Blowing Holes in Various Aspects of Computational Problems, with Applications to Constraint Satisfaction

Peter Jonsson, Victor Lagerkvist, and Gustav Nordh

Department of Computer and Information Science, Linköping University, Sweden
{peter.jonsson, victor.lagerkvist, gustav.nordh}@liu.se

Abstract. We consider methods for constructing NP-intermediate problems under the assumption that P $\neq$ NP. We generalize Ladner's original method for obtaining NP-intermediate problems by using parameters with various characteristics. In particular, this generalization allows us to obtain new insights concerning the complexity of CSP problems. We begin by fully characterizing the problems that admit NP-intermediate subproblems for a broad and natural class of parameterizations, and extend the result further such that structural CSP restrictions based on parameters that are hard to compute (such as tree-width) are covered. Hereby we generalize a result by Grohe on width parameters and NP-intermediate problems. For studying certain classes of problems, including CSPs parameterized by constraint languages, we consider more powerful parameterizations. First, we identify a new method for obtaining constraint languages Γ such that CSP(Γ) are NP-intermediate. The sets Γ can have very different properties compared to previous constructions (by, for instance, Bodirsky & Grohe) and provides insights into the algebraic approach for studying the complexity of infinite-domain CSPs. Second, we prove that the propositional abduction problem parameterized by constraint languages admits NP-intermediate problems. This settles an open question posed by Nordh & Zanuttini.

1 Introduction

Ladner [20] explicitly constructed NP-intermediate problems (under the assumption P $\neq$ NP) by removing strings of certain lengths from NP-complete languages via a diagonalization technique that is colloquially known as *blowing holes in problems*. The languages constructed via blowing are unfortunately famous for being highly artificial: Arora and Barak [1] write the following.

> We do not know of a natural decision problem that, assuming NP $\neq$ P, is proven to be in NP $\setminus$ P but not NP-complete, and there are remarkably few candidates for such languages

More natural examples are known under other complexity-theoretic assumptions. For instance, LogClique (the problem of deciding whether an n-vertex

graph contains a clique of size $\log n$) is NP-intermediate under the exponential-time hypothesis (ETH). The lack of natural NP-intermediate computational problems makes it important to investigate new classes of NP-intermediate problems and, hopefully, increase our understanding of the borderline between P and NP.

We begin (in Section 3) by presenting a diagonalization method for obtaining NP-intermediate problems, based on parameterizing decision problems in different ways. In our framework, a parameter, or a *measure function*, is simply a function ρ from the instances of some decision problem X to the non-empty subsets of $\mathbb{N}$. We say that such a function is *single-valued* if $\rho(I)$ is a singleton set for every instance of X, and *multi-valued* otherwise. Depending on the parameter one obtains problems with different characteristics. Simple applications of our method include the connection between the complexity class XP and NP-intermediate problems observed by Chen et al. [9]. Even though our method is still based on diagonalization we claim that the intermediate problems obtained are qualitatively different from the ones obtained by Ladner's original method, and that they can be used for gaining new insights into the complexity of computational problems. We demonstrate this on different CSP problems in the following sections.

In Section 4, we analyze the applicability of the diagonalization method for single-valued measure functions. Under mild additional assumptions, we obtain a full understanding of when NP-intermediate problems arise when the measure function is single-valued and polynomial-time computable. Unfortunately, CSPs under structural restrictions (i.e. when considering instances with bounded width parameters) are not captured by this result since width parameters are typically not polynomial-time computable. To remedy this, we present a fairly general method for obtaining NP-intermediate problems based on structurally restricted CSPs in Section 4.2. This is a generalization of a result by Grohe [15] who has shown that, under the assumption that FPT $\neq$ W[1], NP-intermediate CSP problems can be obtained by restricting the tree-width of their corresponding primal graphs. Our result imply that this holds also under the weaker assumption that P $\neq$ NP and for many different width parameters. NP-intermediate problems based on structural restrictions have also been identified by Bodirsky & Grohe [4].

Multi-valued measure functions are apparently much harder to study and a full understanding appears difficult to obtain. Despite this, multi-valued measure functions have highly useful properties and we exploit them for studying constraint satisfaction problems parameterized by constraint languages. Our first result is inspired by Bodirsky & Grohe [4] who have proved that there exists an infinite constraint language Γ such that CSP(Γ) is NP-intermediate. We extend this and prove that whenever an infinite language Γ does not satisfy the so called *local-global* property, i.e. when CSP(Γ) $\notin$ P but CSP(Γ') $\in$ P for all finite $\Gamma' \subset \Gamma$, then there exists a language closely related to Γ such that the resulting CSP problem is NP-intermediate. The only requirement is that Γ can be extended by certain operators $\langle\cdot\rangle$. We then provide two very different ex-

tension operators. The first operator $\langle\cdot\rangle_{pow}$ works for languages over both finite and infinite domains but gives relations of arbitrarily high arity. The second operator $\langle\cdot\rangle_{+}$ is limited to idempotent languages over infinite domains but does have the advantage that the arity of any relation is only increased by a small constant factor. Together with the language Γ° from Jonsson & Lööw [18] which does not satisfy the local-global property we are thus able to identify a concrete language $\langle\Gamma^{\circ}\rangle_{+}$ such that $\mathrm{CSP}(\langle\Gamma^{\circ}\rangle_{+})$ is NP-complete, $\mathrm{CSP}(\Gamma') \in \mathrm{P}$ for any finite $\Gamma' \subset \langle\Gamma^{\circ}\rangle_{+}$, and there exists a $\Gamma'' \subset \langle\Gamma^{\circ}\rangle_{+}$ such that $\mathrm{CSP}(\Gamma'')$ is NP-intermediate. The so-called *algebraic approach* [3, 6] has been very successful in studying the computational complexity of both finite- and infinite-domain CSPs. However, this approach is, to a large extent, limited to constraint languages that are finite. If one only considers tractable finite subsets of $\langle\Gamma^{\circ}\rangle_{+}$, we miss that there are both NP-intermediate and NP-complete problems within $\mathrm{CSP}(\langle\Gamma^{\circ}\rangle_{+})$. Hence the constraint language $\langle\Gamma^{\circ}\rangle_{+}$ clearly shows the algebraic approach in its present shape is not able to give a full understanding of $\mathrm{CSP}(\langle\Gamma^{\circ}\rangle_{+})$ and its subclasses.

Our second result (which is presented in Section 5.3) is the propositional *abduction* problem $\textsc{Abd}(\Gamma)$. This problem can be viewed as a non-monotonic extension of propositional logic and it has numerous important applications ranging from automated diagnosis, text interpretation to planning. The complexity of propositional abduction has been intensively studied from a complexity-theoretic point of view (cf. [13, 23]) and the computational complexity is known for every finite Boolean constraint language Γ and many infinite languages [23]. In Nordh & Zanuttini [23], the question of whether such a classification is possible to obtain for infinite languages was left open. Since the abduction problem can loosely be described as a combination of the SAT and UNSAT problems, it might be expected that it, like the parameterized $\mathrm{SAT}(\cdot)$ problem, does not contain any NP-intermediate problems. By exploiting our diagonalization method, we present a constraint language Γ such that $\textsc{Abd}(\Gamma)$ is NP-intermediate.

2 Preliminaries

Let Γ denote a (possibly infinite) set of finitary relations over some (possibly infinite) set D. We call Γ a *constraint language*. Given a relation $R \subseteq D^k$, we let $ar(R) = k$. The reader should note that we will sometimes express Boolean relations as conjunctions of Boolean clauses. The *constraint satisfaction problem* over Γ (abbreviated as $\mathrm{CSP}(\Gamma)$) is defined as follows.

INSTANCE: A set V of variables and a set C of constraint applications $R(v_1, \ldots, v_k)$ where $R \in \Gamma$, $k = ar(R)$, and $v_1, \ldots, v_k \in V$.
QUESTION: Is there a total function $f : V \to D$ such that $(f(v_1), \ldots, f(v_k)) \in R$ for each constraint $R(v_1, \ldots, v_k)$ in C?

For an arbitrary decision problem X, we let $I(X)$ denote its set of instances, and $||I||$ to denote the number of bits needed for representing $I \in I(X)$. By a *polynomial-time reduction* from problem X to problem X', we mean a Turing reduction from X to X' that runs in time $O(p(||I||))$ for some polynomial p.

Definition 1. *Let X be a decision problem. A total and computable function $\rho : I(X) \to 2^{\mathbb{N}} \setminus \{\emptyset\}$ is said to be a* measure function.

If $\rho(I)$ is a singleton set for every $I \in I(X)$, then we say that ρ is *single-valued*, and otherwise that it is *multi-valued*. We abuse notation in the first case and simply assume that $\rho : I(X) \to \mathbb{N}$. The measure function ρ combined with a decision problem X yields a problem $X_\rho(S)$ parameterized by $S \subseteq \mathbb{N}$.

INSTANCE. Instance I of X such that $\rho(I) \subseteq S$.
QUESTION. Is I a yes-instance?

For examples of both single- and multi-valued measure functions we refer the reader to Section 3.2. Finally, we prove a simple lemma regarding single-valued measure functions that will be important later on.

Lemma 2. *Let ρ be a single-valued and polynomial-time computable measure function. Let $S \subseteq \mathbb{N}$ and let T be a non-empty subset of S such that $S \setminus T = \{s_1, \ldots, s_k\}$. If $X_\rho(\{s_i\})$, $1 \leq i \leq k$, is in P, then there is a polynomial-time reduction from $X_\rho(S)$ to $X_\rho(T)$.*

Proof. Let I be an arbitrary instance of $X_\rho(S)$. Compute (in polynomial time) $\rho(I)$. If $\rho(I) \in \{s_1, \ldots, s_k\}$, then we can compute the correct answer in polynomial time. Otherwise, I is an instance of $X_\rho(T)$ and the reduction is trivial. □

3 Generation of NP-intermediate Problems

We will now extend Ladner's method to parameterized problems. Section 3.1 contains the main result and Section 3.2 contains some examples.

3.1 Diagonalization Method

Theorem 3. *Let $X_\rho(\cdot)$ be a computational decision problem with a measure function ρ. Assume that $X_\rho(\cdot)$ and $S \subseteq \mathbb{N}$ satisfies the following properties:*

P0: $I(X)$ is recursively enumerable.
P1: $X_\rho(S)$ is NP-complete.
P2: $X_\rho(T)$ is in P whenever T is a finite subset of S.
P3: $X_\rho(S)$ is polynomial-time reducible to $X_\rho(T)$ whenever $T \subseteq S$ and $S \setminus T$ is finite.

Then, if P $\neq$ NP*, there exists a set $S' \subset S$ such that $X_\rho(S')$ is in* NP $\setminus$ P *and $X_\rho(S)$ is not polynomial-time reducible to $X_\rho(S')$.*

Before the proof, we make some observations that will be used without explicit references. If ρ is single-valued and polynomial-time computable, then P2 implies P3 by Lemma 2. In many examples, $S = \mathbb{N}$ which means that P1 can be restated as NP-completeness of X. If P1 holds, then property P3 simply states that $X_\rho(T)$ is NP-complete for every cofinite $T \subseteq S$. Finally, we remind the reader that the polynomial-time bounds may depend on the choice of S in the definitions of P2 and P3.

The proof is an adaption of Papadimitriou's [24] proof where we use the abstract properties P0 – P3 instead of focusing on the size of instances. Papadimitriou's proof is, in turn, based on Ladner's original proof [20]. It may also be illuminating to compare with Schöning [25] and Bodirsky & Grohe [4].

In the sequel, we let $X_\rho(\cdot)$ be a computational decision problem that together with $S \subseteq \mathbb{N}$ satisfies properties P0 – P3. Let A_X be an algorithm for $X_\rho(S)$, let $M_1, M_2, \ldots$ be an enumeration of all polynomial-time bounded deterministic Turing machines, and let $R_1, R_2, \ldots$ be an enumeration of all polynomial-time Turing reductions. Such enumerations are known to exist, cf. Papadimitriou [24].

We define a function $f : \mathbb{N} \to \mathbb{N}$ that is computed by a Turing machine F and the input n is given to F in unary representation. We let $f(0) = f(1) = 0$. The computation of $f(n)$ starts with the computation of $f(0), f(1), f(2), \ldots$, until the total number of steps F has used in computing this sequence exceeds n. This is possible since F has access to its own description by Kleene's fixed point theorem. Let i be the largest value for which F was able to completely compute $f(i)$ (during these n steps) and let $k = f(i)$.

In the final phase of the execution of the machine F we have two cases depending on whether k is even or odd. In both cases, if this phase requires F to run for more than n computation steps, F stops and returns k (i.e., $f(n) = k$).

The first case is when k is even: here, F enumerates all instances I of $X_\rho(S)$ — this is possible by property P0. For each instance I, F simulates $M_{k/2}$ on the encoding of I, determines whether $A_X(I)$ is accepted, and finally, F computes f for all $x \in \rho(I)$. If $M_{k/2}$ rejects and $A_X(I)$ was accepted, and $f(x)$ is even for all $x \in \rho(I)$, then F returns $k + 1$ (i.e., $f(n) = k + 1$). F also returns $k + 1$ if $M_{k/2}$ accepts and I is not accepted by A_X and $f(x)$ is even for all $x \in \rho(I)$.

The second case is when k is odd. Again, F enumerates all instances I of $X_\rho(S)$. Let $E = \emptyset$. Now, for each instance I, F begins simulating $R_{\lfloor k/2 \rfloor}$ on the encoding of I with an oracle for A_X. Whenever the simulation notices that $R_{\lfloor k/2 \rfloor}$ enters an oracle state, we calculate $\rho(I') = E'$ (where I' is the $X_\rho(S)$ instance corresponding to the input of the oracle tape), and add the members of E' to E. When the simulation is finished we first calculate $f(x)$ for every $x \in E$. If the result of any $f(x)$ operation is odd we return $k + 1$. We then compare the result of the reduction with $A_X(I)$. If the results do not match, i.e. if one is accepted or rejected while the other is not, we return $k + 1$. This completes the definition of f. Note that f can be computed in polynomial time (regardless of the time complexity of computing ρ and A_X) since the input is given in unary.

We now show that f is increasing, i.e. for all $n \geq 0$, $f(n) \leq f(n + 1)$ and $\{f(n) \mid n \in \mathbb{N}\}$ is an unbounded set, unless P = NP. To see this, we first prove by induction that $f(n) \leq f(n + 1)$ for all $n \geq 0$. This obviously holds for $n = 0$ and $n = 1$. Assume that this holds for an arbitrary number $i > 1$. By definition $f(i + 1)$ cannot return a smaller number than $f(i)$ in the first phase of the computation, since the Turing machine F simulates $f(i')$ for all $i' < i$, and returns the largest k for which $f(i')$ was successfully computed within the allotted time. In the second phase, the argument to f is used to determine the

total amount of computation steps, and since f will either return the k from the first phase, or $k+1$, there is no possibility that $f(i) > f(i+1)$.

Let $S_e = \{x \mid x \in S \text{ and } f(x) \text{ is even}\}$. We continue by showing that there is no n_0 such that $f(n) = k_0$ for all $n > n_0$ unless P = NP. If there is such a n_0, then there is also a n_1 such that for all $n > n_1$ the value k computed in the phase where F computes $f(1), f(2), \ldots$ (in n steps) is k_0. If k_0 is even, then on all inputs $n > n_1$ the machine $M_{k_0/2}$ correctly decides $X_\rho(S_e)$ and thus $X_\rho(S_e)$ is in P. But since $f(n) = k_0$ for all $n > n_1$, we have that $S \setminus S_e$ is finite, and thus $X_\rho(S)$ is polynomial-time reducible to $X_\rho(S_e)$ by Property P3, which is a contradiction since $X_\rho(S)$ is NP-complete by Property P1. Similarly if k_0 is odd, then on all inputs $n > n_1$ the function $R_{\lfloor k_0/2 \rfloor}$ is a valid reduction from $X_\rho(S)$ to $X_\rho(S_e)$ and thus $X_\rho(S_e) \notin$ P. But since $f(n) = k_0$ for all $n > n_1$, we have that S_e is finite, and we conclude that $X_\rho(S_e)$ is in P by Property P2, which is a contradiction since $X_\rho(S)$ is NP-complete by Property P1.

We conclude the proof by showing that $X_\rho(S_e)$ is neither in P, nor is $X_\rho(S)$ polynomial-time reducible to $X_\rho(S_e)$, unless P = NP. By Property P1, $X_\rho(S_e)$ is in NP since $S_e \subseteq S$. Assume now that $X_\rho(S_e)$ is in P. Then there is an i such that M_i solves $X_\rho(S_e)$. Thus, by the definition of f, there is an n_1 such that for all $n > n_1$ we have $f(n) = 2i$; this contradicts that f is increasing. Similarly, assume that $X_\rho(S)$ is polynomial-time reducible to $X_\rho(S_e)$. Then, there is an i such that R_i is a polynomial-time reduction from $X_\rho(S)$ to $X_\rho(S_e)$. It follows from the definition of f that there is an n_1 such that $f(n) = 2i-1$ for all $n > n_1$, and this contradicts that f is increasing. □

If the measure function is polynomially bounded (e.g. $\rho(I) \leq p(||I||)$ for some polynomial p), then checking whether an integer x written in binary is in S_e or not can be decided in polynomial time. This follows from the fact that x written in binary can be converted to x written in unary in polynomial time. Another useful observation is the following: it follows from the proof that property P1 (i.e. the NP-hardness of the original problem) can be replaced by hardness for other complexity classes within NP. By noting that $X_\rho(S_e)$ is recursively enumerable, this implies that we can construct infinite chains of problems $X_\rho(T_1), X_\rho(T_2), \ldots$ such that $S_e = T_1 \supset T_2 \supset \ldots$, there does not exist any polynomial-time reductions from $X_\rho(T_i)$ to $X_\rho(T_{i+1})$, and $X_\rho(T_i)$ is not in P for any $i \geq 1$.

3.2 Examples

Ladner's result is now a straightforward consequence of Theorem 3. Let X be an arbitrary NP-complete problem such that $I(X)$ is recursively enumerable. For an arbitrary instance $I \in I(X)$, we let the single-valued measure function ρ be defined such that $\rho(I) = ||I||$. We verify that $X_\rho(\mathbb{N})$ satisfies properties P0 – P3 and conclude that there exists a set $T \subseteq \mathbb{N}$ such that $X_\rho(T)$ is NP-intermediate. Properties P0 and P1 hold by assumption and property P2 holds since $X_\rho(U)$ can be solved in constant time whenever U is finite. If $U \subseteq \mathbb{N}$ and $\mathbb{N} \setminus U = \{x_1, \ldots, x_k\}$, then $X_\rho(\{x_i\})$, $1 \leq i \leq k$, is solvable in constant time and we can apply Lemma 2(2). Thus, property P3 holds, too.

Another straightforward application of single-valued measure functions is the following: Chen et al. [9] have discovered a striking connection between NP-intermediate problems and the parameterized complexity class XP (XP denotes the class of decision problems X that are solvable in time $O(||I||^{f(k)})$ for some polynomial-time computable parameter k and some computable function f).

Proposition 4. *Let X be a decision problem and ρ a polynomial-time computable single-valued measure function such that $X_\rho(\cdot)$ satisfies conditions P0 and P1, and $X_\rho \in$ XP. Then there exists a $T \subseteq \mathbb{N}$ such that $X_\rho(T)$ is NP-intermediate.*

Proof. We note that $X_\rho(S)$ is in P whenever S is a finite subset of $\mathbb{N}$. Hence, X_ρ satisfies P2 and consequently P3. The result follows from Theorem 3. □

To illustrate multi-valued measure functions, we turn our attention to the Subset-Sum problem [19].

Instance: A finite set $Y \subseteq \mathbb{N}$ and a number $k \in \mathbb{N}$.
Question: Is there a $Y' \subseteq Y$ such that $\sum Y' = k$?

We define a multi-valued measure function by letting $\rho((Y,k)) = Y$. Once again, properties P0 and P1 hold by assumption so it is sufficient to prove that Subset-Sum$_\rho(\mathbb{N})$ satisfies P2 and P3. Property P2: instances of Subset-Sum can be solved in time $O(\text{poly}(||I||) \cdot c(I))$, where $c(I)$ denotes the difference between the largest and smallest number in Y [14]. This difference is finite whenever we consider instances of Subset-Sum$_\rho(S)$ where $S \subseteq \mathbb{N}$ is finite. Property P3: arbitrarily choose $S \subseteq \mathbb{N}$ such that that $\mathbb{N} \setminus S$ is finite. We present a polynomial-time Turing reduction from Subset-Sum$_\rho(\mathbb{N})$ to Subset-Sum$_\rho(S)$. Let $I = (Y,k)$ be an instance of Subset-Sum$_\rho(\mathbb{N})$. Let $T = Y \setminus S$, i.e. the elements of the instance which are not members of the smaller set S. Since $\mathbb{N} \setminus S$ is finite, T is a finite set, too. Let $Z = Y \cap S$. For every subset $T'_i = \{x_1, \ldots, x_{im}\}$ of T, we let $I'_i = (Z, k'_i)$, where $k'_i = k - (x_1 + \ldots + x_{im})$. Then, it is easy to see that I is a yes-instance if and only if at least one I'_i is a yes-instance. Finally, we note that the reduction runs in time $O(\text{poly}(||I||) \cdot 2^c)$, where $c = |\mathbb{N} \setminus S|$, and this is consequently a polynomial-time reduction for every fixed S.

4 Single-Valued Measure Functions

This section is divided into two parts: Section 4.1 is concerned with polynomial-time computable single-valued measure functions and Section 4.2 is concerned with structurally restricted CSPs.

4.1 Polynomial-Time Computable Measure Functions

By Theorem 3, we know that properties P0 – P3 are sufficient to assure the existence of NP-intermediate problems. A related question is to what degree the properties are also necessary. Here, we investigate the scenario when P2 and P3 do not necessarily hold.

Theorem 5. *Assume X is a decision problem and ρ is a single-valued measure function such that $X_\rho(\mathbb{N})$ satisfies P0 and P1. Let $S_P = \{s \in \mathbb{N} \mid X_\rho(\{s\}) \in \mathrm{P}\}$ and assume membership in S_P is a decidable problem. Then, at least one of the following holds: (1) there exists a set $T \subseteq S_P$ such that $X_\rho(T)$ is NP-intermediate, (2) there exists a $t \in \mathbb{N}$ such that $X_\rho(\{t\})$ is NP-intermediate, or (3) X_ρ admits no NP-intermediate subproblems.*

Proof. If $X_\rho(\{s\})$ is NP-complete for every $s \in \mathbb{N}$, then we are in case (3) so we assume this is not the case. If there exists $s \in \mathbb{N}$ such that $X_\rho(\{s\})$ is NP-intermediate, then we are in case (2) so we assume this does not hold either. Thus, we may henceforth assume that there exists $s \in \mathbb{N}$ such that $X_\rho(\{s\}) \in P$ and that $X_\rho(\{u\})$ is NP-complete whenever $u \in \mathbb{N} \setminus S_P$. This implies that S_P is non-empty. Once again, we single out two straightforward cases: if $X_\rho(S_P)$ is NP-intermediate, then we are in case (1), and if $X_\rho(S_P)$ is in P, then we are in case (3) (since $X_\rho(\{u\})$ is NP-complete whenever $u \notin S_P$). Hence, we may assume that $X_\rho(S_P)$ is NP-complete (note that $X_\rho(S_P) \in \mathrm{NP}$ since $X_\rho(\mathbb{N}) \in \mathrm{NP}$ by P1), i.e. $X_\rho(S_P)$ satisfies P1. Furthermore, $X_\rho(S_P)$ satisfies P0 since S_P is a decidable set and the instances of X are recursively enumerable. To generate the instances of $X_\rho(S_P)$, we generate the instances of X one after another and output instance I if and only if $\rho(I)$ is in S_P.

We finally show that $X_\rho(S_P)$ satisfies P2 and P3. It is sufficient to prove that $X_\rho(S_P)$ satisfies P2 since ρ is single-valued. Assume there exists a finite set $K \subseteq S_P$ such that $X_\rho(K) \notin \mathrm{P}$. Let $\emptyset \subset K' \subseteq K$ be a subset such that $X_\rho(K')$ is a member of P; such a set exists since $K \subseteq S_P$. For every $k' \in K'$, we know that $X_\rho(\{k'\}) \in \mathrm{P}$. Hence, we can apply Lemma 2 and deduce that there exists a polynomial-time reduction from $X_\rho(K)$ to $X_\rho(K')$. This contradicts the fact that $X_\rho(K)$ is not a polynomial-time solvable problem. We can now apply Theorem 3 and conclude that there exists a set $T \subseteq S_P$ such that $X_\rho(T)$ is NP-intermediate, i.e. we are in case (1). □

Problems parameterized by multi-valued measure functions are apparently very different from those parameterized by single-valued functions. For instance, Lemma 2 breaks down which indicates that the proof strategy used in Theorem 5 is far from sufficient to attack the multi-valued case.

4.2 Structurally Restricted CSPs

When identifying tractable (i.e. polynomial-time solvable) fragments of constraint satisfaction problems and similar problems, two main types of results have been considered in the literature. The first one is to identify constraint languages Γ such that $\mathrm{CSP}(\Gamma) \in \mathrm{P}$, and the second one is to restrict the structure induced by the constraints on the variables. The second case is often concerned with associating some structure with each instance and then identifying sets of structures that yield tractable problems. The classical example of this approach is to study the *primal graph* or *hypergraph* of CSP instances. Given a CSP instance I with variable set V, we define its primal graph $G = (V, E)$ such

that $(v_i, v_j) \in E$ if and only if variables v_i, v_j occur simultaneously in some constraint, and we define the hypergraph $\mathcal{H} = (V, \mathcal{E})$ such that the hyperedge $\{v_{i_1}, ..., v_{i_k}\} \in \mathcal{E}$ if and only if there is a constraint $R(v_{i_1}, \ldots, v_{i_k})$ in I.

When it comes to defining structurally restricted problems that are tractable, one is typically interested in certain parameters of these (hyper)graphs such as *tree-width*, *fractional hypertree width* [16], or *submodular width* [22]. It is, for instance, known that any finite-domain CSP instance I with primal graph $G = (V, E)$ can be solved in $||I||^{O(\mathrm{tw}(G))}$ time [11] where $\mathrm{tw}(G)$ denotes the tree-width of G, and it can be solved in $||I||^{O(\mathrm{fhw}(\mathcal{H}))}$ time [16] where $\mathrm{fhw}(\mathcal{H})$ denotes the fractional hypertree width of $\mathcal{H}$. Since these results rely on the domains being finite, we restrict ourselves to finite-domain CSPs throughout this section. Now note that if given a finite constraint language Γ, then the instances of $\mathrm{CSP}(\Gamma)$ are recursively enumerable and $\mathrm{CSP}(\Gamma)$ is in NP. If Γ is infinite, then this is not so evident and it may, in fact, depend on the representation of relations. We adopt a simplistic approach and represent a relation by listing its tuples. Under this assumption, the instances of $\mathrm{CSP}(\Gamma)$ are recursively enumerable and $\mathrm{CSP}(\Gamma)$ is in NP.

By restricting the CSP problem to instances with tree-width or fractional hypertree width $\leq k$ (for some constant k), it is known that the resulting problem is solvable in polynomial time. This immediately implies that problems like $\mathrm{CSP}_{\mathrm{tw}}$ and $\mathrm{CSP}_{\mathrm{fhw}}$[1] have property P2. If the width parameter under consideration is polynomial-time computable, then we have property P3 (via Lemma 2), too, and conclude that NP-intermediate fragments exist. Unfortunately, this is typically not the case. It is for instance NP-complete to determine whether a given graph G has treewidth at most k or not [2] if k is part of the input. This is a common feature that holds for, or is suspected to hold for, many different width parameters. Hence, width parameters are a natural source of single-valued measure functions that are not polynomial-time computable. Such measure functions are problematic since we cannot prove the existence of NP-intermediate subproblems by using simplifying results like Proposition 4 or Theorem 5. By a few additional assumptions we can however still prove the applicability of Theorem 3. Note that if k is fixed, and thus not part of the input, then the graphs with tree-width $\leq k$ can be recognized in linear time [5]. This is not uncommon when studying width parameters — determining the width exactly is computationally hard but it can be computed or estimated in polynomial time under additional assumptions. We arrive at the following result.

Proposition 6. *Assume that X is a decision problem and ρ is a single-valued measure function such that $X_\rho(\cdot)$ satisfies conditions P0 and P1. Furthermore suppose that for each set $\{0, \ldots, k\}$ there exists a promise algorithm A_k for $X_\rho(\{0, \ldots, k\})$ with the following properties:*

- *if $\rho(I) \leq k$, then A_k returns the correct answer in $p_k(||I||)$ steps where p_k is a polynomial only depending on k, and*

[1] We slightly abuse notation since tw and fhw are not directly defined on problem instances.

– *if $\rho(I) > k$, then A_k either return a correct answer or do not answer at all.*

Then there exists a set $S \subset \mathbb{N}$ such that $X_\rho(S)$ is NP-intermediate.

Proof. Let X^k denote the computational problem X restricted to instances $I \in I(X)$ such that $\rho(I) \geq k$. Assume there exists a k such that $X^k \in P$ and let B be an algorithm for this problem running in time $q(||I||)$ for some polynomial q. For $X_\rho(\{0, \ldots, k-1\})$, we have algorithm A_{k-1} described above. Given an arbitrary instance I of X, we may not be able to compute $\rho(I)$ and choose which algorithm to run. Do as follows: run algorithm A_{k-1} for $p_{k-1}(||I||)$ steps on input I. If A_{k-1} produces an answer, then this is correct. If A_{k-1} does not produce an answer, then we know that $\rho(I) > k-1$ and we can apply algorithm B. All in all, this takes $O(p_{k-1}(||I||) + q(||I||))$ time so $X \in \mathrm{P}$ which leads to a contradiction.

If X^k is in NPI for some k, then we simply let $S = \{k, k+1, \ldots\}$. We can henceforth assume that X^k is NP-complete for all k. Obviously, $X_\rho(\mathbb{N})$ satisfies property P2 since algorithm A_k, $k \geq 0$, runs in polynomial time. We show that it satisfies property P3, too. Let $T \subseteq \mathbb{N}$ be a finite set and let $m = \max T$. We know that X^{m+1} is NP-complete. Hence, there exists a polynomial-time reduction from the NP-complete problem $X_\rho(\mathbb{N})$ to X^{m+1} which, in turn, admits a trivial polynomial-time reduction to $X_\rho(\mathbb{N} \setminus T)$ since $\{m+1, m+2, \ldots\} \subseteq \mathbb{N} \setminus T$. We can now apply Theorem 3 and obtain the set S. □

We apply this result to $\mathrm{CSP}_{\mathrm{tw}}$ and $\mathrm{CSP}_{\mathrm{fhw}}$, respectively. Clearly, both these problems satisfy properties P0 and P1 due to the assumptions that we have made. For $\mathrm{CSP}_{\mathrm{tw}}$, we let A_k work as follows: given a CSP instance I, check whether I has treewidth $\leq k$ using Bodlaender's [5] algorithm. If the algorithm answers "no", then go into an infinite loop. Otherwise, decide whether I has a solution or not in $||I||^{O(k)}$ time. Proposition 6 implies that there exists a set $T \subseteq \mathbb{N}$ such that $\mathrm{CSP}_{\mathrm{tw}}(T)$ is NP-intermediate. We observe that Grohe [15] has shown a similar result under the assumption that FPT $\neq$ W[1] instead of P $\neq$ NP. Many other width parameters can also be used for obtaining NP-intermediate problems. One example is $\mathrm{CSP}_{\mathrm{fhw}}$ for which the proof is very similar but is instead based on Theorem 4.1 in Marx [21].

5 Multi-Valued Measure Functions

In this section we turn our attention to multi-valued measure functions and apply them to constraint problems. Throughout this section we assume that P $\neq$ NP. Here, we want to associate the complexity of CSPs with constraint languages and multi-valued measure functions are convenient for this purpose. Given a constraint satisfaction problem parameterized with a constraint language Γ, let ρ denote the single-valued measure function defined to return the highest arity of any constraint in a given instance: $\rho((V, C)) = \max\{k \mid R(v_1, \ldots, v_k) \in C\}$. Let $\mathrm{CSP}^*_\rho(X)$ denote the $\mathrm{CSP}(\Gamma)$ problem restricted to instances I such that $\rho(I) \in X$, and assume there exists a set $X \subset \mathbb{N}$ such that $\mathrm{CSP}^*_\rho(X)$ is NP-intermediate. Can we from this conclude that there exists a constraint language

$\Gamma' \subset \Gamma$ such that $\mathrm{CSP}(\Gamma')$ is NP-intermediate? In general, the answer is no since the set of valid instances of $\mathrm{CSP}^*_\rho(X)$ are not in a one-to-one correspondence with any constraint language restriction. Note that $\mathrm{CSP}^*_\rho(X)$ is not the same problem as $\mathrm{CSP}(\{R \in \Gamma \mid ar(R) \in X\})$. If we on the other hand define the multi-valued measure function $\sigma((V,C)) = \{k \mid R(v_1,\ldots,v_k) \in C\}$, then for every $X \subset \mathbb{N}$ the problem $\mathrm{CSP}^*_\sigma(X)$ is equivalent to $\mathrm{CSP}(\{R \in \Gamma \mid \mathrm{ar}(R) \in X\})$.

5.1 Constraint Satisfaction Problems and the Local-Global Conjecture

A constraint language Γ is said to have the *local-global* property [4] if $\mathrm{CSP}(\Gamma') \in \mathrm{P}$ for every finite set $\Gamma' \subset \Gamma$ implies $\mathrm{CSP}(\Gamma) \in \mathrm{P}$. The non-existence of languages not having the local-global property is known as the *local-global conjecture*. In Bodirsky & Grohe [4] it is proven that if Γ is a constraint language over a finite domain D that does not exhibit the local-global property, then there exists a constraint language Γ' over D such that $\mathrm{CSP}(\Gamma')$ is NP-intermediate. In this section we prove a more general result not restricted to finite domains based on the notion of *extension operators*. If R is a k-ary relation and Γ a constraint language over a domain D we say that R has a *primitive positive* (p.p.) definition in Γ if $R(x_1,\ldots,x_k) \equiv \exists y_1,\ldots,y_l \,.\, R_1(\mathbf{x_1}) \wedge \ldots R_i(\mathbf{x_i})$, where each $R_j \in \Gamma \cup \{=\}$ and each $\mathbf{x_i}$ is a vector over $x_1,\ldots,x_k$, $y_1,\ldots,y_l$.

Definition 7. *Let Γ be a recursively enumerable constraint language (with a suitable representation of relations in Γ). We say that $\langle\cdot\rangle$ is an* extension operator *if (1) $\langle\Gamma\rangle$ is a recursively enumerable set of p.p. definable relations over Γ and (2) whenever $\Delta \subset \langle\Gamma\rangle$ and $\langle\Gamma\rangle \setminus \Delta$ is finite, then every $R \in \langle\Gamma\rangle \setminus \Delta$ is p.p. definable in Δ.*

Another way of viewing this is that the expressive power of $\langle\Gamma\rangle$ does not change when removing finitely many relations. Since Γ and $\langle\Gamma\rangle$ are recursively enumerable we can enumerate relations in Γ or $\langle\Gamma\rangle$ as $R_1, R_2, \ldots$, and it is not hard to see that this implies that instances of $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\langle\Gamma\rangle)$ are also recursively enumerable. Given an instance I of $\mathrm{CSP}(\Gamma)$ containing the relations $R_{i_1},\ldots,R_{i_k}$, we let $\rho(I) = \{i_1,\ldots,i_k\}$. Let $\mathrm{CSP}^*_\rho(S)$ denote the $\mathrm{CSP}(\Gamma)$ problem over instances I such that $\rho(I) \subseteq S$. Define the measure function ρ' analogous to ρ but for instances over $\mathrm{CSP}(\langle\Gamma\rangle)$, and let $\mathrm{CSP}^\times_{\rho'}(S)$ be the $\mathrm{CSP}(\langle\Gamma\rangle)$ problem restricted to instances I such that $\rho'(I) \subseteq S$.

Theorem 8. *Assume Γ is a constraint language such that $CSP^*_\rho(\mathbb{N})$ satisfies property P0 – P2. Let $\langle\cdot\rangle$ be an extension operator such that $CSP^\times_{\rho'}(\langle\Gamma\rangle)$ satisfies property P0 – P1. Then there exists a $\Gamma' \subset \langle\Gamma\rangle$ such that $CSP(\Gamma')$ is NP-intermediate.*

Proof. We prove that $\mathrm{CSP}^\times_{\rho'}(\mathbb{N})$ satisfies property P0 – P3. The first two properties are trivial by assumption. For property P2 let $T = \{i_1,\ldots,i_k\}$ be an arbitrary finite subset of $\mathbb{N}$ and let $\Theta = \{R_{i_1},\ldots,R_{i_k}\}$. Note that Θ might contain relations which are not included in Γ. For every such relation $R \in \Theta$ we can

however replace it by its p.p. definition in Γ. Let the resulting set of relations be Θ' and let $S = \{i \mid R_i \in \Theta'\}$. Then $\mathrm{CSP}^{\times}_{\rho'}(T)$ and $\mathrm{CSP}^{*}_{\rho}(S)$ are polynomial-time equivalent since T is a finite set. Since $\mathrm{CSP}^{*}_{\rho}(S)$ is solvable in polynomial time by assumption, $\mathrm{CSP}^{\times}_{\rho'}(T)$ is polynomial-time solvable too.

For property P3 let $T \subset \mathbb{N}$ such that $\mathbb{N} \setminus T = \{t_1, \ldots, t_k\}$. To see that there exists a polynomial-time reduction from $\mathrm{CSP}^{\times}_{\rho'}(\mathbb{N})$ to $\mathrm{CSP}^{\times}_{\rho'}(T)$, we let I be an arbitrary instance of $\mathrm{CSP}^{\times}_{\rho'}(\mathbb{N})$. Assume I contains the constraint $R_i(x_1, \ldots, x_m)$, $i \in \mathbb{N} \setminus T$. Since $\langle\cdot\rangle$ is an extension operator the relation R_i is p.p. definable in $\langle\Gamma\rangle \setminus \Delta$ where $\Delta = \{R_i \mid i \in \mathbb{N} \setminus T\}$. Thus, we can replace $R_i(x_1, \ldots, x_m)$ with its p.p. definition in $\langle\Gamma\rangle \setminus \Delta$, and by doing this for all constraints that are not allowed by T, we end up with an instance I' of $\mathrm{CSP}^{\times}_{\rho'}(T)$ that is satisfiable if and only if I is satisfiable. This is a polynomial-time reduction since $\mathbb{N} \setminus T$ is a finite set.

By applying Theorem 3, we can now identify a set $S \subset \mathbb{N}$ such that $\mathrm{CSP}^{\times}_{\rho'}(S)$ is NP-intermediate. This implies that $\mathrm{CSP}(\Gamma')$ is NP-intermediate when $\Gamma' = \{R_i \in \langle\Gamma\rangle \mid i \in S\}$. □

Our first extension operator is based on the idea of extending a relation into a relation with higher arity. For any relation $R \subseteq D^n$, we define the kth power of R to be the relation $R^k(x_0, \ldots, x_{k \cdot n-1}) \equiv R(x_0, \ldots, x_{n-1}) \wedge R(x_n, \ldots, x_{n+n-1}) \wedge R(x_{2n}, \ldots, x_{2n+n-1}) \wedge \ldots \wedge R(x_{(k-1)n}, \ldots, x_{(k-1)n+n-1})$. Given a constraint language Γ, let $\langle\Gamma\rangle_{pow} = \{R^k \mid R \in \Gamma \text{ and } k \in \mathbb{N}\}$. We represent each relation in $\langle\Gamma\rangle_{pow}$ as a pair (R, k). It is easy to see that $\mathrm{CSP}(\langle\Gamma\rangle_{pow}) \in \mathrm{NP}$ if $\mathrm{CSP}(\Gamma) \in \mathrm{NP}$ from which it follows that $\mathrm{CSP}(\langle\Gamma\rangle_{pow})$ is NP-complete. Now assume that $\Delta \subset \langle\Gamma\rangle_{pow}$ and that $\langle\Gamma\rangle_{pow} \setminus \Delta$ is finite. First, for every $R^k \in \langle\Gamma\rangle_{pow} \setminus \Delta$ we can p.p. define R^k in Δ as $R(x_1, \ldots, x_n) \equiv \exists x_{n+1}, \ldots, x_{k' \cdot n+n-1}.R^{k'+1}(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{k' \cdot n+n-1})$, where $k' > k$. Such a k' must exist since we have only removed finitely many relations from $\langle\Gamma\rangle_{pow}$. Hence $\langle\cdot\rangle_{pow}$ is an extension operator. Extension operators are not uncommon in the literature. Well studied examples (provided relations can be suitably represented) include closure under p.p. definitions (known as *co-clones*) and closure under p.p. definitions without existential quantification (known as *partial co-clones*). These are indeed extension operators since $\langle\Gamma\rangle_{pow}$ is always a subset of the partial co-clone of Γ and hence also of the co-clone of Γ. For a general introduction to the field of clone theory we refer the reader to Lau [26].

Let $R_{a,b,c,U} = \{(x, y) \in \mathbb{Z}^2 \mid ax - by \leq c, 0 \leq x, y \leq U\}$ for arbitrary $a, b, U \in \mathbb{N}$ and $c \in \mathbb{Z}$. Furthermore let $\Gamma'_U = \{R_{a,b,c,U} \mid a, b \in \mathbb{N}, c \in \mathbb{Z}\}$ for any $U \in \mathbb{N}$ and the language Γ° be defined as $\Gamma^{\circ} = \bigcup_{i=0}^{\infty} \Gamma'_i$. Note that we can represent each relation in Γ° compactly by four integers written in binary. Due to Jonsson & Lööw [18] it is known that Γ° does not satisfy the local-global property. By combining the language Γ° and the extension operator $\langle\cdot\rangle_{pow}$ with Theorem 8 we thus obtain the following result.

Theorem 9. *There exists a $\Gamma' \subset \langle\Gamma^{\circ}\rangle_{pow}$ such that CSP(Γ') is NP-intermediate.*

Due to the work of Bodirsky & Grohe [4] we already know that the CSP problem over infinite domains is non-dichotomizable. Their result is however

based on reducing an already known NP-intermediate problem to a CSP problem while our language $\Gamma' \subset \langle \Gamma^{\circ} \rangle_{pow}$ is an explicit example of a locally tractable language obtained via blowing holes.

5.2 Locally Tractable Languages with Bounded Arity

The downside of the $\langle \cdot \rangle_{pow}$ operator is that the construction creates relations of arbitrary high arity even if the language only contain relations of bounded arity. In this section we show that simpler extensions are sometimes applicable for constraint languages over infinite domains. For any k-ary relation R we define the $(k+1)$-ary relation R_a as $R_a(x_1, \ldots, x_n, y) \equiv R(x_1, \ldots, x_n) \wedge (y = a)$, where $a \in D$ and $(y = a)$ is the constraint application of the relation $\{(a)\}$. Let $\langle \Gamma \rangle_+ = \{R_a \mid R \in \Gamma, a \in D\}$. If we represent each relation in $\langle \Gamma \rangle_+$ as a tuple (R, a) then obviously $\langle \Gamma \rangle_+$ is recursively enumerable if Γ is recursively enumerable. Now assume that Γ is an infinite constraint language and that $\langle \Gamma \rangle_+ \setminus \Delta$ is finite. For any relation $R_a \in \langle \Gamma \rangle_+ \setminus \Delta$ we first determine a b such that $R_b \in \Delta$. By construction there exists such a b since $\langle \Gamma \rangle_+ \setminus \Delta$ is finite. Then, since Γ is infinite, there exists an m-ary relation $R' \in \Gamma$ such that $R'_a \in \Delta$. Hence we can implement R_a as $R_a(x_1, \ldots, x_n, y) \equiv \exists y', x'_1, \ldots, x'_m . R_b(x_1, \ldots, x_n, y') \wedge R'_a(x'_1, \ldots, x'_m, y)$, by which it follows that $\langle \cdot \rangle_+$ is an extension operator.

Say that a language Γ is *idempotent* if for all $a \in D$ it holds that $\{(a)\}$ is p.p. definable in Γ. We assume that we can find the p.p. definition of $\{(a)\}$) in Γ in polynomial time.

Theorem 10. *Let Γ be an idempotent language over an infinite domain such that Γ does not satisfy the local-global property. Then there exists a constraint language Γ' such that (1) CSP(Γ') is NP-intermediate and (2) Γ' contains only relations of arity $k+1$, where k is the highest arity of a relation in Γ.*

Proof. Let $R_1, R_2, \ldots$ be an enumeration of Γ and define the measure function ρ over an instance I containing the relations $R_{i_1}, \ldots, R_{i_k}$ as $\rho(I) = \{i_1, \ldots, i_k\}$. We note that Γ must be infinite since it does not satisfy the local-global property. Let $\mathrm{CSP}^*_\rho(S)$ denote the CSP(Γ) problem over instances I such that $\rho(I) \subseteq S$. Then $\mathrm{CSP}^*_\rho(\mathbb{N})$ obviously satisfies property P0–P2, and since $\langle \cdot \rangle_+$ is an extension operator, we only need to prove that CSP($\langle \Gamma \rangle_+$) is NP-complete. NP-hardness is easy since CSP(Γ) is trivially polynomial-time reducible to CSP($\langle \Gamma \rangle_+$). For membership in NP we give a polynomial-time reduction from CSP($\langle \Gamma \rangle_+$) to CSP(Γ). Let I be an arbitrary instance of CSP($\langle \Gamma \rangle_+$). For any constraint $R_a(x_1, \ldots, x_n, y)$ we replace it by $R(x_1, \ldots, x_n) \wedge \phi(x'_1, \ldots, x'_m, y)$, where $\exists x'_1, \ldots, x'_m . \phi$ is the p.p. definition of $y = a$, which is computable in polynomial time by assumption. If we repeat the procedure for all R_a in I we get an instance I' of CSP(Γ) which is satisfiable if and only if I is satisfiable. Hence there exists a $\Gamma' \subset \langle \Gamma \rangle_+$ such that CSP(Γ') is NP-intermediate by Theorem 8. Let k denote the highest arity of a relation in Γ. By definition every relation in $\langle \Gamma \rangle_+$ then has its arity bounded by $k+1$, which trivially also holds for Γ'. □

It is not hard to see that for the constraint language Γ° defined in the previous section any constant relation is p.p. definable in polynomial time. For any $a \in \mathbb{N}$ we simply let $(y = a) \equiv \exists x.R_{0,1,a,a}(x, y)$, i.e. the relation $0 \cdot x - 1 \cdot y \leq a \wedge 0 \leq x, y \leq a$. By Theorem 10 and the fact that Γ° only contains relations of arity 2 we therefore obtain the following.

Theorem 11. *There exists a $\Gamma' \subset \langle\Gamma^\circ\rangle_+$ such that (1) CSP(Γ') is NP-intermediate and (2) Γ' contains only relations of arity 3.*

5.3 Propositional Abduction

Abduction is a fundamental form of nonmonotonic reasoning whose computational complexity has been thoroughly investigated [10, 13, 23]. It is known that the abduction problem parameterized with a finite constraint language is always in P, NP-complete, coNP-complete or Σ_2^{P}-complete. For infinite languages the situation differs and the question of whether it is possible to obtain a similar classification was left open in [23]. We will show that there exists an infinite constraint language such that the resulting abduction problem is NP-intermediate.

Let Γ denote a constraint language and define the propositional abduction problem $\textsc{Abd}(\Gamma)$ as follows.

INSTANCE. An instance I of $\textsc{Abd}(\Gamma)$ consists of a tuple (V, H, M, KB), where V is a set of Boolean variables, H is a set of literals over V (known as the *set of hypotheses*), M is a literal over V (known as the the *manifestation*), and KB is a set of constraint applications $C_1(\mathbf{x}_1) \wedge ... \wedge C_k(\mathbf{x}_k)$ where C_i denotes an application of some relation in Γ and $\mathbf{x_i}$, $1 \leq i \leq k$, is a vector of variables in V (KB is known as the *knowledge base*).
QUESTION. Does there exist an *explanation* for I, i.e., a set $E \subseteq H$ such that $KB \wedge \bigwedge E$ is satisfiable and $KB \wedge \bigwedge E \models M$, i.e. $KB \wedge \bigwedge E \wedge \neg M$ is not satisfiable.

Let Γ_{IHSB-} be the infinite constraint language consisting of the relations expressed by the clauses $(x), (\neg x \vee y)$ and all negative clauses, i.e., $\{(\neg x_1 \vee \cdots \vee \neg x_n) \mid n \geq 1\}$. We may represent each relation is Γ_{IHSB-} with a natural number in the obvious way. Let the finite constraint language $\Gamma_{IHSB-/k}$ be the subset of Γ_{IHSB-} that contains all clauses C such that $ar(C) = k$. In light of this we define the multi-valued measure function $\rho(I) = \{ar(C) \mid C \text{ is a negative clause of } KB \text{ in } I\}$. With the chosen representation of relations, ρ is obviously polynomial-time computable. We define the corresponding parameterized abduction problem $\textsc{Abd}^*_\rho(\Gamma)$ such that $I(\textsc{Abd}^*)$ is the set of abduction instances over Γ_{IHSB-}. We now verify that $\textsc{Abd}^*_\rho(\mathbb{N})$ fulfills property P0 – P3.

Property P0 holds trivially while property P1 follows from [23]. For property P2, we note that if T is an arbitrary finite subset of $\mathbb{N}$, then there exists a $k \in T$ such that the clauses of every $\textsc{Abd}^*_\rho(T)$ instance is bounded by k. By [23], we know that $\textsc{Abd}(\Gamma_{IHSB-/k})$ is in P for every k, and hence that $\textsc{Abd}^*_\rho(T)$ is in P for every finite subset of S. To show property P3, we present a polynomial-time reduction from $\textsc{Abd}^*_\rho(\mathbb{N})$ to $\textsc{Abd}^*_\rho(T)$ when $\mathbb{N} \setminus T$ is finite. Let $k = \max(\mathbb{N} \setminus T)$. Arbitrarily choose an instance $I = (V, H, M, KB)$ of $\textsc{Abd}^*_\rho(\mathbb{N})$. Then, for every

clause $C = (\neg x_1 \vee \ldots \vee \neg x_l) \in KB$ such that $l \in S \setminus T$, replace C by the logically equivalent clause $C' = (\neg x_1 \vee \ldots \vee \neg x_{l-1} \vee \neg x_l \vee \underbrace{\neg x_l \ldots \vee \neg x_l}_{k+1-l \ \neg x_l\text{'s}})$ of length $k+1$. If we let the resulting knowledge base be KB' then $I' = (V, H, M, KB')$ is an instance of $\text{ABD}^*_\rho(T)$ which has a solution if and only if I has a solution.

From this and Theorem 3 it follows that that there exists a $S' \subset \mathbb{N}$ such that $\text{ABD}^*_\rho(S')$ is NP-intermediate. Hence we conclude the following.

Theorem 12. *There exists a constraint language $\Gamma'_{IHSB-} \subset \Gamma_{IHSB-}$ such that $\text{ABD}(\Gamma'_{IHSB-})$ is NP-intermediate.*

6 Future Work

One way of obtaining genuinely new NP-intermediate problems is to consider other complexity-theoretic assumptions than P $\neq$ NP. We have pointed out that the LogClique problem is NP-intermediate under the ETH, and that the main difficulty is to provide a lower bound, i.e. proving that LogClique $\notin$ P. One may suspect that providing lower bounds is the main difficulty also when considering other problems. We have seen that CSP problems constitute a rich source of NP-intermediate problems via different kinds of parameterization, Hence, it appears feasible that methods for studying the complexity of parameterized problems will become highly relevant. In particular, *linear fpt-reductions* [7, 8] have been used for proving particularly strong lower bounds which may be used for linking together NP-intermediate problems, parameterized problems, and lower bound assumptions. Another way is to adapt and use recent methods for studying the time complexity of Boolean CSP problems [17]. These methods aim at obtaining reductions that provide a fine-grained picture of time complexity and this may be useful when studying NP-intermediate problems. Additionally, recent results by Dell and van Melkebeek [12] can be used for proving the non-existence of such reductions.

We have shown that the propositional abduction problem has NP-intermediate fragments. One may view abduction as a problem that is closely related to Boolean CSPs. However, there is an important difference: the CSP(Γ) problem is either a member of P or NP-complete for *all* choices of Boolean Γ. Hence, it would be interesting to determine which finite-domain CSP-related problems can be used for obtaining NP-intermediate problems and which of them have the local-global property. Inspired by our result on the abduction problem, we view other forms of non-monotonic reasoning such as circumscription and default logic as potential candidates. Unfortunately, many problems of this type are polynomial-time solvable only in very restricted cases, which makes it hard to find a candidate language resulting in a problem not having the local-global property. Thus, more powerful methods than blowing may be needed for identifying NP-intermediate problems in this and similar cases.

References

1. S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.
2. S. Arnborg, D. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Matrix Analysis and Applications*, 8(2):277–284, 1987.
3. M. Bodirsky. *Complexity Classification in Infinite-Domain Constraint Satisfaction*. Habilitation thesis. Univ. Paris 7, 2012.
4. M. Bodirsky and M. Grohe. Non-dichotomies in constraint satisfaction complexity. In *Proc. 35th International Colloquium on Automata, Languages and Programming (ICALP-2008)*, pages 184–196, 2008.
5. H. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.
6. A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the computational complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.
7. J. Chen, B. Chor, M. Fellows, X. Huang, D. Juedes, I. Kanj, and G. Xia. Tight lower bounds for certain parameterized np-hard problems. In *Proc. IEEE Conference on Computational Complexity (CCC-2004)*, pages 150–160, 2004.
8. J. Chen, X. Huang, I. Kanj, and G. Xia. Linear fpt reductions and computational lower bounds. In *Proc. 36th ACM Symposium on Theory of Computing (STOC-2004)*, pages 212–221, 2004.
9. Y. Chen, M. Thurley, and M. Weyer. Understanding the complexity of induced subgraph isomorphisms. In *Proc. 35th International Colloquium on Automata, Languages and Programming (ICALP-2008)*, pages 587–596, 2008.
10. N. Creignou, J. Schmidt, and M. Thomas. Complexity of propositional abduction for restricted sets of boolean functions. In *Proc. 12th International Conference on the Principles of Knowledge Representation and Reasoning (KR-2010)*, 2010.
11. R. Dechter. *Constraint Processing*. Elsevier Morgan Kaufmann, 2003.
12. H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. In *Proc. 42nd ACM Symposium on Theory of Computing (STOC-2010)*, pages 251–260, 2010.
13. T. Eiter and G. Gottlob. The complexity of logic-based abduction. *Journal of the ACM*, 42(1):3–42, 1995.
14. M. Garey and D. Johnson. "Strong" NP-completeness results: motivation, examples and implications. *Journal of the ACM*, 25(3):499–508, 1978.
15. M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *Journal of the ACM*, 54(1), article 1, 2007.
16. M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *Proc. 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-2006)*, pages 289–298, 2006.
17. P. Jonsson, V. Lagerkvist, G. Nordh, and B. Zanuttini. Complexity of SAT problems, clone theory and the exponential time hypothesis. In *Proc. the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2013)*.
18. P. Jonsson and T. Lööw. Computational complexity of linear constraints over the integers. *Artificial Intelligence*, 195:44–62, 2013.
19. R. M. Karp. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

20. R. Ladner. On the structure of polynomial time reducibility. *Journal of the ACM*, 22:155–171, 1975.
21. D. Marx. Approximating fractional hypertree width. *ACM Transactions on Algorithms*, 6(2), 2010.
22. D. Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. In *Proc. 42nd ACM Symposium on Theory of Computing (STOC-2010)*, pages 735-744, 2010.
23. G. Nordh and B. Zanuttini. What makes propositional abduction tractable. *Artificial Intelligence*, 172:1245–1284, 2008.
24. C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
25. U. Schöning. A uniform approach to obtain diagonal sets in complexity classes. *Theoretical Computer Science*, 18:95–103, 1982.
26. D. Lau. *Function Algebras on Finite Sets: Basic Course on Many-Valued Logic and Clone Theory (Springer Monographs in Mathematics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.