of constraints without changing the satisfiability of an instance [16, 17].

As was suggested in [14], the densification of a CSP instance could also help make inference algorithms more efficient. If the factor tree is densified, then for every clique $c$ of the factor graph, the number of consistent assignments to variables of the clique $c$ is smaller. Thus, reducing the state space for each clique makes the inference faster. The sparsification-densification approach substantially accelerates the computation of the marginals as the number of nodes grows.

It is well-known that the complexity of the sparsification problem, as well as the worst-case sparsifiability, depends on the constraint language, i.e. the types of constraints allowed in CSP. The computational complexity was completely classified for constraint languages consisting of the so-called irreducible relations [18].

For a constraint language that consists of Boolean relations of the form $A_1 \wedge A_2 \wedge ... \wedge A_n \rightarrow B$ (so-called pure Horn clauses), the sparsification task is equivalent to the problem of finding a minimum size cover of a given functional dependency (FD) table. The last problem was studied in database theory long ago and is considered a classical topic. It was shown that this problem is NP-hard both in the general case and in the case a cover is restricted to be a subset of the given FD table. Surprisingly, if we re-define the size of a cover as the number of distinct left-hand side expressions $A_1 \wedge A_2 \wedge ... \wedge A_n$, then the problem is polynomially solvable [19].

An important generalization of the previous constraint language is a set of Horn clauses (i.e. $B$ can be equal to **False**). The sparsification problem for this language is known by the name *Horn minimization*, i.e. it is a problem of finding the minimum size Horn formula that is equivalent to an input Horn formula. Horn minimization is NP-hard if the number of clauses is to be minimized [20, 21], or if the number of literals is to be minimized [22]. Moreover, in the former case Horn minimization cannot be $2^{\log^{1-\epsilon}(n)}$-approximated if NP $\not\subseteq$ DTIME$(n^{\mathrm{polylog}(n)})$ [23].

An example of a tractable sparsification problem is 2-SAT formula minimization [24] which corresponds to the constraint language of binary relations over the Boolean domain.

The key idea of this paper's approach is to consider both densification and sparsification as two operations defined on the same set, i.e. the set of possible constraints. We observe that the densification is a closure operator on a finite set, and therefore, according to Galois theory [25], it can be defined using a functional dependency table, or so-called implicational system $\Sigma$ (over a set of possible constraints and, maybe, some additional literals). It turns out that $\Sigma$ can have a size bounded by some polynomial of the number of variables only if the constraint language is of bounded width (for tractable languages not of bounded width, the size of $\Sigma$ could still be substantially smaller than for NP-hard languages). For the Boolean domain, all languages of bounded width have a polynomial-size implicational system $\Sigma$.

Given an implicational system $\Sigma$, the sparsification problem can be reformulated as a problem of finding the minimal key in $\Sigma$, i.e. such a set of constraints whose densification is the same as the densification of initial constraints. This task was actively studied in database theory, and we exploit the standard algorithm for the solution of the minimal key problem, found by Luchessi and Osborn [26]. If $|\Sigma| = \mathcal{O}(\text{poly}(n))$ and literals of $\Sigma$ are all from the set of possible constraints, this leads us to a $\mathcal{O}(\text{poly}(n) \cdot N^2)$-sparsification algorithm where $N$ is the number of non-redundant sparsifications of an input instance. This algorithm can be applied to the Horn minimization problem, and, to our knowledge, this is the first algorithm that is polynomial in $N$. Of course, in the worst-case $N$ is large.

Besides the mentioned works, densification/sparsification tasks were also studied for soft CSPs, and this unrelated research direction includes graph densification [27–29], binary CSP sparsification [30–34] and spectral sparsification of graphs and hypergraphs [35, 36]. In the 90's it was found that dense CSP instances (i.e. when the number of $k$-ary constraints is $\Theta(n^k)$) admit efficient algorithms for the Max-$k$-CSP and the maximum assignment problems [37–39]. Though we deal with crisp CSPs and not any CSP instance can be densified to $\Theta(n^k)$ constraints, the idea to densify a CSP instance seems natural in this context. Note that the densification of a CSP that we study in our paper is substantially different from the notion of the densification of a graph. Initially, Hardt et al. [27] define the densification of the graph $G = (V, E)$ as a new graph $H = (V, E')$, $E' \supseteq E$ such that the cardinalities of cuts in $G$ and $H$ are proportional. In [28, 29] and in the Ph.D. Thesis [40] the densification was naturally applied in a clustering problem to neighborhood graphs to make more intra-class links and smaller overhead of inter-class links. It was shown that this makes the Laplacian of a graph better conditioned for a subsequent application of spectral methods. A theoretical analysis of the densification/sparsification tasks for soft CSPs requires mathematical tools substantially different from those that we develop in the paper.

## 2 Preliminaries

We assume that P $\neq$ NP. The set $\{1, ..., k\}$ is denoted by $[k]$. Given a relation $\varrho \subseteq R^s$ and a tuple $\mathbf{a} \in R^{s'}$, by $\|\varrho\|$ and $|\mathbf{a}|$ we denote $s$ and $s'$, respectively. A relational structure is a tuple $\mathbf{R} = (R, r_1, ..., r_k)$ where $R$ is finite set, called the domain of $\mathbf{R}$, and $r_i \subseteq R^{\|r_i\|}$, $i \in [k]$. If $p_0 \in [\|\varrho\|]$, then $\text{pr}_{\{p_0\}}(\varrho) = \{a_{p_0} | (a_1, ..., a_k) \in \varrho\}$, if $p_0 < p_1 \leq \|\varrho\|$, then $\text{pr}_{\{p_0, p_1\}}(\varrho) = \{(a_{p_0}, a_{p_1}) | (a_1, ..., a_k) \in \varrho\}$ etc.

### 2.1 The homomorphism formulation of CSP

Let us define first the notion of a homomorphism between relational structures.

**Definition 1** Let $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{R}' = (V', r_1', ..., r_s')$ be relational structures with a common signature (that is arities of $r_i$ and $r_i'$ are the same for every $i \in [s]$). A mapping $h: V \to V'$ is called a *homomorphism* from $\mathbf{R}$ to $\mathbf{R}'$ if for every $i \in [s]$ and for any $(x_1, ..., x_{\|r_i\|}) \in r_i$ we have that $\big((h(x_1), ..., h(x_{\|r_i'\|})\big) \in r_i'$. The set of all homomorphisms from $\mathbf{R}$ to $\mathbf{R}'$ is denoted by $\text{Hom}(\mathbf{R}, \mathbf{R}')$.

The classical CSP can be formulated as a homomorphism problem.

**Definition 2** The **CSP** is a search task with:

- **An instance:** two relational structures with a common signature, $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$.
- **An output:** a homomorphism $h : \mathbf{R} \to \mathbf{\Gamma}$ if it exists, or answer None, if it does not exist.

A finite relational structure $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$ over a fixed finite domain $D$ is sometimes called a template. For such $\mathbf{\Gamma}$ we will denote by $\Gamma$ (without boldface) the set of relations $\{\varrho_1, ..., \varrho_s\}$. The set $\Gamma$ is called the constraint language.

**Definition 3** The **fixed template CSP** for a given template $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$, denoted CSP($\mathbf{\Gamma}$), is defined as follows: given a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$ of the same signature as $\mathbf{\Gamma}$, solve the CSP for an instance ($\mathbf{R}$, $\mathbf{\Gamma}$). If CSP($\mathbf{\Gamma}$) is solvable in a polynomial time, then $\mathbf{\Gamma}$ is called tractable. Otherwise, $\mathbf{\Gamma}$ is called NP-hard [2, 3].

## 2.2 Algebraic approach to CSPs

In the paper we will need standard definitions of primitive positive formulas and polymorphisms.

**Definition 4** Let $\tau = \{\pi_1, ..., \pi_s\}$ be a set of symbols for predicates, with the arity $n_i$ assigned to $\pi_i$. A first-order formula $\Phi(x_1, ..., x_k) = \exists x_{k+1}...x_n \Xi(x_1, ..., x_n)$ where $\Xi(x_1, ..., x_n) = \bigwedge_{t=1}^{N} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ..., x_{o_{tn_{j_t}}})$, $j_t \in [s]$, $o_{tq} \in [n]$ is called the primitive positive formula over the vocabulary $\tau$. For a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$, $\|r_i\| = n_i$, $i \in [s]$, $\Phi^{\mathbf{R}}$ denotes a $k$-ary predicate

$$\{(a_1, ..., a_k) | a_i \in V, i \in [k], \exists a_{k+1}, \cdots, a_n \in V : (a_{o_{t1}}, a_{o_{t2}}, ..., a_{o_{tn_{j_t}}}) \in r_{j_t}, t \in [N]\},$$

i.e. the result of interpreting the formula $\Phi$ on the model $\mathbf{R}$, where $\pi_i$ is interpreted as $r_i$.

For $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$ and $\tau = \{\pi_1, ..., \pi_s\}$, let us denote the set $\{\Psi^{\mathbf{\Gamma}} | \Psi$ is primitive positive formula over $\tau\}$ by $\langle\Gamma\rangle$.

**Definition 5** Let $\varrho \subseteq D^m$ and $f : D^n \to D$. We say that the predicate $\varrho$ is preserved by $f$ (or, $f$ is a polymorphism of $\varrho$) if, for every $\left(x_1^i, ..., x_m^i\right) \in \varrho$, $1 \leq i \leq n$, we have that $\left(f\left(x_1^1, ..., x_1^n\right), ..., f\left(x_m^1, ..., x_m^n\right)\right) \in \varrho$.

For a set of predicates $\Gamma \subseteq \{\varrho | \varrho \subseteq D^m\}$, let $\mathrm{Pol}(\Gamma)$ denote the set of operations $f : D^n \to D$ such that $f$ is a polymorphism of all predicates in $\Gamma$. For a set of operations $F \subseteq \{f | f : D^n \to D\}$, let $\mathrm{Inv}(F)$ denote the set of predicates $\varrho \subseteq D^m$ preserved under the operations of $F$. The next result is well-known [41, 42].

**Theorem 1** (**Geiger, Bodnarchuk, Kaluznin, Kotov, Romov**) *For a set of predicates $\Gamma$ over a finite set $D$, $\langle\Gamma\rangle = Inv(Pol(\Gamma))$.*

It is well-known that the computational complexity of fixed-template CSPs, counting CSPs, VCSPs etc. is determined by the closure $\langle\Gamma\rangle$, and therefore, by the corresponding functional clone $\mathrm{Pol}(\Gamma)$.

## 3 The fixed template densification and sparsification problems

Let us give a general definition of maximality and list some properties of maximal instances.

**Definition 6** An instance $(\mathbf{R}, \mathbf{\Gamma})$ of CSP, where $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$, is said to be maximal if for any $\mathbf{R}' = (V, r_1', ..., r_s')$ such that $r_i' \supseteq r_i$, $i \in [s]$ we have $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$, unless $\mathbf{R}' = \mathbf{R}$.

The following characterization of maximal instances is evident from Definition 6 (also, see Theorem 1 in [43]).

**Theorem 2** *An instance* $(\mathbf{R} = (V, r_1, ..., r_s), \mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s))$ *is maximal if and only if for any* $i \in [s]$ *and any* $(v_1, ..., v_{\|r_i\|}) \notin r_i$ *there exists* $h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$ *such that* $(h(v_1), ..., h(v_{\|r_i\|})) \notin \varrho_i$.

One can prove the following simple existence theorem (Statement 1 in [43]).

**Theorem 3** *For any instance* $(\mathbf{R} = (V, r_1, ..., r_s), \mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s))$ *of CSP, there exists a unique maximal instance* $(\mathbf{R}' = (V, r_1', ..., r_s'), \mathbf{\Gamma})$ *such that* $r_i' \supseteq r_i, i \in [s]$ *and* $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$. *Moreover, if* $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \emptyset$, *then*

$$r_i' = \bigcap_{h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})} h^{-1}(\varrho_i), i \in [s]$$

Thus, the maximal instance $(\mathbf{R}', \mathbf{\Gamma})$ from Theorem 3 can be called the densification of $(\mathbf{R}, \mathbf{\Gamma})$. Let us now formulate constructing $(\mathbf{R}', \mathbf{\Gamma})$ from $(\mathbf{R}, \mathbf{\Gamma})$ as an algorithmic problem.

**Definition 7** The **densification problem**, denoted Dense, is a search task with:

- **An instance:** two relational structures with a common signature, $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$.
- **An output:** a maximal instance $(\mathbf{R}' = (V, r_1', ..., r_s'), \mathbf{\Gamma})$ such that $r_i' \supseteq r_i, i \in [s]$ and $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$.

Also, let $D$ be a finite set and $\mathbf{\Gamma}$ a relational structure with a domain $D$. Then, the **fixed template densification problem** for the template $\mathbf{\Gamma}$, denoted Dense($\mathbf{\Gamma}$), is defined as follows: given a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$ of the same signature as $\mathbf{\Gamma}$, solve the densification problem for an instance $(\mathbf{R}, \mathbf{\Gamma})$.

Let $\Gamma = \{\varrho_1, \cdots, \varrho_s\}$. The language $\Gamma$ is called constant-preserving if there is $a \in D$ such that $(a, \cdots, a) \in \varrho_i$ for any $i \in [s]$. For a pair $(\mathbf{R}, \mathbf{\Gamma})$, where $\Gamma$ is not a constant-preserving language, the corresponding densification is non-trivial, i.e. $\mathbf{R}' \neq (V, V^{\|r_1\|}, \cdots, V^{\|r_s\|})$, if and only if $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \emptyset$. Therefore, the densification problem for such templates $\mathbf{\Gamma}$ is at least as hard as the decision form of CSP. In other words, if the decision form of CSP($\mathbf{\Gamma}$) is NP-hard (which is known to be polynomially equivalent to the search form), then all the more Dense($\mathbf{\Gamma}$) is NP-hard.

For a Boolean constraint language $\Gamma$, we say that $\Gamma$ is Schaefer in one of the following cases: 1) $x \vee y \in \mathrm{Pol}(\Gamma)$, 2) $x \wedge y \in \mathrm{Pol}(\Gamma)$, 3) $x \oplus y \oplus z \in \mathrm{Pol}(\Gamma)$, 4) $\mathrm{mjy}(x, y, z) = (x \wedge y) \vee (x \wedge y) \vee (x \wedge z) \in \mathrm{Pol}(\Gamma)$. The complexity of Dense($\mathbf{\Gamma}$) in the Boolean case can be simply described by the following theorem whose proof uses earlier results of [44] and [45]. For completeness, a detailed proof can be found in Section 11.

**Theorem 4** *For* $D = \{0, 1\}$, $\text{Dense}(\mathbf{\Gamma})$ *is polynomially solvable if and only if* $\Gamma$ *is Schaefer.*

Let us introduce the sparsification problem.

**Definition 8** An instance $(\mathbf{R}, \mathbf{\Gamma})$ of CSP, where $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$, is said to be minimal if for any $\mathbf{T} = (V, t_1, ..., t_s)$ such that $t_i \subseteq r_i, i \in [s]$ we have $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \text{Hom}(\mathbf{T}, \mathbf{\Gamma})$, unless $\mathbf{T} = \mathbf{R}$.

Let us define:

$$\text{Min}(\mathbf{R}, \mathbf{\Gamma}) = \left\{\mathbf{R}' = (V, r_1', ..., r_s') \mid \text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \text{Hom}(\mathbf{R}', \mathbf{\Gamma}), (\mathbf{R}', \mathbf{\Gamma}) \text{ is minimal}\right\} \tag{1}$$

**Definition 9** The **sparsification problem**, denoted Sparse, is a search task with:

– **An instance:** two relational structures with a common signature, $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$.
– **An output:** List of all elements of $\text{Min}(\mathbf{R}, \mathbf{\Gamma})$.

Also, let $D$ be a finite set and $\mathbf{\Gamma}$ a relational structure with a domain $D$. Then, the **fixed template sparsification problem** for the template $\mathbf{\Gamma}$, denoted $\text{Sparse}(\mathbf{\Gamma})$, is defined as follows: given a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$ of the same signature as $\mathbf{\Gamma}$, solve the sparsification problem for an instance $(\mathbf{R}, \mathbf{\Gamma})$.

*Remark 1* In many aplications $\text{Min}(\mathbf{R}, \mathbf{\Gamma})$ is of moderate size, though potentially it can depend on $|V|$ exponentially. Also, $\mathbf{R}' = (V, r_1', ..., r_s') \in \text{Min}(\mathbf{R}, \mathbf{\Gamma})$ is not necessarily a substructure of $\mathbf{R}$, i.e. it is possible that $r_i' \not\subseteq r_i$. Enforcing $r_i' \subseteq r_i, i \in [s]$ in the definition of $\text{Min}(\mathbf{R}, \mathbf{\Gamma})$ is discussed in Remark 2.

## 4 Densification as the closure operator

Let us introduce a set of all possible constraints over $\Gamma$ on a set of variables $V$:

$$\mathcal{C}_V^{\mathbf{\Gamma}} = \{\langle (v_1, ..., v_{\|\varrho_i\|}), \varrho_i \rangle | i \in [s], v_1, ..., v_{\|\varrho_i\|} \in V\}$$

Any instance of $\text{CSP}(\mathbf{\Gamma})$, a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$, induces the following subset of $\mathcal{C}_V^{\mathbf{\Gamma}}$:

$$\mathcal{C}_{\mathbf{R}} = \{\langle (v_1, ..., v_{\|\varrho_i\|}), \varrho_i \rangle | i \in [s], (v_1, ..., v_{\|\varrho_i\|}) \in r_i\}$$

Using that notation, the densification can be understood as an operator $\text{Dense} : 2^{\mathcal{C}_V^{\mathbf{\Gamma}}} \to 2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$ such that:

$$\text{Dense}(\mathcal{C}_{\mathbf{R}}) = \left\{\langle (v_1, ..., v_{\|\varrho_i\|}), \varrho_i \rangle | i \in [s], (v_1, ..., v_{\|\varrho_i\|}) \in \bigcap_{h \in \text{Hom}(\mathbf{R}, \mathbf{\Gamma})} h^{-1}(\varrho_i)\right\}$$

Thus, in the densification process we start from a set of constraints $\mathcal{C}_{\mathbf{R}}$ and simply add possible constraints to $\text{Dense}(\mathcal{C}_{\mathbf{R}})$ while the set of solutions is preserved. Let us also define $\text{Dense}(\mathcal{C}_{\mathbf{R}}) = \mathcal{C}_V^{\mathbf{\Gamma}}$ if $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$. The densification operator satisfies the following conditions:

– $\text{Dense}(\mathcal{C}_{\mathbf{R}}) \supseteq \mathcal{C}_{\mathbf{R}}$ (extensive)

– $\text{Dense}(\text{Dense}(\mathcal{C}_{\mathbf{R}})) = \text{Dense}(\mathcal{C}_{\mathbf{R}})$ (idempotent)
– $\mathcal{C}_{\mathbf{R}'} \subseteq \mathcal{C}_{\mathbf{R}} \Rightarrow \text{Dense}(\mathcal{C}_{\mathbf{R}'}) \subseteq \text{Dense}(\mathcal{C}_{\mathbf{R}})$ (isotone)

Operators that satisfy these three conditions play the central role in universal algebra and are called the closure operators. There exists a duality between closure operators $o : 2^S \to 2^S$ on a finite set $S$ and the so-called implicational systems (or functional dependencies) on $S$. Let us briefly describe this duality (details can be found in [25]).

**Definition 10** Let $S$ be a finite set. An implicational system $\Sigma$ on $S$ is a binary relation $\Sigma \subseteq 2^S \times 2^S$. If $(A, B) \in \Sigma$, we write $A \to B$. A full implicational system on $S$ is an implicational system satisfying the three following properties:

– $A \to B, B \to C$ imply $A \to C$
– $A \subseteq B$ imply $B \to A$
– $A \to B$ and $C \to D$ imply $A \cup C \to B \cup D$.

Any implicational system $\Sigma \subseteq 2^S \times 2^S$ has a minimal superset $\Sigma' \supseteq \Sigma$ that itself is a full implicational system on $S$. This system is called the closure of $\Sigma$ and is denoted by $\Sigma^{\triangleright}$. Let us call $\Sigma_1$ a cover of $\Sigma_2$ if $\Sigma_1^{\triangleright} = \Sigma_2^{\triangleright}$.

**Theorem 5 (p. 264** [25]) *Any implicational system $\Sigma \subseteq 2^S \times 2^S$ defines the closure operator $o : 2^S \to 2^S$ by $o(A) = \{x \in S | A \to \{x\} \in \Sigma^{\triangleright}\}$. Any closure operator $o : 2^S \to 2^S$ on a finite set $S$ defines the full implicational system by $\{A \to B | B \subseteq o(A)\}$.*

From Theorem 5 we obtain that the densification operator $\text{Dense} : 2^{\mathcal{C}_V^{\mathbf{\Gamma}}} \to 2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$ also corresponds to some full implicational system $\Sigma_V^{\mathbf{\Gamma}} \subseteq 2^{\mathcal{C}_V^{\mathbf{\Gamma}}} \times 2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$. Note that the system $\Sigma_V^{\mathbf{\Gamma}}$ depends only on the set $V$ and the template $\mathbf{\Gamma}$, but does not depend on relations $r_i, i \in [s]$ of the relational structure $\mathbf{R}$.

Note the densification problem described in Definition 7 can be understood as a computation of the monotone function $\text{Dense} : \cup_{n=1}^{\infty} 2^{\mathcal{C}_{[n]}^{\mathbf{\Gamma}}} \to \cup_{n=1}^{\infty} 2^{\mathcal{C}_{[n]}^{\mathbf{\Gamma}}}$. With a little abuse of terminology, let us define the class mP/poly as a class of monotone functions $M : \bigcup_{i=0}^{\infty} \{0, 1\}^{n_i} \to \bigcup_{i=0}^{\infty} \{0, 1\}^{m_i}$ for which $M(\{0, 1\}^{n_i}) \subseteq \{0, 1\}^{m_i}$ and $M|_{\{0,1\}^{n_i}}$ can be computed by a circuit of size $\text{poly}(n_i)$ that uses only $\vee$ and $\wedge$ in gates. Thus, $\text{Dense}(\mathbf{\Gamma}) \in$ mP/poly denotes the fact that the corresponding densification operator is in mP/poly.

## 5 The polynomial densification operator

Let denote $\Sigma_n^{\mathbf{\Gamma}} = \Sigma_{[n]}^{\mathbf{\Gamma}}$. The most general languages with a kind of polynomial densification operator can be described as follows.

**Definition 11** The template $\mathbf{\Gamma}$ is said to have a weak polynomial densification operator, if for any $n \in \mathbb{N}$ there exists an implicational system $\Sigma$ on $S \supseteq \mathcal{C}_n^{\mathbf{\Gamma}}$ of size $|\Sigma| = \mathcal{O}(\text{poly}(n))$ that acts on $\mathcal{C}_n^{\mathbf{\Gamma}}$ as the densification operator, i.e. $\Sigma_n^{\mathbf{\Gamma}} = \{(A \to B) \in \Sigma^{\triangleright} | A, B \subseteq \mathcal{C}_n^{\mathbf{\Gamma}}\}$.

Using database theory language [46], the last definition describes such languages $\Gamma$ for which there exists an implicational system of polynomial size whose projection on $\mathcal{C}_n^{\mathbf{\Gamma}}$ coincides with $\Sigma_n^{\mathbf{\Gamma}}$. Note that in Definition 11, a weak densification operator acts on a wider set than $\mathcal{C}_n^{\mathbf{\Gamma}}$: an addition of new literals to $\mathcal{C}_n^{\mathbf{\Gamma}}$, sometimes, allows to substantially simplify a set of implications [47]. Though we are not aware of an example of a language $\Gamma$ for which any cover $\Sigma \subseteq \Sigma_n^{\mathbf{\Gamma}}$ of $\Sigma_n^{\mathbf{\Gamma}}$ is exponential in size, but still $\Gamma$ has a weak polynomial densification operator.

## 6 Main results

Recall that bounded width languages are languages for which $\neg$CSP($\mathbf{\Gamma}$) can be recognized by a Datalog program [1]. Concerning the weak polynomial densification, we obtain the following result.

**Theorem 6** *For the general domain $D$, if $\mathbf{\Gamma}$ has a weak polynomial densification operator, then $\mathbf{\Gamma}$ is of bounded width. For the Boolean case, $D = \{0, 1\}$, $\mathbf{\Gamma}$ has a weak polynomial densification operator if and only if Pol($\Gamma$) contains either $\vee$, or $\wedge$, or mjy($x, y, z$).*

The first part of the latter theorem is proved in Section 7 and the Boolean case is considered in Section 13. We also prove the following statement for the sparsification problem (Section 9).

**Theorem 7** *If $\Sigma \subseteq \Sigma_V^{\mathbf{\Gamma}}$ is a cover of $\Sigma_V^{\mathbf{\Gamma}}$ that can be computed in time poly($|V|$), then given an instance $\mathbf{R} = (V, r_1, ..., r_s)$ of Sparse($\mathbf{\Gamma}$), all elements of* $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$ *can be listed in time $\mathcal{O}(poly(|V|) \cdot |\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})|^2)$.*

## 7 Weak polynomial densification implies bounded width

A set of languages with a weak polynomial densification operator turns out to be a subset of a set of languages of bounded width. Below we demonstrate this fact in two steps. First, we prove that from a weak polynomial densification operator one can construct a polynomial-size monotone circuit that computes $\neg$CSP($\mathbf{\Gamma}$). Further, we exploit a well-known result from a theory of fixed-template CSPs connecting the bounded width with such circuits.

**Theorem 8** *If $\mathbf{\Gamma}$ has a weak polynomial densification operator, then the decision version of $\neg$CSP($\mathbf{\Gamma}$) can be computed by a polynomial-size monotone circuit.*

*Proof* If $\mathbf{\Gamma}$ is constant-preserving, then $\neg$CSP($\mathbf{\Gamma}$) is trivial, i.e. we can assume that $\mathbf{\Gamma}$ is not constant-preserving. Let $\Sigma_n$ be an implicational system on $S_n \supseteq \mathcal{C}_n^{\mathbf{\Gamma}}$ such that $\Sigma_n^{\triangleright} \cap (2^{\mathcal{C}_n^{\mathbf{\Gamma}}})^2 = \Sigma_n^{\mathbf{\Gamma}}$ and $|\Sigma_n| = \mathcal{O}(\mathrm{poly}(n))$. We can assume that $S_n = \mathcal{O}(\mathrm{poly}(n))$ and every rule in $\Sigma_n$ has a form $A \to x$, $x \in S_n$. Let $\mathbf{R}$ be an instance of CSP($\mathbf{\Gamma}$) and $x \in \mathcal{C}_n^{\mathbf{\Gamma}}$. The rule $\mathcal{C}_{\mathbf{R}} \to x$ is in $\Sigma_n^{\triangleright}$ if and only if $x$ is derivable from $\mathcal{C}_{\mathbf{R}}$ using implications from $\Sigma_n$. Formally, the latter means that there is a directed acyclic graph $T = (U, E)$ with a labeling function $l : U \to S_n$ such that: (a) there is only one element with no outcoming edges, the root $r \in U$, and it is labeled by $x$, i.e. $l(r) = x$, (b) every node with no incoming edges is labeled by an element of $\mathcal{C}_{\mathbf{R}}$, (c) if a node $v \in U$ has incoming edges $(c_1, v), ..., (c_{d(v)}, v)$, then $(\{l(c_1), ..., l(c_{d(v)})\} \to l(v)) \in \Sigma_n$. Moreover, the depth of $T$ is bounded by $|S_n|$, because $x$ can be derived from $\mathcal{C}_{\mathbf{R}}$ in no more than $|S_n|$ steps if no attribute is derived twice.

Consider a monotone circuit $M$ whose set of variables, denoted by $W$, consists of $|S_n|$ layers $U_1, ..., U_{|S_n|}$ such that $i$-th layer is a set of variables $v_{i,a}$, $a \in S_n$. For any rule $b \in S_n$ and every $i \in [|S_n| - 1]$ there is a monotone logic gate

$$v_{i+1,b} = v_{i,b} \vee \bigvee_{(\{a_1,...,a_l\} \to b) \in \Sigma_n} (v_{i,a_1} \wedge v_{i,a_2} \wedge ... \wedge v_{i,a_l})$$

that computes the value of $v_{i+1,b}$ from inputs of the previous layer.

Any instance $\mathbf{R}$ can be encoded as a Boolean vector $\mathbf{v_R} \in \{0, 1\}^{S_n}$ such that $\mathbf{v_R}(x) = 1$ if and only if $x \in \mathcal{C}_\mathbf{R}$. If we set input variables of $M$ to $\mathbf{v_R}$, i.e. $v_{1,a} := \mathbf{v_R}(a)$, $a \in S_n$, then output variables of $M$, i.e. $v_{|S_n|,a}$, $a \in S_n$, will satisfy: for any $x \in \mathcal{C}_n^{\mathbf{\Gamma}}$, $v_{|S_n|,x} = 1$ if and only if $(\mathcal{C}_\mathbf{R} \to x) \in \Sigma_n^{\triangleright}$. Let us briefly outline the proof of the last statement.

Indeed, let $v_{|S_n|,x} = 1$, $x \in \mathcal{C}_\mathbf{R}$. For any variable $v_{i,b} \in W$ such that $v_{i,b} = 1$ let us define $\text{early}(v_{i,b}) = v_{i',b}$ where $v_{i',b} = 1$ and $v_{i'-1,b} = 0$ and $\text{source}(v_{i,b}) = \{v_{i'-1,a_1}, v_{i'-1,a_2}, ..., v_{i'-1,a_l}\}$ if $(\{a_1, ..., a_l\} \to b) \in \Sigma_n$ and $v_{i'-1,a_1} = 1, v_{i'-1,a_2} = 1, ..., v_{i'-1,a_l} = 1$. Then, a rooted directed acyclic graph $T_x = (U, E)$ with a labeling $l : U \to S_n$ can be constructed by defining $U = \{\text{early}(v_{i,b}) | v_{i,b} \in W, v_{i,b} = 1\}$ and $l(\text{early}(v_{i,b})) = b$. Edges of $T_x$ are defined in the following way: if $v_{i',b} = \text{early}(v_{i,b})$ and $v_{i',b}$ was assigned to 1 by the gate $v_{i',b} = v_{i'-1,b} \vee (v_{i'-1,a_1} \wedge v_{i'-1,a_2} \wedge ... \wedge v_{i'-1,a_l}) \vee \cdots$ where $\text{source}(v_{i,b}) = \{v_{i'-1,a_1}, v_{i'-1,a_2}, ..., v_{i'-1,a_l}\}$, then we connect nodes $\text{early}(v_{i'-1,a_2}), ..., \text{early}(v_{i'-1,a_l})$ to $v_{i',b}$ by incoming edges. It is easy to see that $T_x$ will satisfy properties (a), (b), (c) listed above. The opposite is also true, if there is a directed acyclic graph with a root $x$ that satisfies the properties (a), (b), (c), then $v_{|S_n|,x} = 1$.

Thus, the expression $o = \bigwedge_{x \in \mathcal{C}_n^{\mathbf{\Gamma}}} v_{|S_n|,x}$ equals 1 if and only if $(\mathcal{C}_\mathbf{R} \to \mathcal{C}_n^{\mathbf{\Gamma}}) \in \Sigma_n^{\mathbf{\Gamma}}$. Since $\mathbf{\Gamma}$ is not constant-preserving, the last means $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$. Thus, $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$ was computed by the polynomial-size monotone circuit $M$ (with an additional gate). □

The core of $\Gamma = \{\varrho_1, ..., \varrho_s\}$ is defined as $\text{core}(\Gamma) = \{\varrho_1 \cap g(D)^{n_1}, ..., \varrho_s \cap g(D)^{n_s}\}$, the constraint language over $g(D)$, where $g \in \text{Hom}(\mathbf{\Gamma}, \mathbf{\Gamma})$ is such that $g(x) = g(g(x))$ and

$$|g(D)| = \min_{h \in \text{Hom}(\mathbf{\Gamma}, \mathbf{\Gamma})} |h(D)|.$$

**Corollary 1** *If $\mathbf{\Gamma}$ has a weak polynomial densification operator, then core$(\Gamma)$ is of bounded width.*

*Proof* If $\mathbf{\Gamma}$ has a weak polynomial densification operator, then by Theorem 8 $\neg\text{CSP}(\Gamma)$ can be solved by a polynomial-size monotone circuit. Therefore, $\neg\text{CSP}(\Gamma')$ where $\Gamma' = \text{core}(\Gamma) \cup \{\{(a)\} | a \in g(D)\}$ can also be solved by a polynomial-size monotone circuit. We can use the standard reduction of $\neg\text{CSP}(\Gamma')$ to $\neg\text{CSP}(\text{core}(\Gamma) \cup \{\rho\})$ where $\rho \in \langle \text{core}(\Gamma) \rangle$ is defined as $\{\langle \pi(a) \rangle_{a \in g(D)} | \pi : g(D) \to g(D), \pi \in \text{pol}(\text{core}(\Gamma))\}$.

The algebra $\mathbb{A}_{\Gamma'} = (g(D), \text{pol}(\Gamma'))$ generates the variety of algebras $var(\mathbb{A}_{\Gamma'})$ (in the sense of Birkhoff's HSP theorem). Proposition 5.1. from [48] states that if $\neg\text{CSP}(\Gamma')$ can be computed by a polynomial-size monotone circuit, then $var(\mathbb{A}_{\Gamma'})$ omits both the unary and the affine type. According to a well-known result [49, 50] this is equivalent to stating that $\Gamma'$ is of bounded width. □

## 8 Algebraic approach to the classification of languages with a polynomial densification operator

Constraint languages for which the densification problem Dense($\mathbf{\Gamma}$) is tractable can be classified using tools of universal algebra. An analogous approach can be applied to classify languages with a weak polynomial densification operator.

**Definition 12** Let $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$ and $\tau = \{\pi_1, ..., \pi_s\}$. A $k$-ary relation $\rho \in \langle\Gamma\rangle$ is called strongly reducible to $\Gamma$ if there exists a quantifier-free primitive positive formula $\Xi(x_1, \cdots, x_n)$[1] (over $\tau$) and $\delta \subseteq D^n$ for some $n \geq k$ such that $\mathrm{pr}_{1:k}\,\Xi^{\mathbf{\Gamma}} = \rho$, $\mathrm{pr}_{1:k}\delta = D^k \backslash \rho$ and $\Xi^{\mathbf{\Gamma}} \cup \delta \in \langle\Gamma\rangle$. A $k$-ary relation $\rho \in \langle\Gamma\rangle$ is called A-reducible to $\Gamma$ if $\rho = \rho_1 \cap \cdots \cap \rho_l$, where $\rho_i \in \langle\Gamma\rangle$ is strongly reducible to $\Gamma$ for $i \in [l]$.

**Definition 13** A constraint language $\Gamma$ is called an A-language if any $\rho \in \langle\Gamma\rangle$ is A-reducible to $\Gamma$.

Examples of A-languages are stated in the following theorems, whose proofs can be found in Section 14.

**Theorem 9** *Let* $\mathbf{\Gamma} = (D = \{0, 1\}, \varrho_1, \varrho_2, \varrho_3)$ *where* $\varrho_1 = \big\{(x, y)|x \vee y\big\}$, $\varrho_2 = \big\{(x, y)|\neg x \vee y\big\}$ *and* $\varrho_3 = \big\{(x, y)|\neg x \vee \neg y\big\}$. *Then,* $\Gamma$ *is an A-language.*

**Theorem 10** *Let* $\mathbf{\Gamma} = (D = \{0, 1\}, \{(0)\}, \{(1)\}, \varrho_{x\wedge y\to z})$ *where* $\varrho_{x\wedge y\to z} = \{(a_1, a_2, a_3) \in D^3|a_1a_2 \leq a_3\}$. *Then,* $\Gamma$ *is an A-language.*

Reducibility of a relation to a language is an interesting notion because of its property stated in the following theorem.

**Theorem 11** *Let* $\Gamma, \Gamma'$ *be constraint languages such that* $\Gamma' \subseteq \langle\Gamma\rangle$, *and every relation in* $\Gamma'$ *is A-reducible to* $\Gamma$. *Then:*

*(a)* Dense($\mathbf{\Gamma}'$) *is polynomial-time Turing reducible to* Dense($\mathbf{\Gamma}$)*;*
*(b)* *if* $\mathbf{\Gamma}$ *has a weak polynomial densification operator, then* $\mathbf{\Gamma}'$ *also has a weak polynomial densification operator;*
*(c)* *if* Dense($\mathbf{\Gamma}$) $\in$ *mP/poly, then* Dense($\mathbf{\Gamma}'$) $\in$ *mP/poly.*

*Proof* Since $\Gamma' \subseteq \langle\Gamma\rangle$, then there is $L = \{\Phi_i|\ i \in [c]\}$ where $\Phi_i$ is a primitive positive formula over the vocabulary $\tau = \{\pi_1, ..., \pi_s\}$, such that $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$, $\mathbf{\Gamma}' = (D, \Phi_1^{\mathbf{\Gamma}}, ..., \Phi_c^{\mathbf{\Gamma}})$.

Let $\mathbf{R}' = (V, r_1', ..., r_c')$ be an instance of Dense($\mathbf{\Gamma}'$). Our goal is to compute a maximal instance $(\mathbf{R}'' = (V, r_1'', ..., r_c''), \mathbf{\Gamma}')$ such that $r_i'' \supseteq r_i', i \in [c]$ and $\mathrm{Hom}(\mathbf{R}'', \mathbf{\Gamma}') = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma}')$, or in other words, to compute Dense($\mathcal{C}_{\mathbf{R}'}$).

First, let us introduce some notations. Let $\Psi$ be any primitive positive formula over $\tau$, i.e. $\Psi = \exists x_{k+1}...x_l \bigwedge_{t\in[N]} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ...)$ where $j_t \in [s]$ and $o_{tx} \in [l]$ and $\mathbf{a} = (a_1, ..., a_k)$ be a tuple of objects. Let us introduce a set of new distinct objects $\mathrm{NEW}(\mathbf{a}, \Psi) = \{a_{k+1}, ..., a_l\}$. Note that the sets $\mathrm{NEW}(\mathbf{a}, \Psi)$ are disjoint for different $(\mathbf{a}, \Psi)$

[1] A quantifier-free pp-formula is a pp-formula without existential quantification.

(also, $\mathrm{NEW}(\mathbf{a}, \Psi) \cap V = \emptyset$). For a tuple $\mathbf{a} = (a_1, ..., a_k)$, the constraint that an assignment to $(a_1, ..., a_k)$ is in $\Psi^{\mathbf{\Gamma}}$ can be expressed by a collection of constraints $\mathfrak{C}(\mathbf{a}, \Psi) = \{\langle (a_{o_{t1}}, a_{o_{t2}}, ...), \varrho_{j_t} \rangle \mid t \in [N]\}$. In other words, we require that an image of $(a_{o_{t1}}, a_{o_{t2}}, ...)$ is in $\varrho_{j_t}$ for any $t \in [N]$. Note that $\mathfrak{C}(\mathbf{a}, \Psi)$ is a set of constraints over a set of variables $\{a_1, ..., a_k\} \cup \mathrm{NEW}(\Psi, \mathbf{a})$ where only relations from $\Gamma$ are allowed.

Let us start with a proof of statement (a). We will describe a reduction to Dense($\mathbf{\Gamma}$) that consists of two steps: first we add new variables and construct an instance of CSP($\mathbf{\Gamma}$) in the same way as it is done in the standard reduction of CSP($\mathbf{\Gamma}'$) to CSP($\mathbf{\Gamma}$); afterwards, we add new variables and constraints and form an instance of Dense($\mathbf{\Gamma}$).

First, for any $i \in [c], \mathbf{a} \in r_i'$, we add objects $\mathrm{NEW}(\mathbf{a}, \Phi_i)$ to the set of variables $V$ and define an extended set $M^0 = V \cup \bigcup_{i \in [c], \mathbf{a} \in r_i'} \mathrm{NEW}(\mathbf{a}, \Phi_i)$. Afterwards, we define a relational structure $(\mathbf{R}^0 = (M^0, r_1^0, ..., r_s^0), \mathbf{\Gamma})$ where $\mathcal{C}_{\mathbf{R}^0} = \bigcup_{i \in [c], \mathbf{a} \in r_i'} \mathfrak{C}(\mathbf{a}, \Phi_i)$. By construction, $\mathrm{pr}_V \mathrm{Hom}(\mathbf{R}^0, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma}')$. Note that this reduction is standard in the algebraic approach to fixed-template CSPs. This is the first step of the construction.

Let us now consider a relation $\Phi_i^{\mathbf{\Gamma}}$ and assume that its arity is $k$. According to the assumption, $\Phi_i^{\mathbf{\Gamma}}$ is A-reducible to $\Gamma$. Therefore, $\Phi_i^{\mathbf{\Gamma}} = \varrho_{i1} \cap \cdots \cap \varrho_{il}$, where $\varrho_{ij}$ is strongly reducible to $\Gamma$ for $j \in [l]$. Thus, there exists a quantifier-free primitive positive formula over $\tau$, $\Xi_j$, involving $r_j$ variables, and $\delta_j \subseteq D^{r_j}$, such that $\varrho_{ij} = \mathrm{pr}_{1:k} \Xi_j^{\mathbf{\Gamma}}$ and $\mathrm{pr}_{1:k} \delta_j = D^k \setminus \varrho_{ij}$ and $\delta_j \cup \Xi_j^{\mathbf{\Gamma}} \in \langle \Gamma \rangle$. Since $\gamma_j = \delta_j \cup \Xi_j^{\mathbf{\Gamma}}$ is pp-definable over $\Gamma$, there exists a primitive positive formula over $\tau$, $\exists x_{r_j+1} \cdots x_{p_j} \Theta_j(x_1, \cdots, x_{p_j})$ where $\Theta_j$ is quantifier-free, such that $(\exists x_{r_j+1} \cdots x_{p_j} \Theta_j(x_1, \cdots, x_{p_j}))^{\mathbf{\Gamma}} = \delta_j \cup \Xi_j^{\mathbf{\Gamma}}$. Let us introduce a set of constraints:

$$\mathfrak{C}(V, \Phi_i) = \bigcup_{(a_1,...,a_k) \in V^k} \bigcup_{j \in [l]} \mathfrak{C}\big((a_1, ..., a_k), \exists x_{k+1}, \cdots, x_{p_j} \Theta_j(x_1, \cdots, x_{p_j})\big).$$

over a set of variables

$$M_i = V \cup \bigcup_{(a_1,...,a_k) \in V^k} \bigcup_{j \in [l]} \mathrm{NEW}\big((a_1, ..., a_k), \exists x_{k+1}, \cdots, x_{p_j} \Theta_j(x_1, \cdots, x_{p_j})).$$

Due to $\mathrm{pr}_{1:k} \delta_j = D^k \setminus \varrho_{ij}$, we have $\mathrm{pr}_{1:k}(\delta_j \cup \Xi_j^{\mathbf{\Gamma}}) = D^k$. Therefore,

$$(\exists x_{k+1} \cdots x_{p_j}\ \Theta_j(x_1, \cdots, x_{p_j}))^{\mathbf{\Gamma}} = \mathrm{pr}_{1:k}(\delta_j \cup \Xi_j^{\mathbf{\Gamma}}) = D^k.$$

Thus, the set of constraints $\mathfrak{C}(V, \Phi_i)$ does not add any restrictions on assignments of $V$ (though it adds restrictions on additional variables).

Let $\mathbf{R} = (M, r_1, ..., r_s)$ be such that $M = V \cup \bigcup_{i \in [c], \mathbf{a} \in r_i'} \mathrm{NEW}(\mathbf{a}, \Phi_i) \bigcup_{i \in [c]} M_i$ and $\mathcal{C}_{\mathbf{R}} = \bigcup_{i \in [c], \mathbf{a} \in r_i'} \mathfrak{C}(\mathbf{a}, \Phi_i) \bigcup_{i \in [c]} \mathfrak{C}(V, \Phi_i)$. By construction, $\mathrm{pr}_V \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma}')$. Let us treat $\mathbf{R}$ as an instance of Dense($\mathbf{\Gamma}$).

The computation of Dense($\mathcal{C}_{\mathbf{R}'}$) can be made by checking whether $\langle (v_1, \cdots, v_k), \Phi_i^{\mathbf{\Gamma}} \rangle \in$ Dense($\mathcal{C}_{\mathbf{R}'}$) for any $v_1, \cdots, v_k \in V$ and a $k$-ary $\Phi_i^{\mathbf{\Gamma}} \in \Gamma$. From the following lemma it follows that such a checking can be reduced to a checking of certain conditions of the form $\langle (u_1, u_2, ...), \varrho_j \rangle \in$ Dense($\mathcal{C}_{\mathbf{R}}$), i.e. to the computation of Dense($\mathcal{C}_{\mathbf{R}}$).

**Lemma 1** *For a $k$-ary $\Phi_i^{\mathbf{\Gamma}}$ and $v_1, \cdots, v_k \in V$ there is a subset $S_i(v_1, \cdots, v_k) \subseteq \mathcal{C}_M^{\mathbf{\Gamma}}$ (that can be computed in time poly($|V|$)) such that the condition $\langle (v_1, \cdots, v_k), \Phi_i^{\mathbf{\Gamma}} \rangle \in$* Dense($\mathcal{C}_{\mathbf{R}'}$) *($\subseteq \mathcal{C}_V^{\mathbf{\Gamma}'}$) is equivalent to a list of conditions $\langle (u_1, u_2, ...), \varrho_j \rangle \in$* Dense($\mathcal{C}_{\mathbf{R}}$) *($\subseteq \mathcal{C}_M^{\mathbf{\Gamma}}$) for $\langle (u_1, u_2, ...), \varrho_j \rangle \in S_i(v_1, \cdots, v_k)$.*

*Proof* Note that $\langle(v_1,\cdots,v_k),\Phi_i^{\boldsymbol{\Gamma}}\rangle\in \mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})\subseteq \mathcal{C}_V^{\boldsymbol{\Gamma}'}$ for $v_1,\cdots,v_k\in V$ if and only if $\mathrm{pr}_{v_1,\cdots,v_k}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\Phi_i^{\boldsymbol{\Gamma}}$. Let us assume that we have $\mathrm{pr}_{v_1,\cdots,v_k}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\Phi_i^{\boldsymbol{\Gamma}}$. The definition of $\mathbf{R}$ implies that we have a set of constraints

$$\mathfrak{C}((v_1,...,v_k),\exists x_{k+1},\cdots,x_{p_j}\Theta_j(x_1,\cdots,x_{p_j}))$$

imposed on $v_1,\cdots,v_k$ and

$$\mathrm{NEW}\big((v_1,...,v_k),\exists x_{k+1},\cdots,x_{p_j}\Theta_j(x_1,\cdots,x_{p_j})\big)=\{v_{k+1},\cdots,v_{p_j}\}$$

(how $\Phi_i$ and $\Theta_j$, $j\in[l]$ are related is described above). Since $\Phi_i^{\boldsymbol{\Gamma}}=\varrho_{i1}\cap\cdots\cap\varrho_{il}$, we conclude $\mathrm{pr}_{v_1,\cdots,v_k}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\varrho_{ij}$, $j\in[l]$. Therefore, $\mathrm{pr}_{v_1,\cdots,v_{p_j}}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\{\mathbf{x}\in\Theta_j^{\boldsymbol{\Gamma}}\mid \mathbf{x}_{1:k}\in\varrho_{ij}\}$, that is $\mathrm{pr}_{v_1,\cdots,v_{r_j}}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\{\mathbf{x}_{1:r_j}\mid\mathbf{x}\in\Theta_j^{\boldsymbol{\Gamma}},\mathbf{x}_{1:k}\in\varrho_{ij}\}=\Xi_j^{\boldsymbol{\Gamma}}$. Since $\Xi_j$ is a quantifier-free primitive positive formula over $\tau$, then the fact $\mathrm{pr}_{v_1,\cdots,v_{r_j}}\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})\subseteq\Xi_j^{\boldsymbol{\Gamma}}$ can be expressed as $(h(v_1),\cdots,h(v_{r_j}))\in\Xi_j^{\boldsymbol{\Gamma}}$ for any $h\in\mathrm{Hom}(\mathbf{R},\boldsymbol{\Gamma})$. In other words, if $\Xi_j=\exists x_{k+1}...x_l\bigwedge_{t\in[N]}\pi_{w_t}(x_{o_{t1}},x_{o_{t2}},...)$, then $\langle(v_{o_{t1}},v_{o_{t2}},...),\varrho_{w_t}\rangle\in\mathrm{Dense}(\mathcal{C}_{\mathbf{R}})\subseteq\mathcal{C}_V^{\boldsymbol{\Gamma}}$ for any $t\in[N]$. Let us set

$$S_i(v_1,\cdots,v_k)=\{\langle(v_{o_{t1}},v_{o_{t2}},...),\varrho_{w_t}\rangle\mid\Xi_j=\exists x_{k+1}...x_l\bigwedge_{t\in[N]}\pi_{w_t}(x_{o_{t1}},x_{o_{t2}},...),\ j\in[l]\}$$

In fact, we proved

$$\langle(v_1,\cdots,v_k),\Phi_i^{\boldsymbol{\Gamma}}\rangle\in\mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})\Rightarrow S_i(v_1,\cdots,v_k)\subseteq\mathrm{Dense}(\mathcal{C}_{\mathbf{R}}).$$

It can be easily checked that the last chain of arguments can be reversed, and

$$S_i(v_1,\cdots,v_k)\subseteq\mathrm{Dense}(\mathcal{C}_{\mathbf{R}})\Rightarrow\langle(v_1,\cdots,v_k),\Phi_i^{\boldsymbol{\Gamma}}\rangle\in\mathrm{Dense}(\mathcal{C}_{\mathbf{R}'}).$$

□

Thus, statement (a) is proved.

Statement (b) directly follows from the previous reduction. Suppose $\boldsymbol{\Gamma}$ has a weak polynomial densification operator, i.e. there is a finite $S_n\supseteq\mathcal{C}_n^{\boldsymbol{\Gamma}}$ and an implicational system $\Delta_n\subseteq 2^{S_n}\times 2^{S_n}$ of size $|\Delta_n|=\mathcal{O}(\mathrm{poly}(n))$ that acts on $\mathcal{C}_n^{\boldsymbol{\Gamma}}$ as the densification operator, i.e. $\Sigma_n^{\boldsymbol{\Gamma}}=\{(A\to B)\in\Delta_n^{\triangleright}|A,B\subseteq\mathcal{C}_n^{\boldsymbol{\Gamma}}\}$.

If $V=[n]$, then $X=V\cup\bigcup_{i\in[c],\mathbf{a}=(a_1,a_2,\cdots,),a_i\in V}\mathrm{NEW}(\mathbf{a},\Phi_i)\bigcup_{i\in[c]}M_i$ ($M_i$ are defined above) is a superset of $V$ whose size is bounded by a polynomial of $n$. Therefore, w.l.o.g. we can assume $X=[m]$ where $m=|X|=\mathcal{O}(\mathrm{poly}(n))$. Let $\Delta_m$ be an implicational system on $S_m\supseteq\mathcal{C}_m^{\boldsymbol{\Gamma}}$ such that $|\Delta_m|=\mathcal{O}(\mathrm{poly}(m))$ and $o_{\Delta_m}(S)=\{x\in\mathcal{C}_m^{\boldsymbol{\Gamma}}|(S\to x)\in\Delta_m^{\triangleright}\}$ acts as the densification operator on subsets of $\mathcal{C}_m^{\boldsymbol{\Gamma}}$. Since $\Delta_m\subseteq 2^{S_m}\times 2^{S_m}$, we can interpret $\Delta_m$ as an implicational system on $S'_m=S_m\cup\mathcal{C}_n^{\boldsymbol{\Gamma}'}$, i.e. we include $\mathcal{C}_n^{\boldsymbol{\Gamma}'}$ into a set of literals of $\Delta_m$. Let us now add to $\Delta_m$ new implications by the following rule: for $\Phi_i=\exists x_{k+1}...x_l\bigwedge_{t\in[N]}\pi_{j_t}(x_{o_{t1}},x_{o_{t2}},...)$, $\mathbf{a}\in[n]^k$ and the corresponding new $l-k$ variables $\mathrm{NEW}(\mathbf{a},\Phi_i)=\{a_{k+1},...,a_l\}$ we add $R(\mathbf{a},\Phi_i):\langle\mathbf{a},\Phi_i^{\boldsymbol{\Gamma}}\rangle\to\{\langle(a_{o_{t1}},a_{o_{t2}},...),\varrho_{j_t}\rangle|t\in[N]\}$. Let us denote

$$\mathfrak{R}_1=\bigcup_{i\in[c],\mathbf{a}=(a_1,a_2,...):a_i\in V}\{R(\mathbf{a},\Phi_i)\}.$$

The second kind of implications that we need to add to $\Delta_m$ is

$$\mathfrak{R}_2=\bigcup_{i\in[c]}\{\emptyset\to\mathfrak{C}(V,\Phi_i)\}.$$

The last set of implications, $\mathfrak{R}_3$, is defined by

$$\mathfrak{R}_3 = \{(S_i(v_1, \cdots, v_k) \to \langle (v_1, \cdots, v_k), \Phi_i^{\mathbf{\Gamma}} \rangle) \mid \langle (v_1, \cdots, v_k), \Phi_i^{\mathbf{\Gamma}} \rangle \in \mathcal{C}_n^{\mathbf{\Gamma}'}\},$$

where $S_i(v_1, \cdots, v_k)$ is described in the previous Lemma, i.e. it equals a set of constraints for which $S_i(v_1, \cdots, v_k) \subseteq \text{Dense}(\mathcal{C}_{\mathbf{R}})$ is equivalent to $\langle (v_1, \cdots, v_k), \Phi_i^{\mathbf{\Gamma}} \rangle \in \text{Dense}(\mathcal{C}_{\mathbf{R}'})$. Thus, we defined a set of implications $\Delta_m \cup \mathfrak{R}_1 \cup \mathfrak{R}_2 \cup \mathfrak{R}_3$. Let us denote a new system by $\Sigma_n$. By the construction of $\Sigma_n$, we have $|\Sigma_n| = \mathcal{O}(\text{poly}(n))$.

Given $\mathcal{C}_{\mathbf{R}'}$, using implications from $\mathfrak{R}_1$, one can derive the set of constraints $\mathcal{C}_{\mathbf{R}^0}$ ($\mathbf{R}^0$ is defined above), and using implications from $\mathfrak{R}_2$ one completes the set of derivable literals to $\mathcal{C}_{\mathbf{R}}$. Then, using initial rules of $\Delta_m$, one can derive from $\mathcal{C}_{\mathbf{R}}$ its closure $\text{Dense}(\mathcal{C}_{\mathbf{R}})$. Finally, using implications from $\mathfrak{R}_3$ one can derive all constraints from $\text{Dense}(\mathcal{C}_{\mathbf{R}'})$. It is not hard to prove that $x \in \mathcal{C}_n^{\mathbf{\Gamma}'}$ is derivable from $\mathcal{C}_{\mathbf{R}'}$ if and only if $x \in \text{Dense}(\mathcal{C}_{\mathbf{R}'})$.

Thus, $\mathbf{\Gamma}'$ also has a weak polynomial densification operator. Note that implications $\mathfrak{R}_2 \cup \mathfrak{R}_3$ are all from $\Sigma_m^{\mathbf{\Gamma} \cup \mathbf{\Gamma}'}$, but an implication $R(\mathbf{a}, \Phi_i) \in \mathfrak{R}_1$ is not, in general, from $\Sigma_m^{\mathbf{\Gamma} \cup \mathbf{\Gamma}'}$.

Statement (c) directly follows from the fact that the function $Q : 2^{\mathcal{C}_V^{\Gamma'}} \to 2^{\mathcal{C}_M^{\Gamma}}$ such that $Q(C_{\mathbf{R}'}) = C_{\mathbf{R}}$ is monotone and can be computed by a polynomial-size monotone circuit. □

## 9 DS-basis and algorithms for Dense($\Gamma$) and Sparse($\Gamma$)

The notion of DS-basis is a formalization of templates for which a small cover of $\Sigma_n^{\mathbf{\Gamma}}$ not only exists but can also be computed efficiently.

**Definition 14** A fixed template $\mathbf{\Gamma}$ is called a DS-basis, if there exists an algorithm $\mathcal{A}$ that solves in time $\mathcal{O}(\text{poly}(n))$ the task with:

- An instance: a natural number $n \in \mathbb{N}$;
- An output: an implicational system $\Sigma \subseteq \Sigma_n^{\mathbf{\Gamma}}$ such that $\Sigma^{\triangleright} = \Sigma_n^{\mathbf{\Gamma}}$.

**Theorem 12** *For any DS-basis $\mathbf{\Gamma}$ there is an algorithm $\mathcal{A}_1$ that, given an instance $\mathbf{R}$ of* Dense($\mathbf{\Gamma}$)*, solves the densification problem for* $(\mathbf{R}, \mathbf{\Gamma})$ *in time* $\mathcal{O}(\text{poly}(|V|))$.

*Proof* For any implicational system $\Sigma \subseteq 2^S \times 2^S$, and any $A, B \subseteq S$, the membership $A \to B \overset{?}{\in} \Sigma^{\triangleright}$ can be checked in time $\mathcal{O}(|\Sigma|)$ by Beeri and Bernstein's algorithm for functional dependencies [51].

Since $\mathbf{\Gamma}$ is the DS-basis, then there exists an algorithm $\mathcal{A}$ using which one can compute in time $\mathcal{O}(\text{poly}(|V|))$ an implicational system $\Sigma \subseteq \Sigma_V^{\mathbf{\Gamma}}$ such that $\Sigma^{\triangleright} = \Sigma_V^{\mathbf{\Gamma}}$. Afterwards, we check whether $\mathcal{C}_{\mathbf{R}} \to x \overset{?}{\in} \Sigma_V^{\mathbf{\Gamma}}$ using Beeri and Bernstein's algorithm for any $x \in \mathcal{C}_V^{\mathbf{\Gamma}}$ and compute $\text{Dense}(\mathcal{C}_{\mathbf{R}}) = \{x \in \mathcal{C}_V^{\mathbf{\Gamma}} | \mathcal{C}_{\mathbf{R}} \to x \in \Sigma^{\triangleright}\}$ in time $\mathcal{O}(|\mathcal{C}_V^{\mathbf{\Gamma}}| \cdot |\Sigma|) = \mathcal{O}(\text{poly}(|V|))$. Finally we set $r_i' = \{(v_1, ..., v_{\|\varrho_i\|}) | \langle (v_1, ..., v_{\|\varrho_i\|}), \varrho_i \rangle \in \text{Dense}(\mathcal{C}_{\mathbf{R}})\}$ for $i \in [s]$. The instance $(\mathbf{R}' = (V, r_1', ..., r_s'), \mathbf{\Gamma})$ is maximal. □

The following theorem is equivalent to Theorem 7 announced in Section 6.

**Theorem 13** *For any DS-basis $\mathbf{\Gamma}$ there is an algorithm $\mathcal{A}_2$ that, given an instance $\mathbf{R}$ of Sparse*($\mathbf{\Gamma}$)*, solves the sparsification problem for* $(\mathbf{R}, \mathbf{\Gamma})$ *in time* $\mathcal{O}(\textit{poly}(|V|) \cdot |\text{Min}(\mathbf{R}, \mathbf{\Gamma})|^2)$.

*Proof* It is easy to see that a set of all possible instances of Sparse($\mathbf{\Gamma}$), $\{\mathbf{R} = (V, \cdots)\}$, is in one-to-one correspondence with a set $2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$. For any implicational system $F$ on $S$, let us call $A \subseteq S$ a minimal key of $F$ for $B$ if $(A \to B) \in F^{\triangleright}$, but for any proper subset $C \subset A$, $(C \to B) \notin F^{\triangleright}$. Let us prove first that $\mathbf{R}' \in \mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$ is and only if $\mathcal{C}_{\mathbf{R}'}$ is a minimal key of $\Sigma_V^{\mathbf{\Gamma}}$ for Dense($\mathcal{C}_{\mathbf{R}}$).

Indeed, if $\mathbf{R}' \in \mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$, then $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$. Since $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$, then $\mathrm{Dense}(\mathcal{C}_{\mathbf{R}}) = \mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})$ (by the definition of the densification operator). Therefore, from the duality between the closure operator Dense and the implication system $\Sigma_V^{\mathbf{\Gamma}}$ we obtain $(\mathcal{C}_{\mathbf{R}'} \to \mathrm{Dense}(\mathcal{C}_{\mathbf{R}})) \in \Sigma_V^{\mathbf{\Gamma}}$. Since the pair $(\mathbf{R}', \mathbf{\Gamma})$ is minimal, we obtain that $\mathcal{C}_{\mathbf{R}'}$ is a minimal key for Dense($\mathcal{C}_{\mathbf{R}}$).

On the contrary, let $\mathcal{C}_{\mathbf{R}'}$ be a minimal key for Dense($\mathcal{C}_{\mathbf{R}}$). Therefore, $\mathrm{Dense}(\mathcal{C}_{\mathbf{R}}) = \mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})$, from which we obtain $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma})$. Any proper subset $\mathcal{C}_{\mathbf{R}''} \subset \mathcal{C}_{\mathbf{R}'}$ has a closure $\mathrm{Dense}(\mathcal{C}_{\mathbf{R}''}) \subset \mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})$. Thus, we obtain that $\mathrm{Hom}(\mathbf{R}', \mathbf{\Gamma}) \neq \mathrm{Hom}(\mathbf{R}'', \mathbf{\Gamma})$ (otherwise, we have $\mathrm{Dense}(\mathcal{C}_{\mathbf{R}''}) = \mathrm{Dense}(\mathcal{C}_{\mathbf{R}'})$). We conclude that the pair $(\mathbf{R}', \mathbf{\Gamma})$ is minimal.

Since $\mathbf{\Gamma}$ is a DS-basis, we construct in advance an implicational system $\Sigma \subseteq \Sigma_V^{\mathbf{\Gamma}}$ such that $\Sigma^{\triangleright} = \Sigma_V^{\mathbf{\Gamma}}$. We proved that the problem of listing of $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$ is equivalent to a listing of all minimal keys for Dense($\mathcal{C}_{\mathbf{R}}$) in the implicational system $\Sigma$. In database theory, this task is called the optimal cover problem and was studied in the 70s [52]. The algorithm of Luchessi and Osborn lists all minimal keys for Dense($\mathcal{C}_{\mathbf{R}}$) in time $\mathcal{O}(|\Sigma| \cdot |\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})| \cdot |\mathrm{Dense}(\mathcal{C}_{\mathbf{R}})| \cdot (|\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})| + |\mathrm{Dense}(\mathcal{C}_{\mathbf{R}})|))$ (see p. 274 of [26]). It is easy to see that the last expression is bounded by $\mathcal{O}(\mathrm{poly}(|V|) \cdot |\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})|^2)$.

Note that main approaches to listing minimal keys in a functional dependency table refer to the method of Luchessi and Osborn. Nowadays, several alternative methods are designed for this and adjacent tasks [53], including efficient parallelization techniques [54]. □

*Remark 2* Sometimes we are interested not in $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$, but in its subset $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}, S) = \{\mathbf{R}' \in \mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}) \mid \mathcal{C}_{\mathbf{R}'} \subseteq S\}$ where $S \subseteq \mathcal{C}_V^{\mathbf{\Gamma}}$. For example, if $S = \mathcal{C}_{\mathbf{R}}$, then listing $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}, S)$ is equivalent to a listing of all non-redundant sparsifications that are subsets of the set of initial constraints. The latter set could have a substantially smaller cardinality than $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma})$. A natural approach to list $\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}, S)$ is to compute a cover $\Sigma'$ of $\Sigma_V^{\mathbf{\Gamma}} \cap (2^S)^2 = \Sigma^{\triangleright} \cap (2^S)^2$ and then list minimal keys of $\Sigma'$ for $S$ (sometimes called candidate keys) by the method of Luchessi and Osborn in time $\mathcal{O}(|\Sigma'| \cdot |\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}, S)| \cdot |S| \cdot (|\mathrm{Min}(\mathbf{R}, \mathbf{\Gamma}, S)| + |S|))$. For the computation of $\Sigma'$, it is natural to exploit the Reduction by Resolution algorithm (RBR) suggested in [55]. The bottleneck of that strategy is that a small cover of $\Sigma^{\triangleright} \cap (2^{\mathcal{C}_{\mathbf{R}}})^2$ may not exist. In such cases RBR's computation takes a long time that can be potentially exponential.

Next, we will show that DS-bases include such templates for which Dense($\mathbf{\Gamma}$) can be solved by a Datalog program.

## 10 Densification by Datalog program

The idea of using Datalog programs for CSP is classical [1, 56, 57].

**Definition 15** If $\Phi(x_1, ..., x_{n_u})$ is a primitive positive formula over $\tau$, then the first-order formula

$$\Psi = \forall x_1, ..., x_{n_u} \big(\Phi(x_1, ..., x_{n_u}) \to \pi_u(x_1, ..., x_{n_u})\big)$$

is called a Horn formula[2] over $\tau$. If a primitive positive definition of $\Phi$ involves $n$ variables, then $\Psi$ is said to be of width $(n_u, n)$ (or, simply, of width $n$). Any Horn formula of width $(n_u, n)$ is equivalent to the universal formula

$$\forall x_1, ..., x_n \big(\bigwedge_{t=1}^{N} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ..., x_{o_{tn_{j_t}}}) \to \pi_u(x_1, ..., x_{n_u})\big),$$

so we will refer to both of them as Horn formulas. For a relational structure $\mathbf{R} = (V, r_1, ..., r_s)$, $\|r_i\| = n_i$, $\mathbf{R} \models \Psi$ denotes $\Phi^{\mathbf{R}} \subseteq r_u$.

For the densification task, the use of Datalog is motivated by the following theorem.

**Theorem 14** *Let* $(\mathbf{R}, \mathbf{\Gamma})$ *be a maximal instance of CSP. For any Horn formula* $\Psi$*, if* $\mathbf{\Gamma} \models \Psi$*, then* $\mathbf{R} \models \Psi$*.*

*Proof* Let $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$ and

$$\Psi = \forall x_1, ..., x_{n_u} \exists x_{n_u+1} ... x_n \, \Xi(x_1, ..., x_n) \to \pi_u(x_1, ..., x_{n_u})$$

where

$$\Xi(x_1, ..., x_n) = \bigwedge_{t=1}^{N} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ..., x_{o_{tn_{j_t}}})$$

such that $\mathbf{\Gamma} \models \Psi$. Let $h : V \to D$ be any mapping and $r_i = h^{-1}(\varrho_i)$. Let us prove that $\mathbf{R} \models \Psi$ where $\mathbf{R} = (V, r_1, ..., r_s)$.

Indeed, for any $\mathbf{a} \in r_i$ we have $h(\mathbf{a}) \in \varrho_i$, $i \in [s]$. From $\mathbf{\Gamma} \models \Psi$ we obtain that the following statement is true: if there exist $a_1, ..., a_n \in D$ such that $(a_{o_{t1}}, a_{o_{t2}}, ..., a_{o_{tn_{j_t}}}) \in \varrho_{j_t}$, $t \in [N]$, then $(a_1, ..., a_{n_u}) \in \varrho_u$.

Suppose now that we are given $b_1, ..., b_n \in V$ such that for any $t \in [N]$ we have $(b_{o_{t1}}, b_{o_{t2}}, ..., b_{o_{tn_{j_t}}}) \in r_{j_t}$. Therefore, for any $t \in [N]$ we have

$$(h(b_{o_{t1}}), h(b_{o_{t2}}), ..., h(b_{o_{tn_{j_t}}})) \in \varrho_{j_t}.$$

From $\mathbf{\Gamma} \models \Psi$ we obtain that $(h(b_1), ..., h(b_{n_u})) \in \varrho_u$. Therefore, $(b_1, ..., b_{n_u}) \in r_u$. Thus, we proved $\mathbf{R} \models \Psi$.

Finally, let $(\mathbf{R}, \mathbf{\Gamma})$ be a maximal instance of CSP and $\mathbf{R} = (V, r_1, ..., r_s)$. By the definition of the maximal instance, we have $r_i = \bigcap_{h \in \mathrm{Hom}(\mathbf{R},\mathbf{\Gamma})} h^{-1}(\varrho_i)$. Horn formulas have the following simple property: if $(V, r_1^1, ..., r_s^1) \models \Psi$ and $(V, r_1^2, ..., r_s^2) \models \Psi$, then $(V, r_1^1 \cap r_1^2, ..., r_s^1 \cap r_s^2) \models \Psi$. Since $(V, h^{-1}(\varrho_1), ..., h^{-1}(\varrho_s)) \models \Psi$ for any $h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$, we conclude $\mathbf{R} \models \Psi$. $\square$

Theorem 14 motivates the following approach to the problem Dense($\mathbf{\Gamma}$). Let $L = \{\Psi_1, ..., \Psi_c\}$ be a finite set of Horn formulas such that $\mathbf{\Gamma} \models \Psi_i$, $i \in [c]$. Given an instance $\mathbf{R} = (V, r_1, ..., r_s)$ of Dense($\mathbf{\Gamma}$), let us define an operator

$$q_i(r_1, ..., r_s) = r_i \cup \bigcup_{\Psi \in L : \Psi = \forall x_{1:n_i}(\Phi(x_1,...,x_{n_i}) \to \pi_i(x_1,...,x_{n_i}))} \Phi^{\mathbf{R}},$$

[2] We slightly abuse the standard terminology, according to which Horn formulas are defined more generally.

called the immediate consequence operator, i.e. it outputs a single application of the rules that contain $\pi_i$ as the head. This induces an operator on relational structures:

$$Q(\mathbf{R}) = (V, q_1(r_1, ..., r_s), ..., q_s(r_1, ..., r_s))$$

Since $q_i(r_1, ..., r_s) \supseteq r_i$, the Algorithm (2) eventually stops at the fixed point of the operator $Q(\mathbf{R})$, i.e. at $Q^{K-1}(\mathbf{R})$ where:

$$\mathbf{R}^0 = \mathbf{R}, \mathbf{R}^k = Q(\mathbf{R}^{k-1}), k \in [K], \mathbf{R}^K = \mathbf{R}^{K-1}. \tag{2}$$

In that algorithm we iteratively add new tuples to predicates $r_i, i \in [s]$ until all Horn formulas in $L$ are satisfied.

Let us denote the output $Q^{K-1}(\mathbf{R})$ of the Algorithm (2) by $\mathbf{R}^L = (V, r_1^L, ..., r_s^L)$. In fact, the Algorithm (2) calculates the fixed point of the operator $Q(\mathbf{R})$ in $O(|\mathbf{R}^L|)$ iterations, where $|\mathbf{R}^L| = \sum_{i=1}^s |r_i^L|$. It is easy to see that $\mathbf{R}^L = (V, r_1^L, ..., r_s^L)$ is a smallest (w.r.t. inclusion) relational structure $\mathbf{T} = (V, t_1, ..., t_s)$ such that $t_i \supseteq r_i, i \in [s]$ and $\mathbf{T} \vDash \Psi_i$, $i \in [c]$. Therefore, $\mathbf{R}^L$ is a good candidate for a maximal instance $(\mathbf{R}' = (V, r_1', ..., r_s'), \mathbf{\Gamma})$, $r_i' \supseteq r_i, i \in [s]$.

**Definition 16** Let $\tau$ be a vocabulary and $\mathrm{F} \notin \tau$ be a stop symbol with an arity 0 assigned to it. Let $L$ be a finite set of Horn formulas over $\tau$ such that $\mathbf{\Gamma} \models \Psi, \Psi \in L$ and $L^{\text{stop}}$ be a finite set of formulas of the form $\Phi \to \mathrm{F}$ where $\Phi$ is a quantifier-free primitive positive formula over $\tau$. It is said that Dense($\mathbf{\Gamma}$) can be solved by the Datalog program $L \cup L^{\text{stop}}$, if for any instance $\mathbf{R}$ of Dense($\mathbf{\Gamma}$), we have: (a) if $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \emptyset$, then $(\mathbf{R}^L, \mathbf{\Gamma})$ is maximal and $\Phi^{\mathbf{R}^L} = \emptyset$ for any $(\Phi \to \mathrm{F}) \in L^{\text{stop}}$, and (b) if $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$, then there is $(\Phi \to \mathrm{F}) \in L^{\text{stop}}$ such that $\Phi^{\mathbf{R}^L} \neq \emptyset$.

**Theorem 15** *If* Dense($\mathbf{\Gamma}$) *can be solved by the Datalog program $L \cup L^{stop}$, then $\mathbf{\Gamma}$ is a DS-basis.*

*Proof* Any $\Psi \in L$ can be represented as

$$\Psi = \forall x_1, ..., x_n \Big( \bigwedge_{t=1}^{N} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ..., x_{o_{tn_{j_t}}}) \to \pi_u(x_1, ..., x_{n_u}) \Big).$$

For any sequence $v_1, ..., v_n \in V$ let us introduce an implication

$$R_\Psi(v_1, ..., v_n) \to \langle (v_1, ..., v_{n_u}), \varrho_u \rangle \tag{3}$$

where $R_\Psi(v_1, ..., v_n) = \big\{ \langle (v_{o_{t1}}, v_{o_{t2}}, ..., v_{o_{tn_{j_t}}}), \varrho_{j_t} \rangle | t \in [N] \big\} \subseteq \mathcal{C}_V^{\mathbf{\Gamma}}$. Analogously, any $\Psi \in L^{\text{stop}}$ can be represented as $\Psi = \big( \bigwedge_{t=1}^{N} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ..., x_{o_{tn_{j_t}}}) \to \mathrm{F} \big)$ and we define an implication

$$R_\Psi(v_1, ..., v_n) \to \mathcal{C}_V^{\mathbf{\Gamma}} \tag{4}$$

where $R_\Psi(v_1, ..., v_n) = \big\{ \langle (v_{o_{t1}}, v_{o_{t2}}, ..., v_{o_{tn_{j_t}}}), \varrho_{j_t} \rangle | t \in [N] \big\} \subseteq \mathcal{C}_V^{\mathbf{\Gamma}}$.

Let us denote

$$\Omega_\Psi^V = \bigcup_{v_1, ..., v_n \in V} \{ R_\Psi(v_1, ..., v_n) \to \langle (v_1, ..., v_{n_u}), \varrho_u \rangle \} \tag{5}$$

if $\Psi \in L$ and

$$\Omega_\Psi^V = \bigcup_{v_1, ..., v_n \in V} \{ R_\Psi(v_1, ..., v_n) \to \mathcal{C}_V^{\mathbf{\Gamma}} \}$$

if $\Psi \in L^{\text{stop}}$ and set

$$\Sigma = \bigcup_{\Psi \in L \cup L^{\text{stop}}} \Omega_\Psi^V$$

Let us first prove the inclusion $\Sigma^{\triangleright} \subseteq \Delta_1 \cup \Delta_2$ where

$$\Delta_1 = \{\mathcal{C}_{\mathbf{R}} \to B | B \subseteq \mathcal{C}_{\mathbf{R}^L}, \text{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \emptyset\}$$

and

$$\Delta_2 = \{\mathcal{C}_{\mathbf{R}} \to B | B \subseteq \mathcal{C}_V^{\mathbf{\Gamma}}, \text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset\}.$$

For this, it is enough to show that $\Delta_1 \cup \Delta_2$ is a full implicational system and $\Sigma \subseteq \Delta_1 \cup \Delta_2$. The mapping $\text{O} : 2^{\mathcal{C}_V^{\mathbf{\Gamma}}} \to 2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$, defined by $\text{O}(\mathcal{C}_{\mathbf{R}}) = \mathcal{C}_{\mathbf{R}^L}$ if $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) \neq \emptyset$ and $\text{O}(\mathcal{C}_{\mathbf{R}}) = \mathcal{C}_V^{\mathbf{\Gamma}}$ if $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$, is the closure operator by its construction. Therefore, Theorem 5 implies that the set $\Delta_1 \cup \Delta_2$ is a full implicational system. The fact $\Sigma \subseteq \Delta_1 \cup \Delta_2$ is obvious, because for any rule of the form (3), there exists an instance $\mathbf{R}$ such that $\mathcal{C}_{\mathbf{R}} = \{\langle (v_{o_{t1}}, v_{o_{t2}}, ..., v_{o_{tn_{j_t}}}), \varrho_{j_t} \rangle | t \in [N]\}$. The naive evaluation algorithm (2) will put the tuple $(v_1, ..., v_{n_u})$ into $r_u$ at the first iteration, because $(v_1, ..., v_{n_u}) \in q_u(\mathbf{R})$. Thus, the head of that rule $\langle (v_1, ..., v_{n_u}), \varrho_u \rangle$ will be in $\mathcal{C}_{\mathbf{R}^L}$. Analogously, any rule of the form (4) is also in $\Delta_1 \cup \Delta_2$. Thus, we proved $\Sigma^{\triangleright} \subseteq \Delta_1 \cup \Delta_2$, and next we need to prove $\Delta_1 \cup \Delta_2 \subseteq \Sigma^{\triangleright}$.

Note that the operator $Q(\mathbf{R})$ operates on $\mathbf{R} = (V, r_1, ..., r_s)$ by computing tuples from $q_i(r_1, ..., r_s), i \in [s]$ in the following way: computing $(v_1, ..., v_{n_i}) \in q_i(r_1, ..., r_s)$ can be modeled as a result of applying one of the rules (3) to attributes from $\mathcal{C}_{\mathbf{R}}$ to obtain the attribute $\langle (v_1, ..., v_{n_i}), \varrho_i \rangle$. Thus, $\mathcal{C}_{\mathbf{R}} \to \mathcal{C}_{Q(\mathbf{R})} \in \Sigma^{\triangleright}$. Therefore, $\mathcal{C}_{\mathbf{R}} \to \mathcal{C}_{Q^l(\mathbf{R})} \in \Sigma^{\triangleright}$ for any $l \in \mathbb{N}$, and we obtain $\mathcal{C}_{\mathbf{R}} \to \mathcal{C}_{\mathbf{R}^L} \in \Sigma^{\triangleright}$. Since $\Sigma^{\triangleright}$ is full, we conclude $\{\mathcal{C}_{\mathbf{R}} \to B | B \subseteq \mathcal{C}_{\mathbf{R}^L}\} \subseteq \Sigma^{\triangleright}$. Moreover, if $\text{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$, we can prove that any rule $\mathcal{C}_{\mathbf{R}} \to B$, $B \subseteq \mathcal{C}_V^{\mathbf{\Gamma}}$ is in $\Sigma^{\triangleright}$. This implies $\Delta_1 \cup \Delta_2 \subseteq \Sigma^{\triangleright}$.

In fact, we proved that the implicational system $\Sigma$ corresponds to the closure operator $\text{O} : 2^{\mathcal{C}_V^{\mathbf{\Gamma}}} \to 2^{\mathcal{C}_V^{\mathbf{\Gamma}}}$ (defined before) with respect to the canonical correspondence of Theorem 5. The closure operator O coincides with the densification operator Dense.

Thus, if $\text{Dense}(\mathbf{\Gamma})$ can be solved by Datalog program $L$, then the implicational system $\Sigma$ satisfies $\Sigma^{\triangleright} = \Sigma_V^{\mathbf{\Gamma}}$ and $\mathbf{\Gamma}$ is a DS-basis. □

Obviously, if $\text{Dense}(\mathbf{\Gamma})$ can be solved by some Datalog program $L \cup L^{\text{stop}}$, then all the more $\neg\text{CSP}(\mathbf{\Gamma})$ can be expressed by Datalog. The following theorems give examples of constraint languages for which $\text{Dense}(\mathbf{\Gamma})$ can be solved by Datalog.

**Theorem 16** *Let* $\mathbf{\Gamma} = (D = \{0, 1\}, \{(0)\}, \{(1)\}, \varrho_{x \wedge y \to z})$ *where* $\varrho_{x \wedge y \to z} = \{(a_1, a_2, a_3) \in D^3 | a_1 a_2 \leq a_3\}$. *Then, there is a finite set of Horn formulas* $L$ *over* $\tau = \{\pi_1, \pi_2, \pi_3\} \cup \{\text{F}\}$ *such that* $\text{Dense}(\mathbf{\Gamma})$ *can be solved by the Datalog program* $L$.

**Theorem 17** *Let* $\mathbf{\Gamma} = (D = \{0, 1\}, \varrho_1, \varrho_2, \varrho_3)$ *where* $\varrho_1 = \{(x, y) | x \vee y\}$, $\varrho_2 = \{(x, y) | \neg x \vee y\}$ *and* $\varrho_3 = \{(x, y) | \neg x \vee \neg y\}$. *Then, there is a finite set of Horn formulas* $L$ *over* $\tau = \{\pi_1, \pi_2, \pi_3\} \cup \{\text{F}\}$ *such that* $\text{Dense}(\mathbf{\Gamma})$ *can be solved by the Datalog program* $L$.

Proof of Theorem 16 is given in Section 15 and proof of Theorem 17 is given in Section 16.

## 11 Classification of Dense($\mathbf{\Gamma}$) for the Boolean case

The problem Dense($\mathbf{\Gamma}$) is tightly connected with the so-called implication and equivalence problems, parameterized by $\mathbf{\Gamma}$.

**Definition 17** Let $\mathbf{\Gamma} = (D, \varrho_1, ..., \varrho_s)$. The **implication problem**, denoted Impl($\mathbf{\Gamma}$), is a decision task with:

- **An instance:** two relational structures $\mathbf{R} = (V, r_1, ..., r_s)$ and $\mathbf{R}' = (V, r_1', ..., r_s')$.
- **An output:** yes, if Hom($\mathbf{R}, \mathbf{\Gamma}$) $\subseteq$ Hom($\mathbf{R}', \mathbf{\Gamma}$), and no, if otherwise.

Theorem 6.5 from [44] (which is based on the earlier result [45]) gives a complete classification of the computational complexity of Impl($\mathbf{\Gamma}$) for Boolean languages.

**Theorem 18** (Schnoor, Schnoor, 2008) *If $\Gamma$ is Schaefer, then Impl($\mathbf{\Gamma}$) can be solved in polynomial time. Otherwise, it is coNP-complete under logspace reductions.*

This theorem directly leads us to the classification of Dense($\mathbf{\Gamma}$).

**Theorem 19** *If $\Gamma$ is Schaefer, then* Dense($\mathbf{\Gamma}$) *is polynomially solvable. Otherwise, it is NP-hard.*

*Proof* Let us show that Dense($\mathbf{\Gamma}$) can be solved in polynomial time using an oracle access to Impl($\mathbf{\Gamma}$). Indeed, let $\mathbf{R} = (V, r_1, ..., r_s)$ be an instance of Dense($\mathbf{\Gamma}$). Then, $\langle(v_1, \cdots, v_d), \varrho\rangle \in \mathcal{C}_V^{\mathbf{\Gamma}}$ is in Dense($\mathcal{C}_{\mathbf{R}}$) if and only if Hom($\mathbf{R}, \mathbf{\Gamma}$) $\subseteq$ Hom($\mathbf{R}', \mathbf{\Gamma}$) where $\mathcal{C}_{\mathbf{R}'} = \{\langle(v_1, \cdots, v_d), \varrho\rangle\}$. Thus, by giving $(\mathbf{R}, \mathbf{R}')$ to an oracle of Impl($\mathbf{\Gamma}$), we decide whether $\langle(v_1, \cdots, v_d), \varrho\rangle \in$ Dense($\mathcal{C}_{\mathbf{R}}$). By doing this for all $\langle(v_1, \cdots, v_d), \varrho\rangle \in \mathcal{C}_V^{\mathbf{\Gamma}}$, we compute the whole set Dense($\mathcal{C}_{\mathbf{R}}$) in polynomial time.

Thus, Dense($\mathbf{\Gamma}$) is polynomial time Turing reducible to Impl($\mathbf{\Gamma}$), and therefore, using Theorem 18, is polynomially solvable if $\Gamma$ is Schaefer.

Let us now show that $\neg$Impl($\mathbf{\Gamma}$) is polynomial time Turing reducible to Dense($\mathbf{\Gamma}$). Given an instance $(\mathbf{R}, \mathbf{R}')$ of $\neg$Impl($\mathbf{\Gamma}$), the inclusion Hom($\mathbf{R}, \mathbf{\Gamma}$) $\subseteq$ Hom($\mathbf{R}', \mathbf{\Gamma}$) holds if and only if $\mathcal{C}_{\mathbf{R}'} \subseteq$ Dense($\mathcal{C}_{\mathbf{R}}$). Thus, by computing Dense($\mathcal{C}_{\mathbf{R}}$) one can efficiently decide whether $\mathcal{C}_{\mathbf{R}'} \subseteq$ Dense($\mathcal{C}_{\mathbf{R}}$), i.e. whether Hom($\mathbf{R}, \mathbf{\Gamma}$) $\subseteq$ Hom($\mathbf{R}', \mathbf{\Gamma}$). If $\mathcal{C}_{\mathbf{R}'} \not\subseteq$ Dense($\mathcal{C}_{\mathbf{R}}$), our reduction outputs "yes", and it outputs "no", if otherwise.

If $\Gamma$ is not Schaefer, then $\neg$Impl($\mathbf{\Gamma}$) is NP-complete, and therefore, Dense($\mathbf{\Gamma}$) is NP-hard. $\square$

## 12 Non-Schaefer languages and mP/poly

For our proof of Theorem 6, we need to show Dense($\mathbf{\Gamma}$) $\notin$ mP/poly for non-Schaefer languages. Note that under NP $\not\subseteq$ P/poly (which is widely believed to be true), any NP-hard problem is outside of P/poly. Therefore, if $\Gamma$ is not Schaefer, then Dense($\mathbf{\Gamma}$) $\notin$ P/poly (and all the more, Dense($\mathbf{\Gamma}$) $\notin$ mP/poly). In the current section we prove Dense($\mathbf{\Gamma}$) $\notin$ mP/poly *unconditionally* and this fact will be used in Section 13.

**Theorem 20** *Let $\Gamma$ be a non-Schaefer language. Then,* Dense($\mathbf{\Gamma}$) $\notin$ *mP/poly.*

**Lemma 2** *For any language* $\Gamma$ *that is not constant preserving, if* $\mathrm{Dense}(\mathbf{\Gamma}) \in$ *mP/poly, then* $\neg CSP(\mathbf{\Gamma}) \in$ *mP/poly.*

*Proof* Let $\mathbf{R}$ be an instance of $\mathrm{CSP}(\mathbf{\Gamma})$ and $\{x_v\}_{v\in\mathcal{C}_V^\Gamma}$ be input boolean variables of a monotone polynomial-size circuit that computes $\mathrm{Dense}(\mathcal{C}_\mathbf{R})$ such that $x_v = 1$ if and only if $v \in \mathcal{C}_\mathbf{R}$. Let $\{y_v\}_{v\in\mathcal{C}_V^\Gamma}$ be output variables of that circuit, and $y_v = 1$ indicates $v \in \mathrm{Dense}(\mathcal{C}_\mathbf{R})$. Then, $\bigwedge_{v\in\mathcal{C}_V^\Gamma} y_v = 1$ if and only if $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$. Thus, emptiness of $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$ can be decided by a polynomial-size monotone circuit. Therefore, $\neg\mathrm{CSP}(\mathbf{\Gamma}) \in$ mP/poly. $\square$

In the case $D = \{0, 1\}$, there is a countable number of clones: in the list below we use the notation from the table on page 76 of [58] (the same results can be found in the table on page 1402 of [59]), together with the notation from the Table 1 of [60]. For every row, listed relations form a basis of the relational co-clone corresponding to the functional clone (notations of clones are given according to [58] and [60]). At the same time, the functional clone equals the set of polymorphisms of the relations. Below we list all Post co-clones $\Gamma$ except for those that: a) satisfy $\{(0)\}, \{(1)\} \in \Gamma$ and b) $\Gamma$ is Schaefer (and we are not interested in such languages in the current section).

| [58] | [60] | Basis of clone | Basis of co-clone |
|---|---|---|---|
| $U$ | $N$ | $\{\neg x, 1\}$ | $\varrho_\mathrm{b} = \big\{\{(x_1, x_2, x_3) \mid x_1 = x_2 \vee x_2 = x_3\}\big\}$ |
| $SU$ | $N_2$ | $\{\neg x\}$ | $\varrho_\mathrm{NAE} = \big\{\{(x_1, x_2, x_3) \mid x_1 \neq x_2 \vee x_1 \neq x_3\}\big\}$ |
| $MU$ | $I$ | $\{0, 1\}$ | $\big\{\{(x_1, x_2) \mid x_1 \leq x_2\}, \varrho_\mathrm{b}\big\}$ |
| $U_0$ | $I_0$ | $\{0\}$ | $\big\{\{(0)\}, \varrho_\mathrm{b}\big\}$ |
| $U_1$ | $I_1$ | $\{1\}$ | $\big\{\{(1)\}, \varrho_\mathrm{b}\big\}$ |

(6)

Next, we will concentrate on languages listed in Table 6.

Our first goal is to study the complexity of $\mathrm{Dense}(\mathbf{\Gamma})$ where $\mathbf{\Gamma} = (\{0, 1\}, \varrho_\mathrm{b})$ where $\varrho_\mathrm{b} = \{(x_2, x_1, x_3) | x_1 = x_2 \vee x_1 = x_3\}$.

**Lemma 3** $\mathrm{Dense}(\mathbf{\Gamma} = (\{0, 1\}, \varrho_\mathrm{b})) \notin$ *mP/poly.*

*Proof* Let us introduce the restriction of $\mathrm{CSP}(\mathbf{\Gamma})$, $\mathbf{\Gamma} = (\{0, 1\}, \varrho_\mathrm{b}, \{(0)\}, \{(1)\})$, in which we assume that in its instance $\mathbf{R} = (V, r, \{Z\}, \{O\})$ the domain $V$ contains two designated variables, $Z$ and $O$, with unary constraints, $Z = 0$ and $O = 1$. This task is denoted by $\mathrm{CSP}_\mathrm{b}$.

It is easy to see that

$$\varrho_\mathrm{NAE}(x, y, z) = \exists t, O, Z\ \varrho_\mathrm{b}(x, t, z) \wedge \varrho_\mathrm{b}(t, Z, y) \wedge \varrho_\mathrm{b}(t, O, y) \wedge [O = 1] \wedge [Z = 0]$$

where $\varrho_\mathrm{NAE} = \{(x_1, x_2, x_3) | x_1 \neq x_2 \vee x_1 \neq x_3\}$. Thus, by $\mathrm{CSP}_\mathrm{b}$ we can model any instance of $\mathrm{CSP}(\{\varrho_\mathrm{NAE}\})$. The standard reduction of $\mathrm{CSP}(\{\varrho_\mathrm{NAE}\})$ to $\mathrm{CSP}_\mathrm{b}$ can be implemented as a monotone circuit. Since $\{\varrho_\mathrm{NAE}, \{(0)\}, \{(1)\}\}$ is not of bounded width and $\neg\mathrm{CSP}(\{\varrho_\mathrm{NAE}\})$ is equivalent to $\neg\mathrm{CSP}(\{\varrho_\mathrm{NAE}, \{(0)\}, \{(1)\}\})$ modulo polynomial-size reductions by monotone circuits (see analogous argument in the proof of Corollary 1), we conclude $\neg\mathrm{CSP}(\{\varrho_\mathrm{NAE}\}) \notin$ mP/poly (using Proposition 5.1. from [48]). Therefore, $\neg\mathrm{CSP}_\mathrm{b} \notin$ mP/poly.

Let us now prove that $\mathrm{Dense}(\mathbf{\Gamma})$, where $\mathbf{\Gamma} = (\{0, 1\}, \varrho_\mathrm{b})$, is outside of mP/poly. Let $\mathbf{R} = (V, r)$ be an instance of $\mathrm{Dense}(\mathbf{\Gamma} = (\{0, 1\}, \varrho_\mathrm{b}))$ and let $\mathbf{R}' = (V, r)$ be such that $r' \supseteq r$ and $(\mathbf{R}', \mathbf{\Gamma})$ is a maximal instance. By construction, for any $i, j \in V$, $(i, j, i) \in r'$

if and only if there is no such $h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$ that satisfies $h(i) = 0$ and $h(j) = 1$. But the last question, i.e. checking the emptiness of $\{h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) | h(i) = 0, h(j) = 1\}$ is equivalent to $\neg\mathrm{CSP}_{\mathrm{b}}$ after setting $Z = i$, $O = j$. The latter argument can be turned into a reduction of $\neg\mathrm{CSP}_{\mathrm{b}}$ to $\mathrm{Dense}(\mathbf{\Gamma} = (\{0, 1\}, \varrho_{\mathrm{b}}))$. Again, this reduction can be implemented as a monotone circuit.

Therefore, $\mathrm{Dense}(\mathbf{\Gamma} = (\{0, 1\}, \varrho_{\mathrm{b}})) \notin \mathrm{mP/poly}$. □

**Lemma 4** *If $\langle\Gamma\rangle$ equals one of $inv(U_0)$, $inv(U_1)$, $inv(SU)$, $inv(MU)$ and $inv(U)$, then $\varrho_{\mathrm{b}}$ is strongly reducible to $\Gamma$.*

*Proof* Let $\Gamma = \{\rho_1, \cdots, \rho_s\}$. Since $\varrho_{\mathrm{b}} \in \mathrm{inv}(U) \subseteq \mathrm{inv}(U_0), \mathrm{inv}(U_1), \mathrm{inv}(SU), \mathrm{inv}(MU)$, then $\varrho_{\mathrm{b}} = \Psi^{\mathbf{\Gamma}}$ for a primitive positive formula $\Psi$ over $\tau = \{\pi_1, \cdots, \pi_s\}$. Let

$$\Psi = \exists x_4 ... x_l \bigwedge_{t \in [N]} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ...).$$

Let us denote $\Phi = \bigwedge_{t \in [N]} \pi_{j_t}(x_{o_{t1}}, x_{o_{t2}}, ...)$ and consider a relation $\gamma = \{\mathbf{x} \in \{0, 1\}^l \mid \mathbf{x} \in \Phi^{\mathbf{\Gamma}}$ or $\mathbf{x}_{1:3} \notin \varrho_{\mathrm{b}}\}$. Let us prove that if $u \in \mathrm{pol}(\Phi^{\mathbf{\Gamma}})$ and $u$ is unary, then $u \in \mathrm{Pol}(\Gamma)$. The latter can be checked by considering all 4 cases: $u(x) = x$, or $\neg x$, or 0, or 1. A unary $u(x) = x$ is a polymorphism of any relation. If $u(x) = c$, then $u \in \mathrm{pol}(\Phi^{\mathbf{\Gamma}})$ means that $\Phi^{\mathbf{\Gamma}}$ is a $c$-preserving relation. Then $\gamma$ is also $c$-preserving. Finally, if $u(x) = \neg x$, then $u \in \mathrm{pol}(\Phi^{\mathbf{\Gamma}})$ means that $\Phi^{\mathbf{\Gamma}}$ is a self-dual relation. Therefore, $\gamma = \Phi^{\mathbf{\Gamma}} \cup \{(0, 1, 0), (1, 0, 1)\} \times D^{l-3}$ is also self-dual, i.e. $u \in \mathrm{Pol}(\Gamma)$.

From the last fact we conclude that $\{u : D \to D \mid u \in \mathrm{Pol}(\Gamma)\} \subseteq \{u : D \to D \mid u \in \mathrm{pol}(\{\gamma\})\}$. Since $\{u : D \to D \mid u \in \mathrm{Pol}(\Gamma)\}$ forms a basis of $\mathrm{Pol}(\Gamma)$ (in all listed cases), then $\gamma \in \mathrm{inv}(\mathrm{Pol}(\Gamma))$, i.e. $\gamma \in \langle\Gamma\rangle$.

Finally, by construction we have $\gamma = \Phi^{\mathbf{\Gamma}} \cup \delta$ where $\mathrm{pr}_{1,2,3}\delta = D^3 \setminus \varrho_{\mathrm{b}}$ and $\mathrm{pr}_{1,2,3}\Phi^{\mathbf{\Gamma}} = \varrho_{\mathrm{b}}$. This is exactly the needed condition for $\varrho_{\mathrm{b}}$ to be strongly reducible to $\Gamma$. □

*Proof of Theorem 20* We have $D = \{0, 1\}$. Our goal is to prove that if $\Gamma$ is non-Schaefer then $\mathrm{Dense}(\mathbf{\Gamma})$ is outside of mP/poly.

Let us first consider the subcase where $\mathrm{CSP}(\mathbf{\Gamma})$ is NP-hard. Then, by construction, $\Gamma$ is not constant preserving and $\mathrm{core}(\Gamma) = \Gamma$. Therefore, $\neg\mathrm{CSP}(\Gamma)$ and $\neg\mathrm{CSP}(\Gamma \cup \{(0)\} \cup \{(1)\})$ can be mutually reduced by polynomial-size monotone circuits (as in the proof of corollary 1). Since $\neg\mathrm{CSP}(\Gamma) \equiv \neg\mathrm{CSP}(\Gamma \cup \{(0)\} \cup \{(1)\}) \notin \mathrm{mP/poly}$ (by proposition 5.1. from [48]), then, by Lemma 2, $\mathrm{Dense}(\mathbf{\Gamma}) \notin \mathrm{mP/poly}$.

Next, let us consider the subcase where $\mathrm{CSP}(\mathbf{\Gamma})$ is tractable. Since we already assumed that $\Gamma$ is not any of 4 Schaeffer classes, this can happen only if $\Gamma$ is constant preserving. Therefore, $\{\{0\}, \{1\}\} \nsubseteq \langle\Gamma\rangle$. All possible variants for $\mathrm{Pol}(\Gamma)$ are listed in Table 6. Since $\langle\{\varrho_{\mathrm{b}}\}\rangle = \mathrm{inv}(U) \subseteq \mathrm{inv}(U_0), \mathrm{inv}(U_1), \mathrm{inv}(SU), \mathrm{inv}(MU)$, Lemma 3 in combination with Lemma 4 and part (c) of Theorem 11 gives us that $\mathrm{Dense}(\mathbf{\Gamma}) \notin \mathrm{mP/poly}$. □

## 13 Proof of Theorem 6

Let us prove first that for the Boolean domain $D = \{0, 1\}$, if $\Gamma$ satisfies one of the following 3 conditions

(a) $\Gamma$ is a subset of $\langle\{\varrho_1, \varrho_2, \varrho_3\}\rangle$ where $\varrho_1 = \{(x, y)\,|x \vee y\}$, $\varrho_2 = \{(x, y)\,|\neg x \vee y\}$ and $\varrho_3 = \{(x, y)\,|\neg x \vee \neg y\}$ (2-SAT);
(b) $\Gamma$ is a subset of $\langle\{\{(0)\}, \{(1)\}, \varrho_{x\wedge y\rightarrow z}\}\rangle$ (Horn case);
(c) $\Gamma$ is a subset of $\langle\{\{(0)\}, \{(1)\}, \varrho_{\neg x\wedge\neg y\rightarrow\neg z}\}\rangle$ (dual-Horn case).

then it has a weak polynomial densification operator.

Note that from Theorems 9 and 10 it follows that in all three cases $\Gamma$ is a subset of the relational clone of an A-language. Part (b) of Theorem 11 claims that $\Gamma$ has a weak polynomial densification operator if languages $\{\varrho_1, \varrho_2, \varrho_3\}$, $\{\{(0)\}, \{(1)\}, \varrho_{x\wedge y\rightarrow z}\}$ have one. Theorems 15, 16 and 17 give us that $(D, \varrho_1, \varrho_2, \varrho_3)$, $(D, \{(0)\}, \{(1)\}, \varrho_{x\wedge y\rightarrow z})$ are DS-templates. Therefore, $\Gamma$ has a weak polynomial densification operator.

It remains to prove that, in the Boolean case, the weak polynomial densification property implies one of these 3 conditions.

For the general domain $D$, if a constraint language $\Gamma$ has a weak polynomial densification operator, then its core is of bounded width (Theorem 8). Thus, in the Boolean case, if $\Gamma$ is not constant-preserving and has a weak polynomial densification operator, then it is of bounded width (i.e. $\Gamma$ is in one of the latter three classes). If $\Gamma$ preserves some constant $c$, then w.l.o.g. we can assume that $c = 0$. From Theorem 4, whose proof is given in Section 11, it is clear that either a) Dense($\mathbf{\Gamma}$) is NP-hard, or b) $\Gamma$ is Schaefer, i.e. $\{\{0\}, \{1\}\} \cup \Gamma$ is tractable. In the first case, existence of a polynomial-size implicational system for the densification operator implies that there exists a monotone circuit of size poly($|V|$) that computes the densification operator Dense (a construction of such a circuit is identical to the one given in Theorem 8). In other words, Dense($\mathbf{\Gamma}$) $\in$ mP/poly. This contradicts to the claim of Theorem 20 that Dense($\mathbf{\Gamma}$) $\notin$ mP/poly for non-Schaefer languages.

Thus, we have option b), and this can happen only if either b.1) $\Gamma$ preserves $\vee$, or $\wedge$, or $\mathrm{mjy}(x, y, z) = (x\wedge y)\vee(x\wedge z)\vee(y\wedge z)$, or b.2) $\Gamma$ preserves $x\oplus y\oplus z$, but does not preserve $\vee$, $\wedge$ and mjy. In the first case, $\Gamma$ satisfies the needed conditions. In the second case, $\Gamma$ is a 0-preserving language, i.e. $0, x \oplus y \oplus z \in \mathrm{Pol}(\Gamma)$, but $\vee, \wedge, \mathrm{mjy} \notin \mathrm{Pol}(\Gamma)$. According to Table 2.1 on page 76 of Marchenkov's textbook [58], there are only two functional clones with these properties, i.e. either b.2.1) $\mathrm{Pol}(\Gamma) = L$ where $L = \{a_0 \oplus a_1x_1 \oplus \cdots \oplus a_kx_k\}$ is a set of all linear functions, or b.2.2) $\mathrm{Pol}(\Gamma) = L_0$ where $L_0 = \{a_1x_1 \oplus\cdots\oplus a_kx_k\}$. In both cases $\rho_L = \{(x, y, z, t) \mid x \oplus y \oplus z \oplus t = 0\} \in \langle\Gamma\rangle$.

**Lemma 5** *If* $Pol(\Gamma) = L_0$ *or* $Pol(\Gamma) = L$*, then* $\rho_L$ *is strongly reducible to* $\Gamma$.

*Proof* Note that $x \oplus y \in L_0 \subseteq L$. Therefore, for any $\varrho \in \langle\Gamma\rangle$ we have $\forall\mathbf{x}, \mathbf{y} \in \varrho \rightarrow \mathbf{x}\oplus\mathbf{y} \in \varrho$ where $\oplus$ is applied component-wise, i.e. $\varrho$ is a linear subspace. Since $\rho_L \in \langle\Gamma\rangle$, then there is a quantifier-free primitive positive formula $\Phi(x_1, \cdots, x_l)$ such that $\rho_L = \mathrm{pr}_{1,2,3,4}\Phi^{\mathbf{\Gamma}}$. Let us set $\Psi(x_1, \cdots, x_l) = \exists x_4\Phi(x_1, \cdots, x_l)$, i.e. $\Psi$ depends on $x_4$ fictitiously. Let us define $\delta = \Psi^{\mathbf{\Gamma}} \setminus \Phi^{\mathbf{\Gamma}}$. Thus, we have $\Phi^{\mathbf{\Gamma}} \cup \delta \in \langle\Gamma\rangle$, $\rho_L = \mathrm{pr}_{1,2,3,4}\Phi^{\mathbf{\Gamma}}$ and $\mathrm{pr}_{1,2,3,4}\delta = \mathrm{pr}_{1,2,3,4}\Psi^{\mathbf{\Gamma}} \setminus \Phi^{\mathbf{\Gamma}} = \mathrm{pr}_{1,2,3,4}\{\mathbf{x} \oplus a(0, 0, 0, 1, 0, \cdots, 0) \mid a \in D, \mathbf{x} \in \Phi^{\mathbf{\Gamma}}\} \setminus \Phi^{\mathbf{\Gamma}} = \mathrm{pr}_{1,2,3,4}\{\mathbf{x}\oplus (0, 0, 0, 1, 0, \cdots, 0) \mid \mathbf{x} \in \Phi^{\mathbf{\Gamma}}\} = \{(x, y, z, t) \mid x\oplus y\oplus z\oplus t = 1\} = D^4 \setminus \rho_L$. The latter is the condition for strong reducibility of $\rho_L$ to $\Gamma$. □

Using part (b) of Theorem 11, the weak polynomial densification property of $\Gamma$ and the latter lemma, we obtain that $\{\rho_L\}$ has a weak polynomial densification operator. The following Lemma contradicts to our conclusion. Therefore, in the Boolean case, the weak polynomial densification property implies one of 3 conditions given above.

**Lemma 6** $\{\rho_L\}$ *does not have a weak polynomial densification operator.*

*Proof* Let us prove the statement by reductio ad absurdum. Suppose that $\Gamma = \{\rho_L\}$ has a weak polynomial densification operator. Therefore, $\text{Dense}(\mathbf{\Gamma}) \in \text{mP/poly}$.

Since the core of $\Gamma' = \{\varrho, \{(0)\}, \{(1)\}\}$ where $\varrho = \{(x, y, z) \mid x \oplus y \oplus z = 0\}$ is not of bounded width, by proposition 5.1 from [48], $\neg\text{CSP}(\mathbf{\Gamma}')$ cannot be computed by a polynomial-size monotone circuit. Let us describe a monotone reduction of $\neg\text{CSP}(\mathbf{\Gamma}')$ to $\text{Dense}(\mathbf{\Gamma})$ which will imply $\neg\text{CSP}(\mathbf{\Gamma}') \in \text{mP/poly}$. This will be a contradiction.

According to [58], $\Gamma = \{\rho_L\}$ is a basis of $\text{Inv}(L_0)$. Therefore, $\langle\{\rho_L\}\rangle$ equals the set of all linear subspaces in $\{0, 1\}^n, n \in \mathbb{N}$. In other words, for any $([n], r)$, $\text{Hom}(([n], r), \mathbf{\Gamma})$ is a linear subspace of $\{0, 1\}^n$, and $\{\text{pr}_{[k]}\text{Hom}(([n], r), \mathbf{\Gamma}) \mid n \in \mathbb{N}, k \leq n, r \subseteq [n]^4\}$ spans all possible linear subspaces.

Let $\mathbf{R}' = ([n], r', Z, O)$ be an instance of $\neg\text{CSP}(\mathbf{\Gamma}' = (D, \varrho, \{(0)\}, \{(1)\}))$. Since $\varrho, \{(0)\} \in \langle\{\rho_L\}\rangle$, a set of constraints $\{\langle(v_1, v_2, v_3), \varrho\rangle \mid (v_1, v_2, v_3) \in r'\} \cup \{\langle v, \{(0)\}\rangle \mid v \in Z\} \cup \{\langle(n+1, n+2, n+3), \varrho\rangle\}$ can be modeled as a set of constraints over $\{\rho_L\}$ with an extended set of variables $[m], m \geq n+3$, or alternatively, as an instance $\mathbf{R}'' = ([m], r)$ of $\text{CSP}(\{\rho_L\})$. Let $\sim$ be an equivalence relation on $[m+1]$ with equivalence classes $\big\{\{i\} \mid i \in [m] \setminus O\big\} \cup \big\{\{m+1\} \cup O\big\}$ and let $\overline{x}$ denote an equivalence class that contains $x \in [m+1]$. A relational structure $\mathbf{R} = ([m+1]/\sim, \overline{r})$ where $\overline{r} = \{(\overline{x}, \overline{y}, \overline{z}, \overline{t}) \mid (x, y, z, t) \in r\}$, considered as an instance of $\text{Dense}(\{\rho_L\})$, satisfies: $(\overline{n+1}, \overline{n+2}, \overline{n+3}, \overline{m+1}) \in \text{Dense}(\mathcal{C}_{\mathbf{R}})$ if and only if $\text{Hom}(\mathbf{R}', \mathbf{\Gamma}') = \emptyset$. Indeed, the constraint $\langle(n+1, n+2, n+3), \varrho\rangle$ that is satisfied for assignments in $\text{Hom}(\mathbf{R}'', \mathbf{\Gamma})$, together with $(\overline{n+1}, \overline{n+2}, \overline{n+3}, \overline{m+1}) \in \text{Dense}(\mathcal{C}_{\mathbf{R}})$, implies that $h(\overline{m+1}) = 0$ for any $h \in \text{Hom}(\mathbf{R}, \mathbf{\Gamma})$. Or, equivalently, $\{h \in \text{Hom}(\mathbf{R}, \mathbf{\Gamma}) \mid h(\overline{m+1}) = 1\} = \emptyset$. The latter is equivalent to $\{h \in \text{Hom}(\mathbf{R}'', \mathbf{\Gamma}) \mid h(x) = 1, x \in O\} = \emptyset$, or $\text{Hom}(\mathbf{R}', \mathbf{\Gamma}') = \emptyset$.

By construction, the indicator Boolean vector of the subset $\mathcal{C}_{\mathbf{R}''} \in 2^{\mathcal{C}^{\mathbf{\Gamma}}_{[m]}}$, i.e. the Boolean vector $\mathbf{x} \in \{0, 1\}^{\mathcal{C}^{\mathbf{\Gamma}}_{[m]}}$, $\mathbf{x}(\langle(v_1, v_2, v_3, v_4), \rho_L\rangle) = 1 \Leftrightarrow \langle(v_1, v_2, v_3, v_4), \rho_L\rangle \in \mathcal{C}_{\mathbf{R}''}$ can be computed from the indicator Boolean vector of $\mathcal{C}_{\mathbf{R}'} \in 2^{\mathcal{C}^{\mathbf{\Gamma}'}_{[n]}}$ by a polynomial-size monotone circuit. Further, the indicator Boolean vector of the subset $\mathcal{C}_{\mathbf{R}} \in 2^{\mathcal{C}^{\mathbf{\Gamma}}_{[m+1]/\sim}}$ can be computed by a polynomial-size monotone circuit from the indicator Boolean vector of $\mathcal{C}_{\mathbf{R}''} \in 2^{\mathcal{C}^{\mathbf{\Gamma}}_{[m]}}$ and the indicator Boolean vector of $O \in 2^{[n]}$. Finally, we feed the indicator vector of $\mathcal{C}_{\mathbf{R}}$ to $\text{Dense}(\mathbf{\Gamma})$ and compute whether $(\overline{n+1}, \overline{n+2}, \overline{n+3}, \overline{m+1}) \in \text{Dense}(\mathcal{C}_{\mathbf{R}})$. Thus, the emptiness of $\text{Hom}(\mathbf{R}', \mathbf{\Gamma}')$ can be decided by a polynomial-size monotone circuit which contradicts $\neg\text{CSP}(\Gamma') \notin \text{mP/poly}$. □

## 14 Proofs of Theorems 9 and 10

*Proof of Theorem 9* Let $\mathbf{\Gamma} = (D = \{0, 1\}, \varrho_1, \varrho_2, \varrho_3)$ where $\varrho_1 = \big\{(x, y) | x \vee y\big\}$, $\varrho_2 = \big\{(x, y) | \neg x \vee y\big\}$ and $\varrho_3 = \big\{(x, y) | \neg x \vee \neg y\big\}$.

First, let us note that any binary relation $\rho \subseteq D^2$ is strongly reducible to $\Gamma$, due to $\rho = \bigcap_{\gamma \in S: \rho \subseteq \gamma} \gamma$ where $S = \{\varrho_1, \varrho_2, \varrho_3, \varrho_2^T\}$, $\varrho_2^T = \{(y, x) \mid (x, y) \in \varrho_2\}$ (in the definition of strong reducibility one can set $\Xi(x, y) = \bigwedge_{i: \rho \subseteq \varrho_i} \pi_i(x, y) \bigwedge_{\rho \subseteq \varrho_2^T} \pi_2(y, x)$ and $\delta = D^2 \setminus \rho$).

It is well-known that $\langle\Gamma\rangle = \text{pol}(\text{mjy})$ where $\text{mjy}(x, y, z) = (x \wedge y) \vee (x \wedge z) \vee (y \wedge z)$ is a majority operation. Every $n$-ary relation $\rho \in \langle\Gamma\rangle$ is defined by its binary projections $\rho_{ij} = \{(x_i, x_j) \mid (x_1, \cdots, x_n) \in \rho\}$, i.e.

$$\rho = \bigcap_{i,j\in[n]} r_{ij}$$

where $r_{ij} = \{(x_1, \cdots, x_n) \mid (x_i, x_j) \in \rho_{ij}\}$. Since $\rho_{ij}$ is strongly reducible to $\Gamma$, $r_{ij}$ also has this property. Thus, $\rho$ is A-reducible to $\Gamma$, and therefore, $\Gamma$ is an A-language. $\square$

**The Horn case** Let $\mathbf{\Gamma} = (D = \{0, 1\}, \{(0)\}, \{(1)\}, \varrho_{x\wedge y\to z})$. In other words, $\langle\Gamma\rangle$ is a set of relations that is closed under component-wise conjunction, i.e. $\mathbf{x}, \mathbf{y} \in \rho \in \langle\Gamma\rangle$ implies $\mathbf{x} \wedge \mathbf{y} \in \rho$.

**Lemma 7** *Let $D = \{0, 1\}$ and $\rho$ be a set of satisfying assignments of a Horn clause, i.e.*

$$\rho = \{(x_1, \cdots, x_n) \mid (x_1 \wedge \cdots \wedge x_n \to 0)\}$$

*or*

$$\rho = \{(x_1, \cdots, x_{n+1}) \mid (x_1 \wedge \cdots \wedge x_n \to x_{n+1})\}.$$

*Then, $\rho$ is strongly reducible to $\Gamma$.*

*Proof* Let us consider first the case of $\Phi = (x_1 \wedge \cdots \wedge x_n \to 0)$. This formula can be given as $\Phi \equiv \exists x_{n+1}, \cdots, x_{2n-1} \Xi(x_1, \cdots, x_{2n-1})$ where

$$\Xi(x_1, \cdots, x_{2n-1}) = (x_1 \wedge x_2 \to x_{n+1}) \wedge (x_{2n-1} = 0) \bigwedge_{i=3}^{n} (x_i \wedge x_{n+i-2} \to x_{n+i-1}).$$

If we define a $2n - 1$-ary $\delta$ as $\{(1, \cdots, 1)\}$, then it can be checked that $\Xi^{\mathbf{\Gamma}} \cup \delta$ is a $\wedge$-closed set. Indeed, for any $\mathbf{x} \in \Xi^{\mathbf{\Gamma}}$ and $\mathbf{y} \in \delta$, we have $\mathbf{x} \wedge \mathbf{y} = \mathbf{x} \in \Xi^{\mathbf{\Gamma}} \cup \delta$. Since both $\Xi^{\mathbf{\Gamma}}$ and $\delta$ are $\wedge$-closed, then we conclude the statement. Therefore, $\Xi^{\mathbf{\Gamma}} \cup \delta \in \langle\Gamma\rangle$. It remains to check that $\mathrm{pr}_{1:n}\Xi^{\mathbf{\Gamma}} = \rho$ and $\mathrm{pr}_{1:n}\delta = \{0, 1\}^n \setminus \rho$. Thus, $\Xi^{\mathbf{\Gamma}} \cup \delta \in \langle\Gamma\rangle$ and $\rho = \{(x_1, \cdots, x_n) \mid (x_1 \wedge \cdots \wedge x_n \to 0)\}$ is strongly reducible to $\Gamma$.

Let us now consider the case of $\Phi = (x_1 \wedge \cdots \wedge x_n \to x_{n+1})$. Let us denote by $(x \wedge y = z)$ the formula $(x \wedge y \to z) \wedge (z \wedge O \to x) \wedge (z \wedge O \to y) \wedge (O = 1)$ where $O$ is an additional fixed variable. Note that $(x \wedge y = z)$ is a quantifier-free primitive positive formula over $\tau$. Thus, we have $\Phi \equiv \exists x_{n+2}, \cdots, x_{2n-1}, O \; \Xi(x_1, \cdots, x_{2n-1}, O)$ where

$$\Xi(x_1, \cdots, x_{2n-1}, O) = (x_1 \wedge x_2 = x_{n+2}) \wedge (x_n \wedge x_{2n-1} \to x_{n+1}) \wedge \bigwedge_{i=3}^{n-1} (x_i \wedge x_{n+i-1} = x_{n+i}).$$

Here we define a $2n$-ary $\delta$ as $\{1\}^n \times \{0\} \times \{1\}^{n-1}$. Let us prove that $\Xi^{\mathbf{\Gamma}} \cup \delta$ is a $\wedge$-closed set. Again, let us consider $\mathbf{x} \in \Xi^{\mathbf{\Gamma}}$ and $\mathbf{y} \in \delta$. If $x_{n+1} = 0$, then $\mathbf{x} \wedge \mathbf{y} = \mathbf{x} \in \Xi^{\mathbf{\Gamma}} \cup \delta$. Otherwise, if $x_{n+1} = 1$, we have either a) $\mathbf{x} = 1^{2n-1}$ and in that case $1^{2n-1} \wedge \mathbf{y} = \mathbf{y} \in \Xi^{\mathbf{\Gamma}} \cup \delta$, or b) at least one of $x_1, \cdots, x_n$ is 0. In the case of b) let $i \in [n]$ be the smallest such that $x_i = 0$, i.e. $x_j = 1$, $j \in [i - 1]$. Therefore, $x_{n+j} = 1$, $j \in [2, i - 1]$ and $x_{n+j} = 0$, $j \in [i, n - 1]$. It remains to check that an assignment $\mathbf{x} \wedge \mathbf{y} = (x_1, \cdots, x_n, 0, x_{n+2}, \cdots, x_{2n-1})$ also satisfies $\Xi$, and therefore, is in $\Xi^{\mathbf{\Gamma}} \cup \delta$. Thus, $\Xi^{\mathbf{\Gamma}} \cup \delta \in \langle\Gamma\rangle$ and $\rho$ is strongly reducible to $\Gamma$. $\square$

*Proof of Theorem 10* Let $\rho \in \langle\Gamma\rangle$ be $n$-ary, i.e. $\rho$ is closed with respect to component-wise conjunction. A classical result about $\wedge$-closed relations (see [61, 62]) states that $\rho$ can be represented as:

$$\rho = \bigcap_{i=1}^{l} \rho_i$$

where $\rho_i = \{(x_1, \cdots, x_n) \mid \Phi_i(x_{s_{i1}}, \cdots, x_{s_{ir_i}})\}$ where $\Phi_i$ is a Horn clause. From the previous Lemma we conclude that each of $\rho_i, i \in [l]$ is strongly reducible to $\Gamma$. Therefore, $\rho$ is A-reducible to $\Gamma$. Since this is true for any $\rho \in \langle\Gamma\rangle$, we conclude that $\Gamma$ is an A-language. □

## 15 Proof of Theorem 16

In this case we have a vocabulary $\tau = \{\pi_1, \pi_2, \pi_3\}$ where $\pi_1, \pi_2$ are unary and $\pi_3$ is assigned an arity 3.

Let $\mathbf{R} = (V, Z, O, r)$ be an instance of Dense($\mathbf{\Gamma}$). Let us define an implicational system $\Sigma$ on $V$ that consists of rules $\{i, j\} \to k$ for any $(i, j, k) \in r$. The implicational system $\Sigma$ defines a closure operator $o_\Sigma(S) = \{x|(S \to x) \in \Sigma^\triangleright\}$. Let $\mathbf{R}' = (V, Z', O', r')$ be a maximal instance such that $Z' \supseteq Z$, $O' \supseteq O$, $r' \supseteq r$ and Hom($\mathbf{R}, \mathbf{\Gamma}$) = Hom($\mathbf{R}', \mathbf{\Gamma}$) $\neq \emptyset$. Note that $(i, j, k) \in r'$ if and only if $k \in o_\Sigma(\{i, j\} \cup O)$ and $Z \cap o_\Sigma(\{i, j\} \cup O) = \emptyset$. Indeed, for any $k \in o_\Sigma(\{i, j\} \cup O)$ we have $(i, j, k) \in r'$, because $\{i, j\} \cup O \to k$ is a consequence of rules in $r$. On the contrary, let $k \notin o_\Sigma(\{i, j\} \cup O)$. Then, $h : V \to D$ defined by $h(v) = 1$ if $v \in o_\Sigma(\{i, j\} \cup O)$ and $h(v) = 0$, if otherwise, is a homomorphism from $\mathbf{R}$ to $\mathbf{\Gamma}$. Therefore, for any $k \notin o_\Sigma(\{i, j\} \cup O)$ we have $(h(i), h(j), h(k)) \notin \varrho_3$. Using Theorem 2, we obtain $(i, j, k) \notin r'$.

Thus, for any $(i, j, k) \in r'$ there exists a derivation of $k$ from $\{i, j\} \cup O$ using only rules $\{i, j\} \to k$, $(i, j, k) \in r$. To such a derivation one can always correspond a rooted binary tree $T$ whose nodes are labeled with elements of $V$, the root is labeled with $k$, and all leaves are labeled by elements of $\{i, j\} \cup O$. Any (non-leaf) node $p$ (a parent) of the tree $T$ has two children $c_1, c_2$ such that $\{l(c_1), l(c_2)\} \to l(p)$ is in $\Sigma$ ($l$ is a labeling function).

Let $x, y$ be two leaves of the tree $T$ with a common parent $z$ such that the distance from $x$ to the root $k$ equals the depth of the tree (i.e. is the largest possible one). The parent of $z$ is denoted by $u$ and all possible branches under $u$ are drawn in Fig. 1: we reduced the number of possible branches to analyze using the rule $\pi_3(x, y, u) \to \pi_3(y, x, u)$ that makes an order of children irrelevant. Circled leaves correspond to leaves labeled by elements of $O$. A leaf that is not circled can be labeled either by $i, j$ or by an element from $O$. For each case, the Figure shows how to reduce the tree $T$ by deleting redundant nodes under $u$. To delete the redundant nodes and connect leaves to $u$ we have to verify that a new reduced branch with a parent $u$ and 2 leaves $x, y$ (or, $x, t$) corresponds to a triple $(x, y, u) \in r^L$ (or, $(x, t, u) \in r^L$), i.e. the resulting triple can be obtained using rules from $L$. Needed rules are indicated near each deletion operation in Fig. 1.

It is easy to see that using such deletions we will eventually obtain a root $k$ with two children labeled by $c_1, c_2 \in \{i, j\} \cup O$. Therefore, the triple $(c_1, c_2, k)$ is in $r^L$. If $\{c_1, c_2\} = \{i, j\}$, then $(i, j, k)$ can be obtained from $(c_1, c_2, k)$ using the rule (1) from the list below. If $c_1 = i$ and $c_2 \in O$ (or, $c_1, c_2 \in O$), then $(i, j, k)$ can be obtained from $(c_1, c_2, k)$ using the rule (2). Thus, $(i, j, k) \in r^L$, i.e. we proved that $r' = r^L$.

Let us show now that $O' = O^L$. Analogously to the previous analysis, $k \in o_\Sigma(O)$ if there is a derivation tree with a root $k$ labeled with elements of $V$ and all leaves are labeled by elements of $O$. Using the same reduction we finally obtain the triple $(i, j, k) \in r^L$, where $i, j \in O$. Using the rule (3), we conclude $k \in O^L$, i.e. we proved the inclusion $O^L \supseteq o_\Sigma(O)$. Therefore, $O^L = o_\Sigma(O)$. Then, $h : V \to D$ defined by $h(v) = 1$ if $v \in o_\Sigma(O)$ and $h(v) = 0$, if otherwise, is a homomorphism from $\mathbf{R}$ to $\mathbf{\Gamma}$. Since for any $v \notin O^L$ we have $h(v) \notin \varrho_2$, then using Theorem 2, we obtain that $o_\Sigma(O) = O^L$ is maximal and $O' = O^L$.

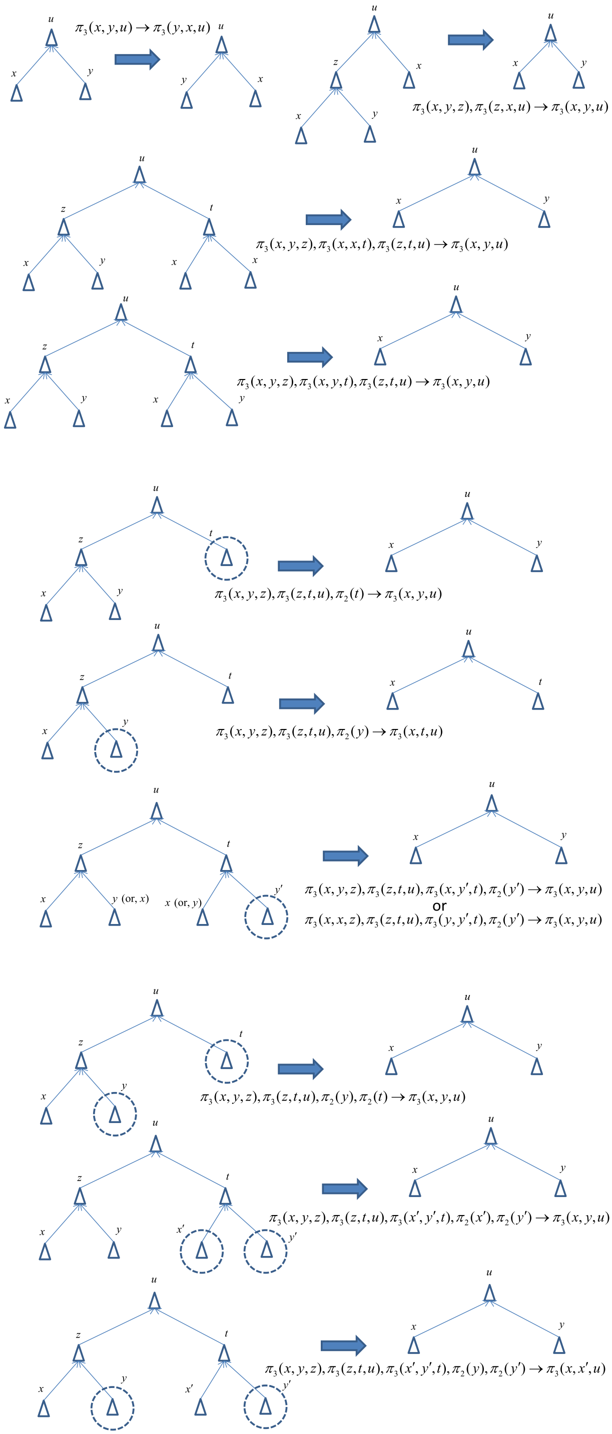


**Fig. 1** A new reduced branch with a parent $u$ and 2 leaves $x$, $y$ (or, $x$, $t$) corresponds to a triple $(x, y, u) \in r^L$. There is no need to list cases with 3 nodes labeled by $O$, because they all are subcases of the listed

Finally, let us prove that $Z' = Z^L$. First, let us prove $Z' = \{v \in V | o_\Sigma(\{v\} \cup O) \cap Z \neq \emptyset\}$. Indeed, if $a \in V$ is such that $o_\Sigma(\{a\} \cup O) \cap Z \neq \emptyset$, then the set $\{h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) | h(a) = 1\}$ is empty. Therefore, $h(a) = 0$ for any $h \in \mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$, which implies $a \in Z'$. On the contrary, if $a \in V$ is such that $o_\Sigma(\{a\} \cup O) \cap Z = \emptyset$, then $h : V \to D$ defined by $h(v) = 1$ if $v \in o_\Sigma(\{a\} \cup O)$ and $h(v) = 0$, if otherwise, is a homomorphism from $\mathbf{R}$ to $\mathbf{\Gamma}$. Therefore, $a \notin Z'$.

Thus, $Z'$ is a set of all elements $a \in V$ such that some element $r \in Z$ can be derived from $\{a\} \cup O$ in the implicational system $\Sigma$. Analogously to the previous case, there is a rooted binary tree $T$ with a root $r \in Z$ whose nodes are labeled by elements of $V$ and leaves are labeled by $\{a\} \cup O$. Using the same technique this tree can be reduced to a root $r$ with two children $c_1$ and $c_2$, such that $\{c_1, c_2\} \subseteq \{a\} \cup O$, $\{c_1, c_2\} \nsubseteq O$ and $(c_1, c_2, r) \in r^L$. W.l.o.g. let $c_1 = a$. If $c_2 \in O$, then using the rule (4) we can deduce $a \in Z^L$. If $c_2 = a$, then using the rule (5) we can deduce $a \in Z^L$. Thus, $Z' \subseteq Z^L$, and consequently, $Z' = Z^L$.

In the case $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$, it is easy to see that we will eventually apply the rule (6).

The complete list of Horn formulas in $L$ is given below:

(1) $\forall x, y, u \ \big(\pi_3(x, y, u) \to \pi_3(y, x, u)\big)$
(2) $\forall x, y, z, u \ \big(\pi_3(x, y, u) \wedge \pi_2(x) \to \pi_3(z, y, u)\big)$
(3) $\forall x, y, z, u \ \big(\pi_3(x, y, u) \wedge \pi_2(x) \wedge \pi_2(y) \to \pi_2(u)\big)$
(4) $\forall x, y, z, u \ \big(\pi_3(x, y, u) \wedge \pi_2(x) \wedge \pi_1(u) \to \pi_1(y)\big)$
(5) $\forall x, y \ \big(\pi_3(x, x, y) \wedge \pi_1(y) \to \pi_1(x)\big)$
(6) $\forall x \ \big(\pi_1(x) \wedge \pi_2(x) \to \mathrm{F}\big)$
(7) $\forall x, y, z, u\big(\pi_3(x, y, z) \wedge \pi_3(z, x, u) \to \pi_3(x, y, u)\big)$
(8) $\forall x, y, z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(x, x, t) \wedge \pi_3(z, t, u) \to \pi_3(x, y, u)\big)$
(9) $\forall x, y, z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(x, y, t) \wedge \pi_3(z, t, u) \to \pi_3(x, y, u)\big)$
(10) $\forall x, y, z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_2(t) \to \pi_3(x, t, u)\big)$
(11) $\forall x, y, z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_2(y) \to \pi_3(x, y, u)\big)$
(12) $\forall x, y, y', z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_3(x, y', t) \wedge \pi_2(y') \to \pi_3(x, y, u)\big)$
(13) $\forall x, y, y', z, t, u\big(\pi_3(x, x, z) \wedge \pi_3(z, t, u) \wedge \pi_3(y, y', t) \wedge \pi_2(y') \to \pi_3(x, y, u)\big)$
(14) $\forall x, y, z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_2(y) \wedge \pi_2(t) \to \pi_3(x, y, u)\big)$
(15) $\forall x, y, x', y', z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_3(x', y', t) \wedge \pi_2(x') \wedge \pi_2(y') \to \pi_3(x, y, u)\big)$
(16) $\forall x, y, x', y', z, t, u\big(\pi_3(x, y, z) \wedge \pi_3(z, t, u) \wedge \pi_3(x', y', t) \wedge \pi_2(y) \wedge \pi_2(y') \to \pi_3(x, x', u)\big)$

This list is not optimized and some formulas could be derivable from others.

## 16 Proof of Theorem 17

Throughout the proof we assume $D = \{0, 1\}$ and $\mathbf{\Gamma} = (D, \varrho_1, \varrho_2, \varrho_3)$ where $\varrho_1 = \big\{(x, y) | x \vee y\big\}$, $\varrho_2 = \big\{(x, y) | \neg x \vee y\big\}$ and $\varrho_3 = \big\{(x, y) | \neg x \vee \neg y\big\}$. For $\rho_1, \rho_2 \subseteq D^2$ let us denote

$$\rho_1 \circ \rho_2 = \{(x, z) | \exists y : (x, y) \in \rho_1 \text{ and } (y, z) \in \rho_2\}$$

**Definition 18** Let $\Gamma_2$ be a set of all nonempty binary relations over $D$. A subset $C \subseteq \mathcal{C}_V^{\mathbf{\Gamma}_2}$ is called full if for any $u, v \in V$ there exists only one $\langle (u, v), \rho\rangle \in C$. A full subset $C \subseteq \mathcal{C}_V^{\mathbf{\Gamma}_2}$ is called path-consistent if for any $\langle (u, v), \rho_1\rangle, \langle (v, w), \rho_2\rangle, \langle (u, w), \rho_3\rangle \in C$ we have $\rho_3 \subseteq \rho_1 \circ \rho_2$ and for any $\langle (u, u), \rho\rangle \in C$ we have $\rho \subseteq \{(a, a) | a \in D\}$.

It is well-known that for binary constraint satisfaction problems, path consistency is equivalent to 3-local consistency [63]. Therefore, if $C \subseteq \mathcal{C}_V^{\mathbf{\Gamma}_2}$ is path-consistent, then the corresponding 2-SAT instance is satisfiable.

Let us introduce the set of formulas:

1. $\forall x$ True $\to \pi_2(x, x)$
2. $\forall x, y \big(\pi_1(x, y) \to \pi_1(y, x)\big)$
3. $\forall x, y \big(\pi_3(x, y) \to \pi_3(y, x)\big)$
4. $\forall x, y, z \big(\pi_2(x, y) \wedge \pi_2(y, z) \to \pi_2(x, z)\big)$
5. $\forall x, y, z \big(\pi_1(x, y) \wedge \pi_2(y, z) \to \pi_1(x, z)\big)$
6. $\forall x, y, z \big(\pi_3(x, y) \wedge \pi_2(z, y) \to \pi_3(x, z)\big)$
7. $\forall x, y, z \big(\pi_3(x, y) \wedge \pi_1(y, z) \to \pi_2(x, z)\big)$

To any relational structure $\mathbf{R} = (V, r_1, r_2, r_3)$, where $r_i, i \in [r]$ is a binary relation, one can correspond the full subset:

$$C(\mathbf{R}) = \{\langle (u, v), \rho_{uv}\rangle | u, v \in V\} \subseteq \mathcal{C}_V^{\mathbf{\Gamma}_2}$$

where

$$\rho_{uv} = \bigcap_{i:(u,v)\in r_i} \varrho_i \bigcap_{i:(u,v)\in r_i^T} \varrho_i^T, \text{ if } u \neq v$$
$$\rho_{uu} = \bigcap_{i:(u,u)\in r_i} \varrho_i \bigcap_{i:(u,u)\in r_i^T} \varrho_i^T \cap \{(a, a)|a \in D\}$$

**Lemma 8** *If* $\mathbf{R} = (V, r_1, r_2, r_3)$ *satisfies the formulas 1-7 and* $r_1 \cap r_2 \cap r_3 \cap r_2^T = \emptyset$, $r_1 \cap r_3 \cap \{(u, u)|u \in V\} = \emptyset$, *then* $C(\mathbf{R})$ *is path-consistent.*

*Proof* Properties 2 and 3 claim that $r_1$ and $r_3$ are symmetric relations, therefore we have $r_1 = r_1^T$ and $r_3 = r_3^T$. Since $r_1 \cap r_2 \cap r_3 \cap r_2^T = \emptyset$, then the set $\{\varrho_i|(u, v) \in r_i\} \cup \{\varrho_i^T|(u, v) \in r_i^T\} \neq \{\varrho_1, \varrho_2, \varrho_3, \varrho_2^T\}$ for any $(u, v)$. Since $\bigcap_{a\in A} a \neq \emptyset$ for any proper subset $A \subset \{\varrho_1, \varrho_2, \varrho_3, \varrho_2^T\}$, then $\rho_{uv} \neq \emptyset$ for any $u \neq v$.

Due to the property 1, we have $(u, u) \in r_2 \cap r_2^T$ for any $u \in V$. Also, $(u, u) \notin r_1 \cap r_3$ because of $r_1 \cap r_3 \cap \{(v, v)|v \in V\} = \emptyset$. Therefore, for any $u \in V$, the set $\{\varrho_i|(u, u) \in r_i\} \cup \{\varrho_i^T|(u, u) \in r_i^T\}$ is a proper subset of $\{\varrho_1, \varrho_3\}$. Thus, $\rho_{uu} \neq \emptyset$ and $\rho_{uu} \subseteq \{(a, a)|a \in D\}$.

Note that for any $u \neq v$: a) $(0, 0) \notin \rho_{uv}$ if and only if $(u, v) \in r_1$, b) $(1, 1) \notin \rho_{uv}$ if and only if $(u, v) \in r_3$, c) $(1, 0) \notin \rho_{uv}$ if and only if $(u, v) \in r_2$, and d) $(0, 1) \notin \rho_{uv}$ if and only if $(v, u) \in r_2$.

Let us prove that $\rho_{uw} \subseteq \rho_{uv} \circ \rho_{vw}$ for any $u, v, w \in V$. Let us first consider the case of distinct $u, v, w$. Let $(a, c) \in \rho_{uw}$. Our goal is to show that there exists $b$ such that $(a, b) \in \rho_{uv}$ and $(b, c) \in \rho_{vw}$. Let us prove the last statement by reductio ad absurdum. Assume that for any $b$ we have $(a, b) \notin \rho_{uv}$, $(b, c) \notin \rho_{vw}$ and $(a, c) \in \rho_{uw}$.

There are 4 possibilities for $(a, c)$: $(0, 0)$, $(1, 1)$, $(0, 1)$ and $(1, 0)$. Let us list all of them and check that $(a, b) \notin \rho_{uv}$ and $(b, c) \notin \rho_{vw}$ and $(a, c) \in \rho_{uw}$ cannot hold for any $b \in \{0, 1\}$.

The case $(a, c) = (0, 0)$: $(0, b) \notin \rho_{uv}$ and $(b, 0) \notin \rho_{vw}$ for $b \in \{0, 1\}$ implies $(u, v) \in r_1 \cap r_2^T$ and $(v, w) \in r_1 \cap r_2$. Due to the property 5 we have $(u, w) \in r_1$ and this contradicts to $(0, 0) \in \rho_{uw}$.

The case $(a, c) = (1, 1)$: $(1, b) \notin \rho_{uv}$ and $(b, 1) \notin \rho_{vw}$ for $b \in \{0, 1\}$ implies $(u, v) \in r_3 \cap r_2$ and $(v, w) \in r_3 \cap r_2^T$. Due to the property 6 we have $(u, w) \in r_3$ and this contradicts to $(1, 1) \in \rho_{uw}$.

The case $(a, c) = (0, 1)$: $(0, b) \notin \rho_{uv}$ and $(b, 1) \notin \rho_{vw}$ for $b \in \{0, 1\}$ implies $(u, v) \in r_1 \cap r_2^T$ and $(v, w) \in r_3 \cap r_2^T$. Due to the property 4 we have $(w, u) \in r_2$ and this contradicts to $(0, 1) \in \rho_{uw}$.

The case $(a, c) = (1, 0)$: $(1, b) \notin \rho_{uv}$ and $(b, 0) \notin \rho_{vw}$ for $b \in \{0, 1\}$ implies $(u, v) \in r_3 \cap r_2$ and $(v, w) \in r_1 \cap r_2$. Due to the property 4 we have $(u, w) \in r_2$ and this contradicts to $(1, 0) \in \rho_{uw}$.

It remains to check path-consistency property for any triple of variables $u, v, w \in V$ where either $u = w \neq v$ or $u = v \neq w$ (i.e. 2-local consistency). The case $u = v = w$ is trivial.

Let us check the case $u = w \neq v$. Let $(a, a) \in \rho_{uu}$. Let us assume that for any $b \in D$ we have $(a, b) \notin \rho_{uv}$. The case $a = 0$ gives $(0, 0) \in \rho_{uu}$, $(0, 0), (0, 1) \notin \rho_{uv}$, and therefore, $(u, u) \notin r_1$, $(u, v) \in r_1 \cap r_2^T$. From property 5 we conclude $(u, u) \in r_1$ and obtain a contradiction. The case $a = 1$ gives $(1, 1) \in \rho_{uu}$, $(1, 0), (1, 1) \notin \rho_{uv}$, and therefore, $(u, u) \notin r_3$, $(u, v) \in r_3 \cap r_2$. From property 6 we conclude $(u, u) \in r_3$ and obtain a contradiction.

Finally, let us check the case $u = v \neq w$. Let $(a, c) \in \rho_{uw}$ and for any $b \in D$ we have $(a, b) \notin \rho_{uu}, (b, c) \notin \rho_{uw}$.

The case $(a, c) = (0, 0)$ gives $(0, 0) \in \rho_{uw}$, $(0, b) \notin \rho_{uu}, (b, 0) \notin \rho_{uw}$. The last is equivalent to $(u, w) \notin r_1$, $(u, u) \in r_1$, $(u, w) \in r_1 \cap r_2$. From property 5 we conclude $(u, w) \in r_1$ and obtain a contradiction.

The case $(a, c) = (1, 1)$ gives $(1, 1) \in \rho_{uw}$, $(1, b) \notin \rho_{uu}, (b, 1) \notin \rho_{uw}$. The last is equivalent to $(u, w) \notin r_3$, $(u, u) \in r_3$, $(u, w) \in r_3 \cap r_2^T$. From property 6 we conclude $(u, w) \in r_3$ and obtain a contradiction.

The case $(a, c) = (0, 1)$ gives $(0, 1) \in \rho_{uw}$, $(0, b) \notin \rho_{uu}, (b, 1) \notin \rho_{uw}$. The last is equivalent to $(u, w) \notin r_2^T$, $(u, u) \in r_1$, $(u, w) \in r_3 \cap r_2^T$. From property 7 we conclude $(w, u) \in r_2$ and obtain a contradiction.

The case $(a, c) = (1, 0)$ gives $(1, 0) \in \rho_{uw}$, $(1, b) \notin \rho_{uu}, (b, 0) \notin \rho_{uw}$. The last is equivalent to $(u, w) \notin r_2$, $(u, u) \in r_3$, $(u, w) \in r_1 \cap r_2$. From property 7 we conclude $(u, w) \in r_2$ and obtain a contradiction. Thus, the lemma is proved. □

**Corollary 2** *Let $L$ be the set of formulas 1-7 and $L^{stop} = \{\pi_1(x, y) \wedge \pi_2(x, y) \wedge \pi_3(x, y) \wedge \pi_2(y, x) \rightarrow \mathrm{F}, \pi_1(x, x) \wedge \pi_3(x, x) \rightarrow \mathrm{F}\}$. Then,* $\mathrm{Dense}(\mathbf{\Gamma})$ *can be solved by the Datalog program $L \cup L^{stop}$.*

*Proof* Let $\mathbf{R}$ be an instance of $\mathrm{Dense}(\mathbf{\Gamma})$. If $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \emptyset$, then $\mathrm{Hom}(\mathbf{R}^L, \mathbf{\Gamma}) = \emptyset$. By construction, $\mathbf{R}^L$ satisfies properties 1-7. If $r_1^L \cap r_2^L \cap r_3^L \cap (r_2^L)^T = \emptyset$ and $r_1^L \cap r_3^L \cap \{(v, v) | v \in V\} = \emptyset$, then, by Lemma 8, the subset $C(\mathbf{R}^L)$ is path-consistent (and therefore, is satisfiable). The last contradicts to $\mathrm{Hom}(\mathbf{R}^L, \mathbf{\Gamma}) = \emptyset$. Therefore, either $r_1^L \cap r_2^L \cap r_3^L \cap (r_2^L)^T \neq \emptyset$ or $r_1^L \cap r_3^L \cap \{(v, v) | v \in V\} \neq \emptyset$. In that case the Datalog program will identify the emptiness of $\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma})$ by applying the rule $\pi_1(x, y) \wedge \pi_2(x, y) \wedge \pi_3(x, y) \wedge \pi_2(y, x) \rightarrow \mathrm{F}$ to $(u, v) \in r_1^L \cap r_2^L \cap r_3^L \cap (r_2^L)^T$ or the rule $\pi_1(x, x) \wedge \pi_3(x, x) \rightarrow \mathrm{F}$ to $(u, u) \in r_1^L \cap r_3^L \cap \{(v, v) | v \in V\}$.

Let us now consider the case $\mathrm{Hom}(\mathbf{R}^L, \mathbf{\Gamma}) \neq \emptyset$. In that case we have $r_1^L \cap r_2^L \cap r_3^L \cap (r_2^L)^T = \emptyset$, $r_1^L \cap r_3^L \cap \{(v, v) | v \in V\} = \emptyset$ and the subset $C(\mathbf{R}^L)$ is path-consistent. A well-known application of Baker-Pixley theorem to languages with a majority polymorphism [64] gives us that path-consistency (or, 3-consistency) implies global consistency. Thus, any 3-consistent solution can be globally extended, i.e.

$$\mathrm{pr}_{u,v}\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) = \mathrm{pr}_{u,v}\mathrm{Hom}(\mathbf{R}^L, \mathbf{\Gamma}) = \rho_{u,v}$$

for any $\langle (u, v), \rho_{uv} \rangle \in C(\mathbf{R}^L)$. Thus,

$$\bigcap_{h \in \mathrm{Hom}(\mathbf{R},\mathbf{\Gamma})} h^{-1}(\varrho_i) = \{(u, v) | \mathrm{pr}_{u,v}\mathrm{Hom}(\mathbf{R}, \mathbf{\Gamma}) \subseteq \varrho_i\} = \{(u, v) | \rho_{u,v} \subseteq \varrho_i\} \subseteq r_i^L$$

The last implies that $(\mathbf{R}^L, \mathbf{\Gamma})$ is a maximal pair, and this completes the proof. □

## 17 Conclusion and open questions

We studied the size of an implicational system $\Sigma$ corresponding to a densification operator on a set of constraints for different constraint languages. It turns out that only for bounded width languages this size can be bounded by a polynomial of the number of variables. This naturally led us to more efficient algorithms for the densification and the sparsification tasks.

An unresolved issue of the paper is a relationship (equality?) between the following classes of constraint languages: a) core languages with a weak polynomial densification operator, b) core languages of bounded width. Also, the complexity classification of $\mathrm{Dense}(\mathbf{\Gamma})$ for the general domain $D$ is still open.

## References

1. Feder, T., & Vardi, M. Y. (1993). Monotone monadic snp and constraint satisfaction. In *Proceedings of the twenty-fifth annual ACM symposium on theory of computing, ser. STOC '93* (pp. 612–622). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/167088.167245.
2. Bulatov, A. A. (2017). A dichotomy theorem for nonuniform csps. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 319–330).
3. Zhuk, D. (2020). A proof of the csp dichotomy conjecture. *Journal of ACM*, *67*, 5. [Online]. Available: https://doi.org/10.1145/3402029.
4. Khanna, S., Sudan, M., Trevisan, L., & Williamson, D. P. (2001). The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, *30*(6), 1863–1920.
5. Cooper, M. C. (2003). Reduction operations in fuzzy or valued constraint satisfaction. *Fuzzy Sets and Systems*, *134*(3), 311–342. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0165011402001343.
6. Cooper, M., De Givry, S., Sanchez, M., Schiex, T., & Zytnicki, M. (2008). Virtual arc consistency for weighted csp. In *Proceedings of the 23rd national conference on artificial intelligence - volume 1, ser. AAAI'08* (pp. 253–258). AAAI Press.
7. Pesant, G. (2005). Counting solutions of csps: a structural approach. In *IJCAI* (pp. 260–265). [Online]. Available: http://ijcai.org/Proceedings/05/Papers/0624.pdf.
8. Bulatov, A. A., Dyer, M., Goldberg, L. A., Jerrum, M., & Mcquillan, C. (2013). The expressibility of functions on the boolean domain, with applications to counting csps. *Journal of ACM, 60*(5). [Online]. Available: https://doi.org/10.1145/2528401.
9. Bulín, J., Krokhin, A., & Opršal, J. (2019). Algebraic approach to promise constraint satisfaction. In *Proceedings of the 51st annual ACM SIGACT symposium on theory of computing, ser. STOC 2019* (pp. 602–613). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/3313276.3316300.

10. Brakensiek, J., & Guruswami, V. (2018). Promise constraint satisfaction: structure theory and a symmetric boolean dichotomy. In *Proceedings of the twenty-ninth annual ACM-SIAM symposium on discrete algorithms, ser. SODA '18* (pp. 1782–1801). USA: Society for Industrial and Applied Mathematics.
11. Carvalho, C., Martin, B., & Zhuk, D. (2017). The complexity of quantified constraints using the algebraic formulation. In K. G. Larsen, H. L. Bodlaender, & J.-F. Raskin (Eds.) *42nd International symposium on mathematical foundations of computer science (MFCS 2017), ser. Leibniz International Proceedings in Informatics (LIPIcs)*, (Vol. 83 pp. 27:1–27:14). Dagstuhl: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2017/8079.
12. Ferguson, A., & O'Sullivan, B. (2006). Relaxations and explanations for quantified constraint satisfaction problems. In F. Benhamou (Ed.) *Principles and practice of constraint programming - CP 2006* (pp. 690–694). Berlin: Springer.
13. Bauland, M., Böhler, E., Creignou, N., Reith, S., Schnoor, H., & Vollmer, H. (2009). The complexity of problems for quantified constraints. *Theory of Computing Systems*, *47*, 454–490.
14. Deng, J., Ding, N., Jia, Y., Frome, A., Murphy, K., Bengio, S., Li, Y., Neven, H., & Adam, H. (2014). Large-scale object classification using label relation graphs. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.) *Computer Vision – ECCV 2014* (pp. 48–64). Cham: Springer International Publishing.
15. Bishop, C. M. (2006). *Pattern recognition and machine learning (information science and statistics)*. Berlin: Springer.
16. Chen, H., Jansen, B. M. P., & Pieterse, A. (2019). Best-case and worst-case sparsifiability of boolean CSPs. In C. Paul, & M. Pilipczuk (Eds.) *13th International symposium on parameterized and exact computation (IPEC 2018), ser. Leibniz International Proceedings in Informatics (LIPIcs)*, (Vol. 115 pp. 15:1–15:13). Dagstuhl: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2019/10216.
17. Lagerkvist, V., & Wahlström, M. (2020). Sparsification of sat and csp problems via tractable extensions. *ACM Trans. Comput. Theory, 12*(2). [Online]. Available: https://doi.org/10.1145/3389411.
18. Hemaspaandra, E., & Schnoor, H. (2011). Minimization for generalized boolean formulas. In *IJCAI International joint conference on artificial intelligence* (p. 04).
19. Maier, D. (1983). *The theory of relational databases*. Rockville: Computer Science Press.
20. Ausiello, G., D'Atri, A., & Saccà, D. (1984). *Minimal representations of directed hypergraphs and their application to database design*, (pp. 125–157). Vienna: Springer.
21. Boros, E., & Cepek, O. (1994). *On the complexity of horn minimization. Rutcor Research Report 1-94*. New Brunswick: Rutgers Center for Operations Research.
22. Hammer, P. L., & Kogan, A. (1993). Optimal compression of propositional horn knowledge bases: complexity and approximation. *Artificial Intelligence*, *64*(1), 131–145. [Online]. Available: http://www.sciencedirect.com/science/article/pii/000437029390062G.
23. Bhattacharya, A., DasGupta, B., Mubayi, D., & Turán, G. (2010). On approximate horn formula minimization. In S. Abramsky, C. Gavoille, C. Kirchner, F. Meyer auf der Heide, & P. G. Spirakis (Eds.) *Automata, languages and programming* (pp. 438–450). Berlin: Springer.
24. Chang, T. (2006). Horn formula minimization. Master's thesis-6890. Rochester Institute of Technology. Accessed from https://scholarworks.rit.edu/theses/6890/.
25. Caspard, N., & Monjardet, B. (2003). The lattices of closure systems, closure operators, and implicational systems on a finite set: a survey. *Discrete Applied Mathematics*, *127*(2), 241–269. Ordinal and Symbolic Data Analysis (OSDA '98), Univ. of Massachusetts, Amherst, Sept. 28-30, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166218X02002093.
26. Lucchesi, C. L., & Osborn, S. L. (1978). Candidate keys for relations. *Journal of Computer and System Sciences*, *17*(2), 270–279. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0022000078900090.
27. Hardt, M., Srivastava, N., & Tulsiani, M. (2012). Graph densification. In *Proceedings of the 3rd innovations in theoretical computer science conference, ser. ITCS '12* (pp. 380–392). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/2090236.2090266.
28. Escolano, F., Curado, M., Lozano, M. A., & Hancook, E.R. (2016). Dirichlet graph densifiers. In A. Robles-Kelly, M. Loog, B. Biggio, F. Escolano, & R. Wilson (Eds.) *Structural, syntactic and statistical pattern recognition* (pp. 185–195). Cham: Springer International Publishing.
29. Curado, M., Escolano, F., Lozano, M. A., & Hancock, E.R. (2018). Semi-supervised graph rewiring with the dirichlet principle. In *2018 24th International Conference on Pattern Recognition (ICPR)* (pp. 2172–2177).
30. Benczúr, A. A., & Karger, D. R. (1996). Approximating s-t minimum cuts in $\tilde{O}$(n2) time. In *Proceedings of the twenty-eighth annual ACM symposium on theory of computing, ser. STOC '96* (pp. 47–55). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/237814.237827.

31. Batson, J., Spielman, D. A., & Srivastava, N. (2012). Twice-ramanujan sparsifiers. *SIAM Journal on Computing*, *41*(6), 1704–1721.
32. Andoni, A., Chen, J., Krauthgamer, R., Qin, B., Woodruff, D. P., & Zhang, Q. (2016). On sketching quadratic forms. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science, ser. ITCS '16* (pp. 311–319). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/2840728.2840753.
33. Filtser, A., & Krauthgamer, R. (2017). Sparsification of two-variable valued constraint satisfaction problems. *SIAM Journal on Discrete Mathematics*, *31*(2), 1263–1276.
34. Butti, S., & Zivný, S. (2019). Sparsification of Binary CSPs. In R. Niedermeier, & C. Paul (Eds.) *36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019), ser. Leibniz International Proceedings in Informatics (LIPIcs)*, (Vol. 126 pp. 17:1–17:8). Dagstuhl: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2019/10256.
35. Fung, W. S., Hariharan, R., Harvey, N. J., & Panigrahi, D. (2011). A general framework for graph sparsification. In *Proceedings of the forty-third annual ACM symposium on theory of computing, ser. STOC '11* (pp. 71–80). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/1993636.1993647.
36. Soma, T., & Yoshida, Y. (2019). Spectral sparsification of hypergraphs. In *Proceedings of the thirtieth annual ACM-SIAM symposium on discrete algorithms, ser. SODA '19* (pp. 2570–2581). USA: Society for Industrial and Applied Mathematics.
37. Arora, S., Karger, D., & Karpinski, M. (1995). Polynomial time approximation schemes for dense instances of np-hard problems. In *Proceedings of the twenty-seventh annual ACM symposium on theory of computing, ser. STOC '95* (pp. 284–293). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/225058.225140.
38. Arora, S., Frieze, A., & Kaplan, H. (1996). A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. In *Proceedings of 37th conference on foundations of computer science* (pp. 21–30).
39. Frieze, A., & Kannan, R. (1996). The regularity lemma and approximation schemes for dense problems. In *Proceedings of 37th conference on foundations of computer science* (pp. 12–20).
40. Curado, M. (2018). *Structural similarity: Applications to object recognition and clustering*. Ph.D. dissertation: University of Alicante.
41. Bodnarchuk, V., Kaluznin, L., Kotov, V., & Romov, B. (1969). Galois theory for post algebras. *Cybernetics*, *5*(1–2), 243–252, 531–539.
42. Geiger, D. (1968). Closed systems of functions and predicates. *Pac. J. Math.*, *27*(1), 95–100.
43. Takhanov, R. S. (2007). Maximum predicate descriptions of sets of mappings. *Computational Mathematics and Mathematical Physics*, *47*(9), 1570–1581. [Online]. Available: https://doi.org/10.1134/S0965542507090175.
44. Schnoor, H., & Schnoor, I. (2008). *Partial polymorphisms and constraint satisfaction problems*, (pp. 229–254). Berlin: Springer. [Online]. Available: https://doi.org/10.1007/978-3-540-92800-3_9.
45. Böhler, E., Hemaspaandra, E., Reith, S., & Vollmer, H. (2002). Equivalence and isomorphism for boolean constraint satisfaction. In J. Bradfield (Ed.) *Computer science logic* (pp. 412–426). Berlin: Springer.
46. Levene, M., & Loizou, G. (1999). *A guided tour of relational databases and beyond*. Berlin: Springer.
47. Fischer, P. C., Jou, J., & Tsou, D.-M. (1983). Succinctness in dependency systems. *Theoretical Computer Science*, *24*(3), 323–329. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0304397583900075.
48. Larose, B., Valeriote, M., & Zádori, L. (2009). Omitting types, bounded width and the ability to count. *IJAC*, *19*, 647–668, 08.
49. Bulatov, A. A. (2009). Bounded relational width. Simon Fraser University, Tech. Rep.
50. Barto, L., & Kozik, M. (2009). Constraint satisfaction problems of bounded width. In *2009 50th Annual IEEE symposium on foundations of computer science* (pp. 595–603).
51. Beeri, C., & Bernstein, P. A. (1979). Computational problems related to the design of normal form relational schemas. *ACM Transactions on Database Systems*, *4*(1), 30–59. [Online]. Available: https://doi.org/10.1145/320064.320066.
52. Codd, E. F. (1972). Further normalization of the data base relational model. *Data Base Systems*, 33–64. [Online]. Available: https://ci.nii.ac.jp/naid/10003016060/en/.
53. Benito-Picazo, F., Cordero, P., Enciso, M., & Mora, A. (2017). Reducing the search space by closure and simplification paradigms. *The Journal of Supercomputing*, *73*(1), 75–87. [Online]. Available: https://doi.org/10.1007/s11227-016-1622-1.

54. Sridhar, R., & Iyengar, S. S. (1990). Efficient parallel algorithms for functional dependency manipulations. In *[1990] Proceedings second international symposium on databases in parallel and distributed systems* (pp. 126–137).
55. Gottlob, G. (1987). Computing covers for embedded functional dependencies. In *Proceedings of the sixth ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems, ser. PODS '87* (pp. 58–69). New York: Association for Computing Machinery. [Online]. Available: https://doi.org/10.1145/28659.28665.
56. Bodirsky, M., & Dalmau, V. (2013). Datalog and constraint satisfaction with infinite templates. *Journal of Computer and System Sciences*, *79*(1), 79–100. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022000012001213.
57. Larose, B., Loten, C., & Tardif, C. (2006). A characterisation of first-order constraint satisfaction problems. In *21st Annual IEEE Symposium on Logic in Computer Science (LICS'06)* (pp. 201–210).
58. Marchenkov, S. (2000). *Closed classes of boolean functions*. Nauka: Fizmatlit.
59. Marchenkov, S. (1998). The invariants of post classes. *Fundam. Prikl. Mat.*, *4*(4), 1385–1404. [Online]. Available: http://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=fpm&paperid=360&option_lang=eng.
60. Böhler, E., Reith, S., Schnoor, H., & Vollmer, H. (2005). Bases for boolean co-clones. *Information Processing Letters*, *96*(2), 59–66.
61. Horn, A. (1951). On sentences which are true of direct unions of algebras. *The Journal of Symbolic Logic*, *16*(1), 14–21. [Online]. Available: http://www.jstor.org/stable/2268661.
62. McKinsey, J. C. C. (1943). The decision problem for some classes of sentences without quantifiers. *The Journal of Symbolic Logic*, *8*(2), 61–76. [Online]. Available: http://www.jstor.org/stable/2268172.
63. Dechter, R. (2003). *Constraint processing*. San Francisco: Morgan Kaufmann Publishers Inc.
64. Jeavons, P., Cohen, D., & Cooper, M.C. (1998). Constraints, consistency and closure. *Artificial Intelligence*, *101*(1), 251–265. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370298000228.