Backdoors into Heterogeneous Classes of SAT and CSP[☆]

Serge Gaspers[a], Neeldhara Misra[b], Sebastian Ordyniak[c,*], Stefan Szeider[c], Stanislav Živný[d]

[a] *UNSW and NICTA, Sydney, Australia*
[b] *Indian Institute of Science, Bangalore, India*
[c] *Algorithms and Complexity group, TU Wien, Vienna, Austria*
[d] *University of Oxford, Oxford, UK*

Abstract

In this paper we extend the classical notion of strong and weak backdoor sets for SAT and CSP by allowing that different instantiations of the backdoor variables result in instances that belong to different base classes; the union of the base classes forms a heterogeneous base class. Backdoor sets to heterogeneous base classes can be much smaller than backdoor sets to homogeneous ones, hence they are much more desirable but possibly harder to find. We draw a detailed complexity landscape for the problem of detecting strong and weak backdoor sets into heterogeneous base classes for SAT and CSP.

Keywords: constraint satisfaction, satisfiability, backdoor set, parameterized complexity

1. Introduction

We consider the propositional satisfiability problem (SAT) and the constraint satisfaction problem (CSP). Backdoors are small sets of variables of a SAT or CSP instance that represent "clever reasoning shortcuts" through the search space. Backdoor sets were originally introduced by Williams, Gomes, and Selman ([2, 3]) as a tool for the analysis of decision heuristics in propositional satisfiability. Since then, backdoor sets have been widely used in the areas of propositional satisfiability [2, 4, 5, 6, 7, 8, 9], and also for material discovery [10], abductive reasoning [11], argumentation [12], quantified Boolean formulas [13], and planning [14, 15]. A backdoor set is defined with respect to some fixed *base class* for which the computational problem under consideration is polynomial-time tractable (alternatively, it can be defined with respect to a polynomial-time subsolver). The size of the backdoor set can be seen as a distance measure that indicates how far the instance is from the target class. One distinguishes between strong and weak backdoor sets; the latter applies only to satisfiable instances. Once a strong backdoor set of size k is identified, one can decide the satisfiability of the instance by deciding the satisfiability of at most d^k "easy" instances that belong to the tractable base class, where d denotes the size of the domain for the variables; for SAT we have $d = 2$. Each of the easy instances is obtained by one of the d^k possible instantiations of the k variables in the backdoor set. Hence, the satisfiability check is *fixed-parameter tractable* [16] for the combined parameter backdoor size and domain size $(k + d)$. A similar approach works for weak backdoor sets, where the computation of a certificate for satisfiability (i.e., a satisfying assignment) is fixed-parameter tractable for the combined parameter $(k + d)$.

The fixed-parameter tractability of using the backdoor set for deciding satisfiability or the computation of a certificate for satisfiability triggers the question of whether *finding* a backdoor set of size at most k is also fixed-parameter tractable. In particular, for every base class $\mathcal{C}$ one can ask whether the detection of a

strong or weak backdoor set into $\mathcal{C}$ of size at most k is fixed-parameter tractable for parameter k (possibly in combination with restrictions on the input or other parameters). A systematic study of the parameterized complexity of backdoor set detection was initiated by Nishimura et al. [17] for SAT, who showed that the detection of strong backdoor sets into the classes HORN and 2CNF is fixed-parameter tractable, but the detection of weak backdoor sets into any of these two classes is W[2]-hard (and hence unlikely to be fixed-parameter tractable). Since then, the parameterized complexity of backdoor set detection has become an active research topic as outlined in a survey [18].

In this work, we provide two significant extensions to the exciting research on fixed-parameter tractable backdoor set detection. First, we introduce *heterogeneous* base classes, which can be composed of several homogeneous base classes. We show that heterogeneous base classes are particularly well-suited for strong backdoor sets, since they allow that different instantiations of the backdoor variables result in instances that belong to different base classes, which leads to arbitrary reductions in the size of a strong backdoor set even compared to the smallest strong backdoor set for each individual homogeneous base class. This is in contrast to weak backdoor sets, where the size of a weak backdoor set into a heterogeneous base class is equal to the size of the smallest backdoor set into any of the individual base classes. Here we show that also the complexity of weak backdoor set detection into heterogeneous base classes is tight to its complexity on the individual base classes. Second, we extend the scope of backdoor set detection from SAT to CSP, considering target classes that are defined by tractable constraint languages in terms of certain *closure properties under polymorphisms*.

Heterogeneous Base Classes. Consider the following SAT instance $F_n = \{C, D_1, \ldots, D_n\}$ where $C = (x \lor \neg a_1 \lor \cdots \lor \neg a_n)$ and $D_i = (\neg x \lor b_i \lor c_i)$. It is easy to see that any strong backdoor set into HORN needs to contain at least one of the variables b_i or c_i from each clause D_i, hence such a backdoor set must be of size $\Omega(n)$; on the other hand, any strong backdoor set into 2CNF must contain at least $(n-2)$ variables from the clause C; hence such a backdoor must be of size at $\Omega(n)$ as well. However, $F_n[x = \text{false}]$ is Horn, and $F_n[x = \text{true}]$ is a 2CNF, hence the singleton $\{x\}$ constitutes a strong backdoor set into the *"heterogeneous"* base class HORN$\cup$2CNF. This example shows that by considering heterogeneous base classes we can access structural properties of instances that are not accessible by backdoor sets into homogeneous base classes. Identifying a base class with a class of instances that are solvable by a particular polynomial-time subsolver, one can consider a heterogeneous base class as a "portfolio subsolver," where for each instance the best suitable subsolver from the portfolio is chosen.

SAT Backdoor Sets. A natural question at this point is whether the fixed-parameter tractability results for the detection of strong backdoor sets into individual base classes can be extended to more powerful heterogeneous base classes. In this work, we completely characterize the complexity landscape for heterogeneous base classes obtained arbitrary combinations of the well-known Schaefer classes [19], in the following denoted by HORN (for Horn formulas), HORN$^-$ (for Anti-Horn formulas), 2CNF (for 2-CNF formulas), 0-VAL (for zero valid formulas), and 1-VAL (for one valid formulas). To state the classification, we briefly introduce some terminology. We say that a pair $(\mathcal{C}, \mathcal{C}')$ of Schaefer classes is a *bad pair* if $\mathcal{C} \in \{\text{HORN}, 0\text{-VAL}\}$ and $\mathcal{C}' \in \{\text{HORN}^-, 1\text{-VAL}\}$.

Let $\mathcal{C}$ be a class of CNF-formulas such that $\mathcal{C} = \bigcup_{s \in S} s$ for some subset S of the Schaefer classes. Our main result for SAT backdoor sets (Theorem 6) is that STRONG $\mathcal{C}$-BACKDOOR DETECTION is fixed parameter tractable if and only if S does not contain a bad pair of Schaefer classes, otherwise it is W[2]-hard. On the other hand, detecting weak backdoors is always W[2]-hard for any choice of $\mathcal{C}$, and we show this by building on the known hardness results for when $\mathcal{C}$ is a singleton set (Theorem 8).

We also show that STRONG $\mathcal{C}$-BACKDOOR DETECTION as well as WEAK $\mathcal{C}$-BACKDOOR DETECTION are fixed-parameter tractable for the combined parameter k and the maximum length r of a clause of the input formula (Theorem 7 and 9). These algorithms formalize the intuition that all the hardness results in the previous classification result exploit clauses of unbounded length.

CSP Backdoor Sets. The identification of tractable classes of CSP instances has been subject of extensive studies. A prominent line of research, initiated by Schaefer [19] in his seminal paper on Boolean CSP, is

to identify tractable classes by restricting the relations that may appear in constraints to a prescribed set, a constraint language. Today, many constraint languages have been identified that give rise to tractable classes of CSPs [20, 21, 22]; typically such languages are defined in terms of certain *closure properties*, which ensure that the relations are closed under pointwise application of certain operations on the domain. For instance, consider a CSP instance whose relations are closed under a constant operation $f(x) = d$ for some $d \in D$ (such a operation is a polymorphism of the relations in the instance). Then note that every relation is either empty or forced to contain the tuple $\langle d, d, \ldots, d \rangle$. Thus, given a particular instance, we may either declare it unsatisfiable (if it contains a constraint over the empty relation), or satisfy it trivially by setting every variable to d. Further examples of polymorphisms for which closure properties yield tractable CSP are min, max, majority, affine, and Mal'cev operations [23, 24].

We study the problem of finding strong and weak backdoor sets into tractable classes of CSP instances defined by certain polymorphisms. Our main result for CSP backdoors (Theorem 10) establishes fixed-parameter tractability for a wide range of such base classes. In particular, we show that the detection of strong and weak backdoor sets is fixed-parameter tractable for the combined parameter backdoor size, domain size, and the maximum arity of constraints. In fact, this result entails *heterogeneous* base classes, as different instantiations of the backdoor variables can lead to reduced instances that are closed under different polymorphisms (even polymorphisms of different type). We complement our main result with hardness results that show that we lose fixed-parameter tractability when we omit either domain size or the maximum arity of constraints from the parameter (Theorems 12 and 13). Hence, Theorem 10 is tight in a certain sense.

Related Work. Recently, two papers dealing with strong backdoor set detection for the constraint satisfaction problem have appeared [25, 26], which nicely supplement (however are mostly orthogonal to) the results in this paper. Carbonnel et al. [25] show W[2]-hardness for strong backdoor set detection parameterized by the size of the backdoor set even for CSP-instances with only one constraint (however with unbounded domain and unbounded arity). They also give a fixed-parameter algorithm for strong backdoor set detection parameterized by the size of the backdoor set and the maximum arity of any constraint, if the base class is "h-Helly" for any fixed integer h. However, as is also mentioned in Carbonnel and Cooper [27], the "h-Helly" property is rather restrictive and orthogonal to our approach of considering the domain as an additional parameter.

Ganian et al. [26] show fixed-parameter tractability of strong backdoor detection parameterized by the size of the backdoor to a very general family of heterogeneous base classes, i.e., base classes defined as all CSP-instances obtained from the disjoint union of tractable, finite, and semi-conservative constraint languages. These base classes are orthogonal to the base classes considered in this paper. They are more general in the sense that they allow for the CSP-instance to consist of pairwise disjoint subinstances each belonging to a different tractable class, and they are more specific in the sense that they only consider finite and semi-conservative constraint languages, which also restricts the CSP-instances to bounded domain and bounded arity.

2. Preliminaries

SAT. A *literal* is a propositional variable x or a negated variable $\neg x$. We also use the notation $x = x^1$ and $\neg x = x^0$. A *clause* is a finite set of literals that does not contain a complementary pair x and $\neg x$. A propositional formula in conjunctive normal form, or *CNF formula* for short, is a set of clauses. For a clause C we write $\mathrm{var}(C) = \{\, x : x \in C \text{ or } \neg x \in C \,\}$, and for a CNF formula F we write $\mathrm{var}(F) = \bigcup_{C \in F} \mathrm{var}(C)$.

For a set X of propositional variables we denote by $X \to \{0, 1\}$ the set of all mappings from X to $\{0, 1\}$, the truth assignments on X. We denote by $\overline{X}$ the set of literals corresponding to the negated variables of X. For $\tau : X \to \{0, 1\}$ we let $\mathrm{true}(\tau) = \{\, x^{\tau(x)} : x \in X \,\}$ and $\mathrm{false}(\tau) = \{\, x^{1-\tau(x)} : x \in X \,\}$ be the sets of literals set by τ to 1 and 0, respectively. Given a CNF formula F and a truth assignment $\tau : X \to \{0, 1\}$ we define $F[\tau] = \{\, C \setminus \mathrm{false}(\tau) : C \in F, \ C \cap \mathrm{true}(\tau) = \emptyset \,\}$. If $\tau : \{x\} \to \{0, 1\}$ and $\epsilon = \tau(x)$, we simply write $F[x = \epsilon]$ instead of $F[\tau]$.

A CNF formula F is *satisfiable* if there is some $\tau : \text{var}(F) \rightarrow \{0,1\}$ with $F[\tau] = \emptyset$, otherwise F is *unsatisfiable*.

CSP. Let D be a set and n and n' be natural numbers. An n-ary relation on D is a subset of D^n. For a tuple $t \in D^n$, we denote by $t[i]$, the i-th entry of t, where $1 \leq i \leq n$. For two tuples $t \in D^n$ and $t' \in D^{n'}$, we denote by $t \circ t'$, the concatenation of t and t'.

An instance of a *constraint satisfaction problem* (CSP) I is a triple $\langle V, D, C \rangle$, where V is a finite set of variables over a finite set (domain) D, and C is a set of constraints. A *constraint* $c \in C$ consists of a *scope*, denoted by $V(c)$, which is an ordered list of a subset of V, and a relation, denoted by $R(c)$, which is a $|V(c)|$-ary relation on D. To simplify notation, we sometimes treat ordered lists without repetitions, such as the scope of a constraint, like sets. For a variable $v \in V(c)$ and a tuple $t \in R(c)$, we denote by $t[v]$, the i-th entry of t, where i is the position of v in $V(c)$. For a CSP instance $I = \langle V, D, C \rangle$ we sometimes denote by $V(I), D(I), C(I)$, and $\delta(I)$, its set of variables V, its domain D, its set of constraints C, and the maximum arity of any constraint of I, respectively.

Let $V' \subseteq V$ and $\tau : V' \rightarrow D$. For a constraint $c \in C$, we denote by $c[\tau]$, the constraint whose scope is $V(c) \setminus V'$ and whose relation contains all $|V(c[\tau])|$-ary tuples t such that there is a $|V(c)|$-ary tuple $t' \in R(c)$ with $t[v] = t'[v]$ for every $v \in V(c[\tau])$ and $t'[v] = \tau(v)$ for every $v \in V'$. We denote by $I[\tau]$ the CSP instance with variables $V \setminus V'$, domain D, and constraints $C[\tau]$, where $C[\tau]$ contains a constraint $c[\tau]$ for every $c \in C$.

A *solution* to a CSP instance I is a mapping $\tau : V \rightarrow D$ such that $\langle \tau[v_1], \ldots, \tau[v_{|V(c)|}] \rangle \in R(c)$ for every $c \in C$ with $V(c) = \langle v_1, \ldots, v_{|V(c)|} \rangle$.

Backdoors. Backdoors are defined relative to some fixed class $\mathcal{C}$ of instances of the problem under consideration (i.e., SAT or CSP). One usually assumes that the problem is tractable for instances from $\mathcal{C}$, as well as that the recognition of $\mathcal{C}$ is tractable.

In the context of SAT, we define a *strong $\mathcal{C}$-backdoor set* of a CNF formula F to be a set B of variables such that $F[\tau] \in \mathcal{C}$ for each $\tau : B \rightarrow \{0,1\}$. A *weak $\mathcal{C}$-backdoor set* of F is a set B of variables such that $F[\tau]$ is satisfiable and $F[\tau] \in \mathcal{C}$ holds for some $\tau : B \rightarrow \{0,1\}$. If we know a strong $\mathcal{C}$-backdoor set of F, we can decide the satisfiability of F by checking the satisfiability of 2^k "easy" formulas $F[\tau]$ that belong to $\mathcal{C}$. Thus SAT decision is fixed-parameter tractable in the size k of the backdoor. Similarly, in the context of CSP, we define a *strong $\mathcal{C}$-backdoor set* of a CSP instance $I = \langle V, D, C \rangle$ as a set B of variables such that $I[\tau] \in \mathcal{C}$ for every $\tau : B \rightarrow D$. We also call a strong $\mathcal{C}$-backdoor a *strong backdoor set into $\mathcal{C}$*. A *weak $\mathcal{C}$-backdoor set* of I is a set B of variables such that $I[\tau]$ is satisfiable and $I[\tau] \in \mathcal{C}$ holds for some $\tau : B \rightarrow D$. If we know a strong $\mathcal{C}$-backdoor set of I of size k, we can reduce the satisfiability of I to the satisfiability of d^k CSP instances in $\mathcal{C}$ where $d = |D|$. Thus deciding the satisfiability of a CSP instance is fixed-parameter tractable in the combined parameter $d + k$.

The challenging problem is—for SAT and for CSP—to find a strong, or weak $\mathcal{C}$-backdoor set of size at most k if one exists.

For each class $\mathcal{C}$ of SAT or CSP instances, we define the following problem.

STRONG $\mathcal{C}$-BACKDOOR DETECTION
Input: A SAT or CSP instance I and a nonnegative integer k.
Question: Does I have a strong $\mathcal{C}$-backdoor set of size at most k?

The problem WEAK $\mathcal{C}$-BACKDOOR DETECTION is defined similarly.

Parameterized Complexity. We provide basic definitions of parameterized complexity; for an in-depth treatment we refer to the recent monograph [16]. A problem is *parameterized* if each problem instance I is associated with a nonnegative integer k, the parameter. A parameterized problem is *fixed-parameter tractable* (or *FPT*, for short) if there is an algorithm, A, a constant c, and a computable function f, such that A solves instances of input size n and parameter k in time $f(k)n^c$. Fixed-parameter tractability extends the conventional notion of polynomial-time tractability, the latter being the special case where f is a polynomial. The so-called Weft-hierarchy $\text{W}[1] \subseteq \text{W}[2] \subseteq \ldots$ contains classes of parameterized decision problems that are presumed to be larger than FPT. It is believed that problems that are hard for any of the classes in the

Weft-hierarchy are not fixed-parameter tractable. The classes are closed under *fpt-reductions* that are fixed-parameter tractable many-one reductions, which map an instance x with parameter k of one problem to a decision-equivalent instance x' with parameter k' of another problem, where $k' \leq f(k)$ for some computable function f.

For instance, the following problem is well-known to be W[2]-complete [16].

HITTING SET   **Parameter:** k
Input: A set system $\mathcal{Q}$ over a set (universe) U, and a positive integer k
Question: Is there a subset $X \subseteq U$, with $|X| \leq k$ such that every set $S \in \mathcal{Q}$ contains at least one element from X?

---

3. Backdoor Detection for SAT

Schaefer's base classes [19] give rise to classes of CNF formulas defined in terms of syntactical properties of clauses.[1] A clause is

- *Horn* if it contains at most one positive literal,

- *Anti-Horn* if it contains at most one negative literal,

- *2CNF* if it contains at most two literals,[2]

- *0-valid* if it contains at least one negative literal, and

- *1-valid* if it contains at least one positive literal.

A CNF formula is *Horn*, *Anti-Horn*, etc. if it contains only Horn, Anti-Horn, etc. clauses. We denote the respective classes of CNF formulas by HORN, HORN$^-$, 2CNF, 0-VAL, and 1-VAL.

3.1. Strong Backdoor Sets

In this section we study the parameterized complexity of STRONG $\mathcal{C}$-BACKDOOR DETECTION, where $\mathcal{C}$ is the union of any subset of the Schaefer classes. In the case that $\mathcal{C}$ consists only of a single Schaefer class this has been intensively studied before [17, 18] and it is known that STRONG $\mathcal{C}$-BACKDOOR DETECTION is polynomial for 0-VAL and 1-VAL, and FPT for the remaining Schaefer classes.

The results in this subsection are as follows. First we show that STRONG $\mathcal{H}$-BACKDOOR DETECTION is fixed-parameter tractable parameterized by the size of the backdoor set for various combinations of the Schaefer classes, i.e., the case that $\mathcal{H} = 2\text{CNF} \cup \mathcal{C}$ for any $\mathcal{C} \in$ Schaefer is shown in Theorem 2, the cases that $\mathcal{H} = \text{HORN} \cup 0\text{-VAL}$ and $\mathcal{H} = \text{HORN}^- \cup 1\text{-VAL}$ are shown in Theorem 3, and finally Theorem 4 shows tractability for the cases that $\mathcal{H} = 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$ and $\mathcal{H} = 2\text{CNF} \cup \text{HORN}^- \cup 1\text{-VAL}$. In contrast, Theorem 5 captures a condition for hardness of the strong backdoor detection problem, which is complementary to all the cases in the first three results. Together, and combined with known results in the literature, these theorems give us a complete complexity classification, which is summarized in Theorem 6.

Our hardness results rely on the fact that the clauses have unbounded length. Theorem 7 shows that if we consider the maximum length of any clause as an additional parameter, then the problem of finding a strong heterogeneous backdoor becomes fixed-parameter tractable for any combination of the Schaefer classes.

As mentioned above we employ branching algorithms for the detection of strong backdoor sets. In particular, all algorithms are based on a depth-bounded search tree approach and mainly differ in the set of employed branching rules. Informally, the algorithms construct a search tree, where each node is labeled with a potential backdoor set and the crucial difference between the algorithms lies in chosing the set of

---

[1] Affine Boolean formulas considered by Schaefer do not correspond naturally to a class of CNF formulas, hence we do not consider them here. However, we do consider the affine case in the context of Boolean CSPs; cf. Theorem 12.

[2] A clause containing exactly two literals is also known as a *Krom* clause [28].

possible extensions of the potential backdoor set on which to branch on. In this way backdoor set detection can be reduced to the following problem, which given a potential backdoor set B' computes the set $\mathcal{Q}$ of potential extensions of B', which can be branched on in the next step.

$\mathcal{C}$-Strong Branch

Input: A CNF formula F with variables V, a non-negative integer k, and a subset $B' \subseteq V$ with $|B'| \leq k$.

Question: Output:

- Yes iff B' is a strong $\mathcal{C}$-backdoor set of F, or

- a family $\mathcal{Q}$ of subsets of $V \setminus B'$ such that:

  B1 for every $Q \in \mathcal{Q}$ it holds that $|B' \cup Q| \leq k$, and

  B2 for every strong $\mathcal{C}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ there is a $Q \in \mathcal{Q}$ such that $B' \cup Q \subseteq B$.

The next theorem shows that Strong $\mathcal{C}$-Backdoor Detection can be reduced to $\mathcal{C}$-Strong Branch via a branching algorithm that uses an algorithm $\mathcal{A}$ for $\mathcal{C}$-Strong Branch as a subroutine to determine the set of braching choices. Importantly the running time of the resulting branching algorithm for Strong $\mathcal{C}$-Backdoor Detection crucially depends on the "type" of braching rules returned by $\mathcal{A}$, or in other words the branching factor. To be able to quantify the branching factor we introduce the function $T_\mathcal{A}(k)$ defined recursively as

$$T_\mathcal{A}(k) = \max_{\mathcal{Q} \in \mathcal{O}} \left(|\mathcal{Q}| T_\mathcal{A}(k - \min_{Q \in \mathcal{Q}} |Q|) \right) \text{ and } T_\mathcal{A}(0) = 0$$

where $\mathcal{O}$ is the set of all possible output sets $\mathcal{Q}$ of $\mathcal{A}$ on any input F, $B' \subseteq V$ with $|B'| \leq k$, and k.

Theorem 1. *Let $\mathcal{C}$ be a class of CNF formulas, F be a CNF formula, k a non-negative integer, and let $\mathcal{A}$ be an algorithm solving $\mathcal{C}$-Strong Branch. Then Strong $\mathcal{C}$-Backdoor Detection can be solved in time $O(R_\mathcal{A}(F,k) \cdot T_\mathcal{A}(k))$, where $R_\mathcal{A}(F,k)$ is the time required by $\mathcal{A}$ on any input F, $B' \subseteq V$ with $|B'| \leq k$, and k.*

Proof. We will employ a branching algorithm that employs the algorithm $\mathcal{A}$ solving $\mathcal{C}$-Strong Branch as a subroutine. The algorithm uses a depth-bounded search tree approach to find a strong $\mathcal{C}$-backdoor set of size at most k. Let F be any CNF-formula with variables V.

We construct a search tree T, for which every node is labeled by a set B of at most k variables of V. Additionally, every leaf node has a second label, which is either Yes or No. T is defined inductively as follows. The root of T is labeled by the empty set. Furthermore, if t is a node of T, whose first label is B', then the children of t in T are obtained as follows. First we run the algorithm $\mathcal{A}$ on F, B', and k. If $\mathcal{A}$ returns Yes then t becomes a leaf node, whose second label is Yes. Otherwise let $\mathcal{Q}$ be the set of subsets of $V \setminus B'$ returned by $\mathcal{A}$. If $\mathcal{Q} = \emptyset$, then there is no strong $\mathcal{C}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ and consequently t becomes a leaf node whose second label is No. Otherwise, i.e., if $\mathcal{Q} \neq \emptyset$ then t has a a child for every $Q \in \mathcal{Q}$, whose first label is $B' \cup Q$. This completes the definition of T. If T has a leaf node, whose second label is Yes, then the algorithm returns the first label of that leaf node as a strong $\mathcal{C}$-backdoor set of F. Otherwise the algorithm returns No. This completes the description of the algorithm.

We now show the correctness of the algorithm. First, suppose the search tree T built by the algorithm has a leaf node t whose second label is Yes. Here, the algorithm returns the first label, say B of t. By definition of T, we obtain that $|B| \leq k$ and B is a strong $\mathcal{C}$-backdoor set of F.

Now consider the case where the algorithm returns No. We need to show that F has no strong $\mathcal{C}$-backdoor set of size at most k. Assume, for the sake of contradiction that such a set B exists.

Observe first that for every leaf node t of T whose second label is No, it holds that $B' \nsubseteq B$. This is because the algorithm $\mathcal{A}$ on input F, B', and k returned the empty set for $\mathcal{Q}$. Hence if t is a leaf node whose first label B' satisfies $B' \subseteq B$, then the second label of t must be Yes.

It hence remains to show that T has a leaf node whose first label is a set B' with $B' \subseteq B$. This will complete the proof about the correctness of the algorithm. We will show a slightly stronger statement, namely, that for every natural number ℓ, either T has a leaf whose first label is contained in B or T has an inner node of distance exactly ℓ from the root whose first label is contained in B. We show the latter by induction on ℓ.

The claim obviously holds for $\ell = 0$. So assume that T contains a node t at distance ℓ from the root of T whose first label, say B', is a subset of B. If t is a leaf node of T, then the claim is shown. Otherwise, the algorithm $\mathcal{A}$ determined that B' is not a strong $\mathcal{C}$-backdoor set of F and returned a set $\mathcal{Q}$ of subsets of $V \setminus B'$ satisfying B2. Hence there is a $Q \in \mathcal{Q}$ with $B' \cup Q \subseteq B$ and because t contains a child t' whose first label is $B' \cup Q$, we obtain that t' is a node at distance $\ell + 1$ satisfying the induction invariant. This concludes the proof concerning the correctness of the algorithm.

The running time of the algorithm is obtained as follows. Let T be a search tree obtained by the algorithm. Then the running time of the depth-bounded search tree algorithm is $O(|V(T)|)$ times the maximum time that is spend on any node of T. Since $|V(T)| \leq T_{\mathcal{A}}(k)$ and the time spend on any node is at most $R_{\mathcal{A}}(F, k)$, the stated running time follows. $\qquad\square$

Our first tractability result is for Strong (2CNF $\cup\, \mathcal{C}$)-Backdoor Detection, where $\mathcal{C} \in$ Schaefer. Before formally stating the algorithm, we provide a brief intuition for the branching rules. Typically, we branch on an obstructing clause (or a pair of obstructing clauses), a situation that occurs in one of two flavors:

(1) Either there is a clause C that is not in 2CNF $\cup\, \mathcal{C}$, or,

(2) there is a pair of clauses C and C', where $C \in$ 2CNF $\setminus\, \mathcal{C}$ and $C' \in \mathcal{C} \setminus$ 2CNF.

Consider the case when $\mathcal{C} = 0$-VAL. In scenario (1), the formula has a clause C of length at least three with only positive literals. Because C has only positive literals, it follows that no subclause of C is 0-valid. Hence every backdoor set has to hit all but at most two literals from C. It follows that if C has more than $k + 2$ literals, then there is no strong backdoor of size k to 0-VAL $\cup$ 2CNF. Otherwise, the clause has at most $k + 2$ literals on which we can branch exhaustively. In scenario (2), we know that C has at most two literals and because $C \notin 0$-VAL we obtain that any strong backdoor set that does not contain a variable from C must reduce C' to a clause with at most two literals. Hence, any strong backdoor set has to contain either a variable from C (but there are at most two of those) or must contain a variable from any subset of three variables in C'.

Consider the case when $\mathcal{C} = $ HORN. In scenario (1) above, the formula has a clause C of length at least three with at least two positive literals, say x and y. Let ℓ be any other literal in C. It is clear that any strong backdoor set must contain one of x, y or ℓ, and we branch accordingly. In scenario (2), we know that C has at most two literals and C' contains at least two positive literals. Because every strong backdoor set that does not contain a variable from C must reduce C' to a clause with at most two literals, we can branch on the variables in C and any subset of three variables in C'.

The intuitive explanations for 1-VAL and HORN$^-$ are analogous. Note that the criteria described above need to be refined further at internal nodes of the branching process. We now turn to a formal exposition of the algorithm.

Theorem 2. *Let $\mathcal{C} \in$ Schaefer. Then,* Strong (2CNF $\cup\, \mathcal{C}$)-Backdoor Detection *is fixed-parameter tractable parameterized by the size of the backdoor set.*

Proof. Let $\mathcal{H} = $ 2CNF $\cup\, \mathcal{C}$. We will provide an algorithm $\mathcal{A}$ solving $\mathcal{H}$-Strong Branch in time $O(2^k|F|)$ and satisfying $T_{\mathcal{A}}(k) = 9^k$. The result then follows from Theorem 1.

Let F be the given CNF formula with variables V, k the given integer, and $B' \subseteq V$ with $|B'| \leq k$. The algorithm $\mathcal{A}$ first checks whether $F[\tau] \in \mathcal{H}$ for every assignment $\tau : B' \to \{0, 1\}$. If that is the case then $\mathcal{A}$ returns Yes. Otherwise let $\tau : B' \to \{0, 1\}$ be an assignment such that $F[\tau] \notin \mathcal{H}$, we consider two cases:

1. if $|B'| = k$, then there is no strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ and the algorithm $\mathcal{A}$ correctly returns the set $\mathcal{Q} = \emptyset$,

2. otherwise, i.e., if $|B'| < k$, then we again distinguish two cases:

   (a) there is a clause $C \in F[\tau]$ with $C \notin \mathcal{H}$, then we again distinguish two cases:

      i. if $\mathcal{C} \in \{\text{HORN}, \text{HORN}^-\}$, then let O be a subset of C with $|O| = 3$ and $O \notin \mathcal{C}$. Then $\mathcal{A}$ returns the set $\mathcal{Q} = \{\{v\} : v \in \text{var}(O)\}$.

      To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $B \cap O \neq \emptyset$. Suppose for a contradiction that this is not the case and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C. Then, $F[\tau^*]$ contains a superset of O as a clause and hence $F[\tau^*] \notin 2\text{CNF} \cup \mathcal{C}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set of F.

      ii. if $\mathcal{C} \in \{1\text{-VAL}, 0\text{-VAL}\}$, then we again distinguish two cases:
         A. $|B'| + |C| - 2 > k$, then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \emptyset$.

         The correctness of this step follows from the fact that every strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ has to contain all but at most two variables of C and hence B would have size larger than k.

         B. otherwise, i.e., if $|B| + |C| - 2 \leq k$, then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q}$ containing all sets $\text{var}(C \setminus O)$ for every two element subset O of C.

         To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It remains to show that $|\text{var}(C) \setminus B| \leq 2$. Assume not, and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C. Then, $C[\tau^*] \in F[\tau^*]$ and because $|C[\tau^*]| > 2$ and $C[\tau^*] \notin \mathcal{C}$, also $F[\tau^*] \notin 2\text{CNF} \cup \mathcal{C}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set of F.

   (b) otherwise, there is a clause C in $F[\tau]$ with $C \in 2\text{CNF} \setminus \mathcal{C}$ and a clause C' in $F[\tau]$ with $C' \in \mathcal{C} \setminus 2\text{CNF}$. Let O be the set of variables containing the (at most) two variables from C plus any three element subset of variables from C'. Then, the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \{\{v\} : v \in O\}$.

      To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It remains to show that $B \cap \text{var}(O) \neq \emptyset$. Assume not, and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C'. Then, $F[\tau^*]$ contains the clause C together with a subclause C'' of C' containing at least three literals. Because $C \notin \mathcal{C}$ and $C'' \notin 2\text{CNF}$, we obtain that $F[\tau^*] \notin 2\text{CNF} \cup \mathcal{C}$.

This completes the description and the proof of correctness of the algorithm $\mathcal{A}$. The running time of $\mathcal{A}$ is the number of assignments of the at most k variables in B' times the time required to test whether the reduced formula is in $\mathcal{H}$, i.e., $\mathcal{A}$ runs in time $O(2^k |F|)$. It remains to obtaine the stated bound of 9^k for the function $T_{\mathcal{A}}(k)$. According to the three branching rules given in (2.a.i), (2.a.ii.B), and (2.b) we obtain that $T_{\mathcal{A}}(k)$ can be bounded by the maximum of the following recurrence relations: (1) $T_{\mathcal{A}}(k) = 3T_{\mathcal{A}}(k-1)$ (2.a.i), (2) $T_{\mathcal{A}}(k) = r^2 T_{\mathcal{A}}(k - (r - 2))$ for any $r > 2$ (2.a.ii.B), and (3) $T_{\mathcal{A}}(k) = 5T_{\mathcal{A}}(k-1)$ (2.b). In case of (1) $T_{\mathcal{A}}(k)$ is at most 3^k, in case of (2) the maximum of 9^k is obtained for $T_{\mathcal{A}}(k)$ by setting $r = 3$, and in case of (3) $T_{\mathcal{A}}(k)$ is at most 5^k. It follows that $T_{\mathcal{A}}(k) \leq 9^k$. $\qquad \square$

We now turn to the case when the base classes are either $\text{HORN} \cup 0\text{-VAL}$ or $\text{HORN}^- \cup 1\text{-VAL}$.

As before, in scenario (1), the length of an obstructing clause must be bounded in a YES-instance. Indeed, consider a clause C that is not in $\text{HORN} \cup 0\text{-VAL}$. Such a clause has only positive literals, and any strong backdoor set must contain all these literals but one. Therefore, if the clause has more than $k + 1$ literals, we reject the instance, otherwise the clause length is bounded, allowing for a bounded branching strategy.

In scenario (2), on the other hand, we have a pair of obstructing clauses, say $C \in \text{HORN} \setminus 0\text{-VAL}$, and $C' \in 0\text{-VAL} \setminus \text{HORN}$. Now, note that C can have only one literal by definition. Let O be any subset of two positive literals from C'. Note that any strong backdoor must intersect either O or the unique literal in C, which again leads us to a feasible branching step. We now turn to a formal description of the algorithm.

Theorem 3. STRONG (HORN $\cup$ 0-VAL)-BACKDOOR DETECTION *and* STRONG (HORN$^-$ $\cup$ 1-VAL)-BACKDOOR DETECTION *are fixed-parameter tractable parameterized by the size of the backdoor set.*

Proof. We only give the proof for the class HORN $\cup$ 0-VAL, as the proof for the class HORN$^-$ $\cup$ 1-VAL is analogous. Let $\mathcal{H} = \text{HORN} \cup \text{0-VAL}$. We will provide an algorithm $\mathcal{A}$ solving $\mathcal{H}$-STRONG BRANCH in time $O(2^k|F|)$ and satisfying $T_\mathcal{A}(k) = 3^k$. The result then follows from Theorem 1.

Let F be the given CNF formula with variables V, k the given integer, and $B' \subseteq V$ with $|B'| \leq k$. The algorithm $\mathcal{A}$ first checks whether $F[\tau] \in \mathcal{H}$ for every assignment $\tau : B' \to \{0,1\}$. If that is the case then $\mathcal{A}$ returns YES. Otherwise let $\tau : B' \to \{0,1\}$ be an assignment such that $F[\tau] \notin \mathcal{H}$, we consider two cases:

1. if $|B'| = k$, then there is no strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ and the algorithm $\mathcal{A}$ correctly returns the set $\mathcal{Q} = \emptyset$,

2. otherwise, i.e., if $|B'| < k$, then we again distinguish two cases:

   (a) there is a clause $C \in F[\tau]$ with $C \notin \mathcal{H}$, then we again distinguish two cases:

   i. $|B| + |C| - 1 > k$, then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \emptyset$.

   To show the correctness of this step it is sufficient to show that F does not have a strong $\mathcal{H}$-backdoor set B with $B' \subseteq B$ and $|B| \leq k$. Assume for a contradiction that this is not the case and let B be strong $\mathcal{H}$-backdoor set of F with $B' \subseteq B$ and $|B| \leq k$. Because $|B'| + |C| - 1 > k$ we obtain that B misses at least two literals of C. Let $\tau' : B \to \{0,1\}$ be any assignment that is compatible with τ and does not satisfy any literal from C. Then $F[\tau']$ contains a subclause of C that contains at least two literals. It follows that $F[\tau'] \notin \mathcal{H}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set for F.

   ii. otherwise, i.e., if $|B| + |C| - 1 \leq k$, then $\mathcal{A}$ returns the set $\mathcal{Q} = \{\, \text{var}(C) \setminus \{v\} : v \in \text{var}(C) \,\}$.

   To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $|\text{var}(C) \setminus B| \leq 1$. Assume not, and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C. Then, $C[\tau^*] \in F[\tau^*]$ and because $|C[\tau^*]| > 1$ and $C[\tau^*] \notin \text{0-VAL}$, also $F[\tau^*] \notin \text{HORN} \cup \text{0-VAL}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set of F.

   (b) otherwise, there is a clause C in $F[\tau]$ with $C \in \text{HORN} \setminus \text{0-VAL}$ and a clause C' in $F[\tau]$ with $C \in \text{0-VAL} \setminus \text{HORN}$. Let O be the set of variables containing the one variable from C plus any two variables corresponding to two positive literals from C'. Then $\mathcal{A}$ returns the set $\mathcal{Q} = \{\, \{v\} : v \in O \,\}$.

   To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $B \cap O \neq \emptyset$. Assume not, and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C'. Then, $F[\tau^*]$ contains the clause C together with a subclause C'' of C' containing at least two positive literals. Because $C \notin \text{0-VAL}$ and $C'' \notin \text{HORN}$, we obtain that $F[\tau^*] \notin \mathcal{H}$.

This completes the description and the proof of correctness of the algorithm $\mathcal{A}$. The running time of $\mathcal{A}$ is the number of assignments of the at most k variables in B' times the time required to test whether the reduced formula is in $\mathcal{H}$, i.e., $\mathcal{A}$ runs in time $O(2^k|F|)$. It remains to obtaine the stated bound of 3^k for the function $T_\mathcal{A}(k)$. According to the two branching rules given in (2.a.ii) and (2.b) we obtain that $T_\mathcal{A}(k)$ can be bounded by the maximum of the following recurrence relations: (1) $T_\mathcal{A}(k) = rT_\mathcal{A}(k - (r - 1))$ for any $r \geq 2$ (2.a.ii) and (2) $T_\mathcal{A}(k) = 3T_\mathcal{A}(k-1)$ (2.b). In case of (1) the maximum of 2^k is obtained for $T_\mathcal{A}(k)$ by setting $r = 2$, and in case of (2) $T_\mathcal{A}(k)$ is at most 3^k. It follows that $T_\mathcal{A}(k) \leq 3^k$. $\qquad\square$

In our next result, we consider heterogeneous base classes comprised of three Schaefer's classes, namely 2CNF $\cup$ HORN $\cup$ 0-VAL and 2CNF $\cup$ HORN$^-$ $\cup$ 1-VAL. We refer the reader to Figure 1 for an overview of the branching strategies employed here.

Theorem 4. Strong (2CNF∪HORN∪0-VAL)-Backdoor Detection *and* Strong (2CNF∪HORN⁻∪ 1-VAL)-Backdoor Detection *are fixed-parameter tractable parameterized by the size of the backdoor set.*

Proof. We only give the proof for the class $2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$, since the proof for the class $2\text{CNF} \cup \text{HORN}^- \cup 1\text{-VAL}$ is analogous. Let $\mathcal{H} = 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$. We will provide an algorithm $\mathcal{A}$ solving $\mathcal{H}$-Strong Branch in time $O(2^k|F|)$ and satisfying $T_{\mathcal{A}}(k) = 9^k$. The result then follows from Theorem 1.

Let F be the given CNF formula with variables V, k the given integer, and $B' \subseteq V$ with $|B'| \leq k$. The algorithm $\mathcal{A}$ first checks whether $F[\tau] \in \mathcal{H}$ for every assignment $\tau : B' \to \{0,1\}$. If that is the case then $\mathcal{A}$ returns Yes. Otherwise let $\tau : B' \to \{0,1\}$ be an assignment such that $F[\tau] \notin \mathcal{H}$, we consider two cases:

1. if $|B'| = k$, then there is no strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ and the algorithm $\mathcal{A}$ correctly returns the set $\mathcal{Q} = \emptyset$,

2. otherwise, i.e., if $|B'| < k$, then we again distinguish two cases:

   (a) there is a clause $C \in F[\tau]$ with $C \notin \mathcal{H}$, then we again distinguish two cases:

      i. $|B'| + |C| - 2 > k$, then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \emptyset$.

         To show the correctness of this step it is sufficient to show that F does not have a strong $\mathcal{H}$-backdoor set B with $B' \subseteq B$ and $|B| \leq k$. Assume for a contradiction that this is not the case and let B be strong $\mathcal{H}$-backdoor set of F with $B' \subseteq B$ and $|B| \leq k$. Because $|B'| + |C| - 2 > k$, we obtain that B misses at least three literals from C. Hence, there is an assignment $\tau' : B \to \{0,1\}$ that is compatible with τ such that $F[\tau']$ contains a clause $C' \subseteq C$ with $|C'| > 2$. Clearly, $C' \notin 2\text{CNF} \cup 0\text{-VAL}$ and because C contained only positive literals (because $C \notin 0\text{-VAL}$) also $C' \notin \text{HORN}$. Hence, $C' \notin 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$ contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set for F.

      ii. otherwise, i.e., if $|B| + |C| - 2 \leq k$, then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \{\, \text{var}(C \setminus O) : O \subseteq \text{var}(C) \wedge |O| = 2 \,\}$.

         To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $|\text{var}(C) \setminus B| \leq 2$. Assume not, and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ that does not satisfy any literal of C. Then, $C[\tau^*] \in F[\tau^*]$ and because $|C[\tau^*]| > 2$ and $C[\tau^*]$ contains only positive literals, also $F[\tau^*] \notin 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set for F.

   (b) there is a clause $C \in F[\tau]$ with $C \in 2\text{CNF}$ and $C \notin 0\text{-VAL}$, we again distinguish two cases:

      i. there is a clause $C' \in F[\tau]$ with $C' \notin 2\text{CNF} \cup \text{HORN}$ (note that $C' \in 0\text{-VAL}$). Let O be a set of variables containing all variables of C (at most two) plus three variables of C' of which two correspond to positive literals in C'. Then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \{\, \{v\} : v \in O \,\}$.

         To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $B \cap O \neq \emptyset$. Assume not and let $\tau^* : B \to \{0,1\}$ be an assignment compatible with τ, which does not satisfy any literal of C'. Then, $F[\tau^*]$ contains C as well as a subclause C'' of C' of length at least three, which contains at least two positive literals. Then, $C'' \notin \text{HORN} \cup KROM$ and because $C \notin 0\text{-VAL}$, we obtain that $F[\tau^*] \notin 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set for F.

      ii. otherwise, there are clauses $C_{2\text{CNF}}, C_{\text{HORN}} \in F[\tau]$ with $C_{2\text{CNF}} \in 2\text{CNF} \setminus \text{HORN}$ and $C_{\text{HORN}} \in \text{HORN} \setminus 2\text{CNF}$. Let O be a set of variables containing all variables of $C_{2\text{CNF}}$ (at most two) and three variables of C_{HORN}. Then the algorithm $\mathcal{A}$ returns the set $\mathcal{Q} = \{\, \{v\} \,|\, v \in O \,\}$.

         To show the correctness of this step assume that there is a strong $\mathcal{H}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $B \cap O \neq \emptyset$. Assume not and let $\tau^* : B \to \{0,1\}$
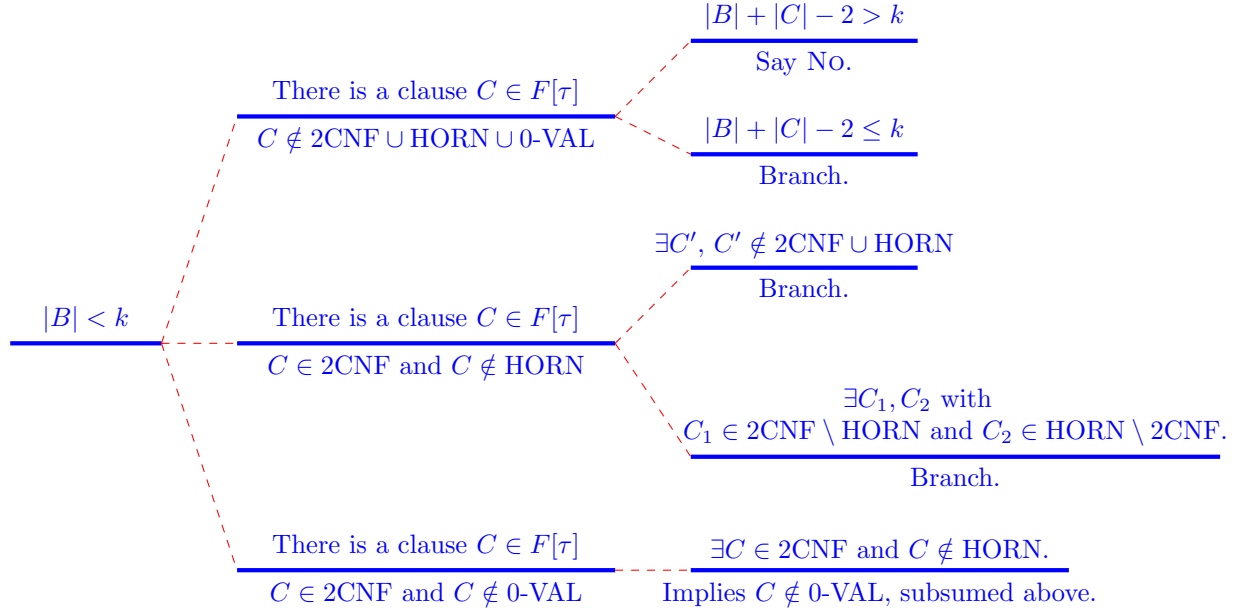
Figure 1: The branching cases for Theorem 4.

be an assignment compatible with τ, which does not satisfy any literal of C_{HORN}. Then, $F[\tau^*]$ contains C_{2CNF} as well as a subclause C' of C_{HORN} of length at least three. Because $C_{\text{2CNF}} \in 2\text{CNF} \setminus \text{HORN}$ it follows that $C_{\text{2CNF}} \notin 0\text{-VAL}$. Because also $C' \notin 2\text{CNF}$, we obtain that $F[\tau^*] \notin 2\text{CNF} \cup \text{HORN} \cup 0\text{-VAL}$, contradicting our assumption that B is a strong $\mathcal{H}$-backdoor set for F.

(c) there is a clause $C \in F[\tau]$ with $C \in 2\text{CNF}$ and $C \notin \text{HORN}$. In this case also $C \notin 0\text{-VAL}$ and hence the case is covered by case (2.b).

(d) The cases (2.a), (2.b), and (2.c) completely cover all possible cases, because if none of these cases apply then $F[\tau] \in 0\text{-VAL}$, which contradicts the choice of τ. To see this assume there is a clause $C \in F[\tau]$ with $C \notin 0\text{-VAL}$. Because of the cases (2.b) and (2.c), we obtain that $C \notin 2\text{CNF}$. Furthermore, because of case (2.a), we obtain that $C \in \text{HORN}$. Hence, $C \in \text{HORN} \setminus (0\text{-VAL} \cup 2\text{CNF})$ but such a clause cannot exist.

This completes the description and the proof of correctness of the algorithm $\mathcal{A}$. The running time of $\mathcal{A}$ is the number of assignments of the at most k variables in B' times the time required to test whether the reduced formula is in $\mathcal{H}$, i.e., $\mathcal{A}$ runs in time $O(2^k|F|)$. It remains to obtain the stated bound of 9^k for the function $T_{\mathcal{A}}(k)$.

According to the three branching rules given in (2.a.ii), (2.b.i), and (2.b.ii) we obtain that $T_{\mathcal{A}}(k)$ can be bounded by the maximum of the following recurrence relations: (1) $T_{\mathcal{A}}(k) = r^2 T_{\mathcal{A}}(k - (r-2))$ for any $r > 2$ (2.a.ii), (2) $T_{\mathcal{A}}(k) = 5T_{\mathcal{A}}(k-1)$ (2.b.i), and (3) $T_{\mathcal{A}}(k) = 5T_{\mathcal{A}}(k-1)$ (2.b.ii). In case of (2) and (3) $T_{\mathcal{A}}(k)$ is at most 5^k and in case of (1) the maximum of 9^k is obtained for $T_{\mathcal{A}}(k)$ by setting $r = 3$. It follows that $T_{\mathcal{A}}(k) \leq 9^k$.

□

We say that a pair $(\mathcal{C}, \mathcal{C}')$ of Schaefer classes is a *bad pair* if $\mathcal{C} \in \{\text{HORN}, 0\text{-VAL}\}$ and $\mathcal{C}' \in \{\text{HORN}^-, 1\text{-VAL}\}$. Our next result establishes hardness for the case when the base class contains a bad pair.

Theorem 5. *For every $S \subseteq$ Schaefer that contains at least one bad pair of Schaefer classes, it holds that* Strong $\mathcal{C}$-Backdoor Detection *is* W[2]*-hard parameterized by the size of the backdoor set, where* $\mathcal{C} = \bigcup_{s \in S} s$.

11

Proof. We give a parameterized reduction from the W[2]-complete HITTING SET problem. Given an instance $(\mathcal{Q}, U, k)$ for HITTING SET, construct a formula F as follows. The variables of F are $\mathcal{Q} \cup \{ d_Q^1, \ldots, d_Q^{|U \setminus Q|} : Q \in \mathcal{Q} \}$. For each set $Q \in \mathcal{Q}$, there is one clause $c_Q = Q \cup \{ d_Q^1, \ldots, d_Q^{|U \setminus Q|} \}$. There is also one clause $c_U = \{ \neg u : u \in U \}$. This completes the description of the reduction.

We claim that $\mathcal{Q}$ has a hitting set of size at most k if and only if the formula F has a strong $\mathcal{C}$-backdoor set of size at most k. Suppose $X \subseteq U$, $|X| \leq k$, is a hitting set. To show that X is also a strong $\mathcal{C}$-backdoor set, consider any assignment $\tau : X \to \{0, 1\}$. If $\tau(x) = 0$ for some $x \in X$, then τ satisfies the clause c_U. Thus, $F[\tau] \in \mathcal{C}$ since each clause in $F[\tau]$ contains no negative literal and at least one positive literal and is hence in 1-VAL $\cap$ HORN$^-$. If $\tau(x) = 1$ for all $x \in X$, then all clauses $c_Q, Q \in \mathcal{Q}$, are satisfied by τ since X is a hitting set. The only remaining clause is HORN and 0-VAL since it has no positive literal and at least one negative literal.

For the other direction, suppose that X is a strong $\mathcal{C}$-backdoor set of size at most k. Obtain X' from X by replacing each $d_Q^i \in X$ for any i with $1 \leq i \leq |U \setminus Q|$ by some variable from Q.

The set X' is also a strong $\mathcal{C}$-backdoor set of size at most k. Therefore, the assignment $\tau : X' \to \{0, 1\}$ with $\tau(x) = 1$ for all $x \in X'$ must satisfy all clauses $c_Q, Q \in \mathcal{Q}$. Otherwise, $F[\tau]$ contains both a long clause containing only positive and a long clause containing only negative literals and hence $F[\tau] \notin \bigcup_{s \in S} s$. Thus, X' is a hitting set for $\mathcal{Q}$ of size at most k. $\qquad\square$

We now summarize our results in the following theorem.

Theorem 6. *Let $S \subseteq$ Schaefer and $\mathcal{C} = \bigcup_{s \in S} s$. Then, STRONG $\mathcal{C}$-BACKDOOR DETECTION is fixed parameter tractable parameterized by the size of the backdoor set if S does not contain a bad pair of Schaefer classes, otherwise it is W[2]-hard.*

Proof. If S contains a bad pair then the result follows from Theorem 5. So assume that S does not contain a bad pair, we distinguish the following cases:

- if $|S| = 1$, then the theorem follows from known results, see, for example, [17, 18].

- if $|S| = 2$, then either 2CNF $\in S$ in which case the result follows from Theorem 2, or $S = \{\text{HORN}, \text{0-VAL}\}$ or $S = \{\text{HORN}^-, \text{1-VAL}\}$ in which case the result follows from Theorem 3.

- if $|S| = 3$, then either 2CNF $\in S$ and hence $S = \{\text{2CNF}, \text{HORN}, \text{0-VAL}\}$ or $S = \{\text{2CNF}, \text{HORN}^-, \text{1-VAL}\}$ in which case the result follows from Theorem 4, or 2CNF $\notin S$ in which case S contains at least one bad pair, and the result follows from Theorem 5.

- if $|S| > 3$, then S contains at least one bad pair, and the result follows from Theorem 5.

$\qquad\square$

It is crucial for these hardness proofs that clauses have unbounded length. Indeed, if clause-lengths are bounded or if we add the maximum clause length to the parameter, then strong backdoor detection becomes FPT for any combination of Schaefer classes.

Theorem 7. *Let $\mathcal{C}$ be a class of CNF-formulas such that $\mathcal{C} = \bigcup_{s \in S} s$ for some $S \subseteq$ Schaefer. Then, STRONG $\mathcal{C}$-BACKDOOR DETECTION is fixed-parameter tractable for the combined parameter k and the maximum length r of a clause of the input formula.*

Proof. We will provide an algorithm $\mathcal{A}$ solving $\mathcal{C}$-STRONG BRANCH in time $O(2^k |F|)$ and satisfying $T_{\mathcal{A}}(k) = (|S| r)^{k+1}$. The result then follows from Theorem 1.

Let F be the given CNF formula with variables V, k the given integer, and $B' \subseteq V$ with $|B'| \leq k$. The algorithm $\mathcal{A}$ first checks whether $F[\tau] \in \mathcal{C}$ for every assignment $\tau : B' \to \{0, 1\}$. If that is the case then $\mathcal{A}$ returns YES. Otherwise let $\tau : B' \to \{0, 1\}$ be an assignment such that $F[\tau] \notin \mathcal{C}$, we consider two cases:

1. if $|B'| = k$, then there is no strong $\mathcal{C}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$ and the algorithm $\mathcal{A}$ correctly returns the set $\mathcal{Q} = \emptyset$.

12

2. otherwise, i.e., if $|B'| < k$, then let C_s be an arbitrary clause in $F[\tau]$ with $C_s \notin s$ for every $s \in S$. The algorithm $\mathcal{A}$ then returns the set $\mathcal{Q} = \bigcup_{s \in S}\{\,\{v\} : v \in \mathrm{var}(C_s)\,\}$.

To show the correctness of this step assume that there is a strong $\mathcal{C}$-backdoor set B of F with $B' \subseteq B$ and $|B| \leq k$. It suffices to show that $B \cap \bigcup_{s \in S} \mathrm{var}(C) \neq \emptyset$. Let $\tau^* : B \to \{0,1\}$ be any assignment of the variables in B that agrees with τ on the variables in B' and let $s \in S$ be such that $F[\tau^*] \in s$. Because B is a strong $\mathcal{C}$-backdoor set, s clearly exists. We claim that $\mathrm{var}(C) \cap B \neq \emptyset$ for every clause $C \in F[\tau]$ with $C \notin S$ (and hence in particular for the clause C_s chosen by $\mathcal{A}$). Indeed, suppose not. Then $C \in F[\tau^*]$ a contradiction to our assumption that $F[\tau^*] \in s$.

This completes the description and the proof of correctness of the algorithm $\mathcal{A}$. The running time of $\mathcal{A}$ is the number of assignments of the at most k variables in B' times the time required to test whether the reduced formula is in $\mathcal{C}$, i.e., $\mathcal{A}$ runs in time $O(2^k|F|)$. It remains to obtain the stated bound of $(|S|r)^{k+1}$ for the function $T_{\mathcal{A}}(k)$, which follows because $|\mathcal{Q}| \leq |S|r$ and $|Q| = 1$ for every $Q \in \mathcal{Q}$. $\qquad\square$

3.2. Weak Backdoor Sets

It turns out that the complexity of finding weak backdoors into heterogeneous classes is tightly tied to the complexity of finding weak backdoors into the individual base classes. Let us briefly consider the intuition for this scenario before stating the result. Suppose we are given algorithms for finding weak backdoors into base classes $\mathcal{C}_1$ and $\mathcal{C}_2$. Observe that a CNF-formula has a weak backdoor of size at most k into $(\mathcal{C}_1 \cup \mathcal{C}_2)$ if, and only if, it has a weak backdoor of size at most k into at least one of $\mathcal{C}_1$ or $\mathcal{C}_2$. Thus, if we have a positive algorithmic result for the weak backdoor question with respect to the individual base classes, then these algorithms can be easily combined to obtain an algorithm for finding weak backdoors into the corresponding heterogeneous base class. On the other hand, a hardness result for even one of the involved classes can usually be leveraged to obtain a hardness result for the entire heterogeneous problem, by adding gadgets that isolate the issue to only one of the base classes, typically the one that was hard to begin with. In particular, we obtain the following result.

Theorem 8. *Let $S \subseteq$ Schaefer and $\mathcal{C} = \bigcup_{s \in S} s$. Then,* WEAK $\mathcal{C}$-BACKDOOR DETECTION *is* W[2]-*hard parameterized by the size of the backdoor set.*

Before we proceed further, we introduce the notion of an obstruction which we will use in the proof. Specifically, for $S, S' \in$ Schaefer, we say that a clause C is an S-obstruction to S' if $C \in S$ and $C \notin S'$. For any pair of Schaefer classes S and S', it is easy to construct examples of S-obstructions to S'. In particular, we have:

1. $(\neg x, \neg y, \neg z)$ is a HORN-obstruction to 2CNF, and also a 0-VAL-obstruction to 2CNF.

2. (x, y, z) is an HORN$^-$-obstruction to 2CNF, and also a 1-VAL-obstruction to 2CNF.

3. $(x, y, \neg z)$ is a 0-VAL-obstruction to HORN.

4. (x, y) is an X-obstruction to HORN for all $X \in \{$2CNF, HORN$^-$, 1-VAL$\}$.

5. $(\neg x, \neg y, z)$ is a 1-VAL-obstruction to HORN$^-$.

6. $(\neg x, \neg y)$ is an X-obstruction to HORN$^-$ for all $X \in \{$2CNF, HORN, 0-VAL$\}$.

7. (x) is an X-obstruction to 1-VAL for all $X \in \{$2CNF, 1-VAL, HORN, HORN$^-\}$.

8. $(\neg x)$ is an X-obstruction to 1-VAL for all $X \in \{$2CNF, 1-VAL, HORN, HORN$^-\}$.

We are now ready to describe our reduction towards Theorem 8.

13

Proof. If $|S| = 1$, then the result follows directly from [18, Proposition 1]. Otherwise, let $S \in \mathcal{C}$. We reduce an instance (F, k) of WEAK S-BACKDOOR DETECTION to WEAK $\mathcal{C}$-BACKDOOR DETECTION. To this end, for all $S' \in \mathcal{C}$ such that $S \neq S'$, introduce $(k+1)$ variable-disjoint copies of a S-obstruction to S' using fresh variables for each obstruction, and in particular, disjoint from $\text{var}(F)$. We use F' to denote the formula obtained from F after adding these obstructions to F, and the reduced instance is given by (F', k).

In the forward direction, if X is a weak backdoor to S for F, and if $\tau : X \to \{0, 1\}$ is such that $F[\tau] \in S$, then observe that $F'[\tau] \in S$, since all the added obstructions (which are unaffected by τ) were in S by definition. It is also easy to see that any satisfying assignment for $\text{var}(F) \setminus X$ can be easily extended to an assignment involving the new variables that satisfy all the obstructions (recall that the obstructions were variable disjoint, and therefore can be easily satisfied independently).

On the other hand, let X be a weak backdoor to $\mathcal{C}$ for F'. Since X has at most k variables, for any $\tau : X \to \{0, 1\}$, $F'[\tau]$ contains at least one S'-obstructing clause for each $S' \in \mathcal{C}$ such that $S' \neq S$. Therefore, $F'[\tau]$ must necessarily belong to S, and therefore $X \cap \text{var}(F)$ is easily checked to be a weak backdoor to S for F, as desired. $\qquad\square$

On the other hand, as with the strong backdoors, we do obtain tractability when we have bounded clause lengths. We establish this formally below.

Theorem 9. *Let $\mathcal{C}$ be a class of CNF-formulas such that $\mathcal{C} = \bigcup_{s \in S} s$ for some $S \subseteq$ Schaefer. Then, WEAK $\mathcal{C}$-BACKDOOR DETECTION is fixed-parameter tractable for the combined parameter k and the maximum length r of a clause of the input formula.*

Proof. The algorithm uses a depth-bounded search tree approach to find a weak $\mathcal{C}$-backdoor set of size at most k. Let F be any CNF-formula with variables V and let r be the maximum length of any clause of F.

We construct a search tree T, for which every node is labeled by a set B of at most k variables of V. Additionally, every leaf node has a second label, which is either YES or NO. T is defined inductively as follows. The root of T is labeled by the empty set. Furthermore, if t is a node of T, whose first label is B, the children of t in T are obtained as follows.

If there is an assignment $\tau : B \to \{0, 1\}$, such that $F[\tau] \in \mathcal{C}$ and $F[\tau]$ is satisfiable, then B is a weak $\mathcal{C}$-backdoor set of size at most k, and hence t becomes a leaf node, whose second label is YES. Otherwise, i.e., for every assignment $\tau : B \to \{0, 1\}$ either $F[\tau] \notin \mathcal{C}$ or $F[\tau] \in \mathcal{C}$ but $F[\tau]$ is not satisfiable, we consider two cases: (1) $|B| = k$, then t becomes a leaf node, whose second label is NO, and (2) $|B| < k$, then for every assignment $\tau : B \to \{0, 1\}$ such that $F[\tau] \notin \mathcal{C}$, for every $s \in S$, and every variable v that occurs in some clause $C \notin s$, t has a child whose first label is $B \cup \{v\}$.

If T has a leaf node, whose second label is YES, then the algorithm returns the first label of that leaf node. Otherwise the algorithm returns NO. This completes the description of the algorithm.

We now show the correctness of the algorithm. First, suppose the search tree T built by the algorithm has a leaf node t whose second label is YES. Here, the algorithm returns the first label, say B of t. By definition of T, we obtain that $|B| \leq k$ and the set B is a weak $\mathcal{C}$-backdoor set of F, as required.

Now consider the case where the algorithm returns NO. We need to show that there is no weak $\mathcal{C}$-backdoor set of size at most k for F. Assume, for the sake of contradiction that such a set B exists.

Observe that if T has a leaf node t whose first label is a set B' with $B' \subseteq B$, then the second label of t must be YES. This is because, either $|B'| < k$ in which case the second label of t must be YES, or $|B'| = k$ in which case $B' = B$ and by the definition of B it follows that the second label of t must be YES.

It hence remains to show that T has a leaf node whose first label is a set B' with $B' \subseteq B$. This will complete the proof about the correctness of the algorithm. We will show a slightly stronger statement, namely, that for every natural number ℓ, either T has a leaf whose first label is contained in B or T has an inner node of distance exactly ℓ from the root whose first label is contained in B. We show the latter by induction on ℓ.

The claim obviously holds for $\ell = 0$. So assume that T contains a node t at distance ℓ from the root of T whose first label, say B', is a subset of B. If t is a leaf node of T, then the claim is shown. Because B is a weak $\mathcal{C}$-backdoor set for F, there is an assignment $\tau^* : B \to \{0, 1\}$ such that $F[\tau^*] \in \mathcal{C}$ and $F[\tau^*]$ is satisfiable. In particular, there is some $s \in S$ such that $F[\tau^*] \in s$. Let $\tau : B' \to \{0, 1\}$ be the assignment

that agrees with τ^* on the variables of B'. Because t is not a leaf node, there is a clause $C \in F[\tau]$ with $C \notin s$ such that t has a child t' for every $v \in \text{var}(C)$. We claim that $\text{var}(C) \cap B \neq \emptyset$ and hence t has a child, whose first label is a subset of B, as required. Indeed, suppose not. Then $C \in F[\tau^*]$ a contradiction to our assumption that $F[\tau^*] \in s$. This concludes our proof concerning the correctness of the algorithm.

The running time of the algorithm is obtained as follows. Let T be a search tree obtained by the algorithm. Then the running time of the depth-bounded search tree algorithm is $O(|V(T)|)$ times the maximum time that is spent on any node of T. Since the number of children of any node of T is bounded by $2^k|S|r$ (recall that r denotes the maximum length of any clause of F) and the longest path from the root of T to some leaf of T is bounded by $k + 1$, we obtain that $|V(T)| \leq O((2^k|S|r)^{k+1})$. Furthermore, the time required for any node t of T is at most $O(2^k|F|)$. Putting everything together, we obtain $O((2^k|S|r)^{k+1}2^k|F|)$, as the total running time of the algorithm. This shows that WEAK $\mathcal{C}$-BACKDOOR DETECTION is fixed-parameter tractable parameterized by k, r. $\qquad\square$

We close this section by noting that backdoor sets with *empty clause detection*, as proposed by [5] can be considered as backdoor sets into the heterogeneous base class obtained by the union of a homogeneous base class $\mathcal{C}$ and the class of all formulas that contain the empty clause. The detection of strong backdoor sets with empty clause detection is not fixed-parameter tractable for many natural base classes, including Horn and 2CNF [29].

4. Base Classes via Closure Properties

In this section we provide a very general framework that will give rise to a wide range of heterogeneous base classes for CSP.

Given an r-ary relation R over some domain D and an n-ary operation $\phi : D^n \to D$, we say that R is *closed under* ϕ, if for all collections of n tuples $t_1, \ldots, t_n$ from R, the tuple $\langle \phi(t_1[1], \ldots, t_n[1]), \ldots, \phi(t_1[r], \ldots, t_n[r]) \rangle$ belongs to R. The operation ϕ is also said to be a *polymorphism of* R. We denote by $\text{Pol}(R)$ the set of all operations ϕ such that R is closed under ϕ.

Let $I = \langle V, D, C \rangle$ be a CSP instance and $c \in C$. We write $\text{Pol}(c)$ for the set $\text{Pol}(R(c))$ and we write $\text{Pol}(I)$ for the set $\bigcap_{c \in C} \text{Pol}(c)$. We say that I is closed under an operation ϕ, or ϕ is a polymorphism of I, if $\phi \in \text{Pol}(I)$.

We say an operation ϕ is *tractable* if every CSP instance closed under ϕ can be solved in polynomial time.

Let $\mathbb{P}(\phi)$ be a predicate for the operation ϕ. We call $\mathbb{P}(\phi)$ a *tractable polymorphism property* if the following conditions hold.

- There is a constant $c_{\mathbb{P}}$ such that for all finite domains D, all operations ϕ over D with property $\mathbb{P}$ are of arity at most $c_{\mathbb{P}}$.

- Given an operation ϕ and a domain D, one can check in polynomial time whether $\mathbb{P}(\phi)$ holds on all of the at most $D^{c_{\mathbb{P}}}$ tuples over D,

- Every operation with property $\mathbb{P}$ is tractable.

Every tractable polymorphism property $\mathbb{P}$ gives rise to a natural base class $\mathcal{C}_{\mathbb{P}}$ consisting of all CSP-instances I such that $\text{Pol}(I)$ contains some polymorphism ϕ with $\mathbb{P}(\phi)$.

In the following we will provide several illustrative examples for classes of CSP instances that can be defined via tractable polymorphism predicates. The chosen examples are mostly based on the Schaefer classes or important generalizations thereof. For the definition of the examples we need to define the following types of operations.

- An operation $\phi : D^n \to D$ is *idempotent* if for every $d \in D$ it holds that $\phi(d, \ldots, d) = d$;

- An operation $\phi : D \to D$ is *constant* if there is a $d \in D$ such that for every $d' \in D$, it holds that $\phi(d') = d$;

15

- An operation $\phi : D^2 \to D$ is a *min/max* operation if there is an ordering of the elements of D such that for every $d, d' \in D$, it holds that $\phi(d, d') = \phi(d', d) = \min\{d, d'\}$ or $\phi(d, d') = \phi(d', d) = \max\{d, d'\}$, respectively;

- An operation $\phi : D^3 \to D$ is a *majority* operation if for every $d, d' \in D$ it holds that $\phi(d, d, d') = \phi(d, d', d) = \phi(d', d, d) = d$;

- An operation $\phi : D^3 \to D$ is an *minority* operation if for every $d, d' \in D$ it holds that $\phi(d, d, d') = \phi(d, d', d) = \phi(d', d, d) = d'$;

- An operation $\phi : D^3 \to D$ is a *Mal'cev* operation if for every $d, d' \in D$ it holds that $\phi(d, d, d') = \phi(d', d, d) = d'$.

It is known that every constant, min/max, majority, minority, and Mal'cev operation is tractable [23, 20]. We denote by VAL, MINMAX, MAJ, MIN, and MAL the class of CSP instances I for which $\mathrm{Pol}(I)$ contains a constant, a min/max, a majority, a minority, or a Mal'cev polymorphism, respectively.

Thus VAL, MINMAX, MAJ, MIN, and MAL are the classes $\mathcal{C}_{\mathbb{P}}$ for $\mathbb{P} \in \{$constant, min/max, majority, minority, Mal'cev$\}$, respectively.

In terms of the above definitions we can state the results of [23] and [20] as follows.

Proposition 1. *Constant, min/max, majority, minority, and Mal'cev are tractable polymorphism properties.*

We want to note here that Proposition 1 also applies for other much more general types of operations such as semilattice operations (sometimes called ACI operation) [23] (a generalization of min/max), k-ary near unanimity operations [30, 31] (a generalization of majority), k-ary edge operations [32] (a generalization of Mal'cev), and the two operations of arities three and four [33] that capture the bounded width property [34] (a generalization of semilattice and near unanimity operations).

In the next sections we will study the problems STRONG $\mathcal{C}_{\mathbb{P}}$-BACKDOOR DETECTION for tractable polymorphism properties $\mathbb{P}$.

5. Tractability of Backdoor Detection for CSP

In this section we will show that both STRONG $\mathcal{C}_{\mathbb{P}}$-BACKDOOR DETECTION and WEAK $\mathcal{C}_{\mathbb{P}}$-BACKDOOR DETECTION parameterized by the size of the backdoor set, the size of the domain, and the maximum arity of the CSP instance are fixed-parameter tractable for any tractable polymorphism property $\mathbb{P}$. We start by giving the tractability results for strong backdoor sets.

Theorem 10. *Let $\mathbb{P}$ be a tractable polymorphism property. Then STRONG $\mathcal{C}_{\mathbb{P}}$-BACKDOOR DETECTION is fixed-parameter tractable for the combined parameter size of the backdoor set, size of the domain, and the maximum arity of the given CSP instance.*

Proof. Let $\mathbb{P}$ be a tractable polymorphism property, and let $\langle I, k \rangle$ with $I = \langle V, D, C \rangle$ be an instance of STRONG $\mathcal{C}_{\mathbb{P}}$-BACKDOOR DETECTION. Let P be the set of all operations on D that have property $\mathbb{P}$. Then, P can be constructed in fpt-time with respect to the size of the domain , because there are at most $|D|^{|D|^{c_{\mathbb{P}}}}$ $c_{\mathbb{P}}$-ary operations on D and for each of them we can test in polynomial time, $|D|^{O(c_{\mathbb{P}})}$, whether it satisfies property $\mathbb{P}$. The algorithm uses a depth-bounded search tree approach to find a strong $\mathcal{C}_{\mathbb{P}}$-backdoor set of size at most k.

We construct a search tree T, for which every node is labeled by a set B of at most k variables of V. Additionally, every leaf node has a second label, which is either YES or NO. T is defined inductively as follows. The root of T is labeled by the empty set. Furthermore, if t is a node of T, whose first label is B, then the children of t in T are obtained as follows. If for every assignment $\tau : B \to D$ there is an operation $\phi \in P$ such that $I[\tau]$ is closed under ϕ, then B is a strong P-backdoor set of size at most k, and hence t becomes a leaf node, whose second label is YES. Otherwise, i.e., if there is an assignment $\tau : B \to D$ such

16

that $I[\tau]$ is not closed under any operation $\phi \in P$, we consider two cases: (1) $|B| = k$, then t becomes a leaf node, whose second label is No, and (2) $|B| < k$, then for every operation $\phi \in P$ and every variable v in the scope of some constraint $c \in C[\tau]$ that is not closed under ϕ, t has a child whose first label is $B \cup \{v\}$.

If T has a leaf node, whose second label is Yes, then the algorithm returns the first label of that leaf node. Otherwise the algorithm returns No. This completes the description of the algorithm.

We now show the correctness of the algorithm. First, suppose the search tree T built by the algorithm has a leaf node t whose second label is Yes. Here, the algorithm returns the first label, say B of t. By definition, we obtain that $|B| \leq k$ and for every assignment $\tau : B \to D$, it holds that $I[\tau]$ is closed under some operation in P, as required.

Now consider the case where the algorithm returns No. We need to show that there is no set B of at most k variables of I such that $\text{Pol}(I[\tau]) \cap P \neq \emptyset$ for every assignment τ of the variables of B. Assume, for the sake of contradiction that such a set B exists.

Observe that if T has a leaf node t whose first label is a set B' with $B' \subseteq B$, then the second label of t must be Yes. This is because, either $|B'| < k$ in which case the second label of t must be Yes, or $|B'| = k$ in which case $B' = B$ and by the definition of B it follows that the second label of t must be Yes.

It hence remains to show that T has a leaf node whose first label is a set B' with $B' \subseteq B$. This will complete the proof about the correctness of the algorithm. We will show a slightly stronger statement, namely, that for every natural number ℓ, either T has a leaf whose first label is contained in B or T has an inner node of distance exactly ℓ from the root whose first label is contained in B. We show the latter by induction on ℓ.

The claim obviously holds for $\ell = 0$. So assume that T contains a node t at distance ℓ from the root of T whose first label, say B', is a subset of B. If t is a leaf node of T, then the claim is shown. Otherwise, there is an assignment $\tau : B' \to D$ such that $I[\tau]$ is not closed under any operation from P. Let $\tau^* : B \to D$ be any assignment of the variables in B that agrees with τ on the variables in B' and let $\phi \in P$ be such that $I[\tau^*]$ is closed under ϕ. Because B is a strong P-backdoor set, the polymorphism ϕ clearly exists. By definition of the search tree T, t has a child t' for every variable v in the scope of some constraint $c \in C[\tau]$ that is not closed under ϕ. We claim that $V(c) \cap B \neq \emptyset$ and hence t has a child, whose first label is a subset of B, as required. Indeed, suppose not. Then $c \in C[\tau^*]$ a contradiction to our assumption that $I[\tau^*]$ is closed under ϕ. This concludes our proof concerning the correctness of the algorithm.

The running time of the algorithm is obtained as follows. Let T be a search tree obtained by the algorithm. Then the running time of the depth-bounded search tree algorithm is $O(|V(T)|)$ times the maximum time that is spent on any node of T. Since the number of children of any node of T is bounded by $|P|\delta(I)$ (recall that $\delta(I)$ denotes the maximum arity of any constraint of I) and the longest path from the root of T to some leaf of T is bounded by $k+1$, we obtain that $|V(T)| \leq O((|P|\delta(I))^{k+1})$. Furthermore, the time required for any node t of T is at most $O(|D|^k \text{comp\_rest}(I, \tau)|C(I[\tau])||P|\text{check\_poly}(c, \phi))$, where $\text{comp\_rest}(I, \tau)$ is the time required to compute $I[\tau]$ for some assignment τ of at most k variables and $\text{check\_poly}(c, \phi)$ is the time required to check whether a constraint c of $I[\tau]$ preserves the operation $\phi \in P$. Observe that $\text{comp\_rest}(I, \tau)$ and $|C(I[\tau])|$ are polynomial in the input size. The same holds for $\text{check\_poly}(c, \phi)$, because ϕ is a $c_{\mathbb{P}}$-ary operation. Now, the total running time required by the algorithm is the time required to compute the set P plus the time required to compute T. Putting everything together, we obtain $O((|P|\delta(I))^{k+1}|D|^k|P|n^{O(1)}) = O((|P|\delta(I)|D|)^{k+2}n^{O(1)}) = O((|D|^{|D|^{c_{\mathbb{P}}}}\delta(I)|D|)^{k+2}n^{O(1)})$, r as the total running time of the algorithm, where n denotes the input size of the CSP instance. This shows that Strong $\mathcal{C}_{\mathbb{P}}$-Backdoor Detection is fixed-parameter tractable parameterized by k, $\delta(I)$, and $|D|$. $\square$

Because of Proposition 1, we obtain the following.

Corollary 1. *Let $\mathcal{C}$ be a base class consisting of the union of some of the classes* MINMAX, MAJ, MIN, *and* MAL. *Then* Strong $\mathcal{C}$-Backdoor Detection *is fixed-parameter tractable for the combined parameter size of the backdoor set, size of the domain, and the maximum arity of the given CSP instance.*

We are now ready to provide analogous results for weak backdoor sets. The proof is very similar to the case of strong backdoor sets, however, it becomes slightly simpler, because we do not need to consider

different operations for different assignments (since there is only one assignment, required for a weak backdoor set). We can hence branch on the possible operations before we start the depth-bounded search tree procedure, which also results in a slight improvement in the running time of the algorithm.

Theorem 11. *Let $\mathbb{P}$ be a tractable polymorphism property. Then* Weak $\mathcal{C}_{\mathbb{P}}$-Backdoor Detection *is fixed-parameter tractable for the combined parameter size of the backdoor set, size of the domain, and the maximum arity of the given CSP instance.*

Proof. Let $\mathbb{P}$ be a tractable property, and let $\langle I, k \rangle$ with $I = \langle V, D, C \rangle$ be an instance of Weak $\mathcal{C}_{\mathbb{P}}$-Backdoor Detection. Let P be the set of all operations on D that have property $\mathbb{P}$. Then, P can be constructed in fpt-time with respect to the size of the domain , because there are at most $|D|^{|D|^{c_{\mathbb{P}}}}$ $c_{\mathbb{P}}$-ary operations on D and for each of them we can test in polynomial time, $|D|^{O(c_{\mathbb{P}})}$, whether it satisfies property $\mathbb{P}$. The algorithm uses a depth-bounded search tree approach to find a weak $\{\phi\}$-backdoor set of size at most k for every $\phi \in P$.

The algorithm works by constructing a search tree T_ϕ for every operation $\phi \in P$. In T_ϕ every node is labeled by a set B of at most k variables of V. Additionally, every leaf node has a second label, which is either Yes or No. T_ϕ is defined inductively as follows. The root of T_ϕ is labeled by the empty set. Furthermore, if t is a node of T_ϕ, whose first label is B, then the children of t in T_ϕ are obtained as follows. If there is an assignment $\tau : B \to D$ such that $I[\tau]$ is closed under ϕ and $I[\tau]$ has a solution , then B is a weak P-backdoor set of size at most k, and hence t becomes a leaf node, whose second label is Yes. Otherwise, i.e., for every assignment $\tau : B \to D$ either $I[\tau]$ is not closed under ϕ or $I[\tau]$ has no solution, we consider two cases: (1) $|B| = k$, then t becomes a leaf node, whose second label is No, and (2) $|B| < k$, then for every assignment $\tau : B \to D$ t has the following children: for every every variable v in the scope of some constraint $c \in I[\tau]$ that is not closed under ϕ, t has a child whose first label is $B \cup \{v\}$.

If there is a $\phi \in P$ such T_ϕ has a leaf node, whose second label is Yes, then the algorithm returns the first label of that leaf node. Otherwise the algorithm returns No. This completes the description of the algorithm.

We now show the correctness of the algorithm. First, suppose there is a $\phi \in P$ such that the search tree T_ϕ built by the algorithm has a leaf node t whose second label is Yes. Here, the algorithm returns the first label, say B of t. By the construction of T_ϕ, we obtain that $|B| \leq k$ and there is an assignment $\tau : B \to D$ such that $I[\tau]$ is closed ϕ and $I[\tau]$ has a solution, as required.

Now consider the case where the algorithm returns No. We need to show that there is no weak P-backdoor set of size at most k for I. Assume, for the sake of contradiction that such a set B exists. Because B is a weak P-backdoor set for I there must exist an operation $\phi \in P$ and an assignment $\tau : B \to D$ such that $I[\tau]$ is closed under ϕ and $I[\tau]$ has a solution.

Observe that if T_ϕ has a leaf node t whose first label is a set B' with $B' \subseteq B$, then the second label of t must be Yes. This is because, either $|B'| < k$ in which case the second label of t must be Yes, or $|B'| = k$ in which case $B' = B$ and by the definition of B it follows that the second label of t must be Yes.

It hence remains to show that T_ϕ has a leaf node whose first label is a set B' with $B' \subseteq B$. This will complete the proof about the correctness of the algorithm. We will show a slightly stronger statement, namely, that for every natural number ℓ, either T_ϕ has a leaf whose first label is contained in B or T_ϕ has an inner node of distance exactly ℓ from the root whose first label is contained in B. We show the latter by induction on ℓ.

The claim obviously holds for $\ell = 0$. So assume that T_ϕ contains a node t at distance ℓ from the root of T_ϕ whose first label, say B', is a subset of B. If t is a leaf node of T_ϕ, then the claim is shown. Otherwise, for every assignment $\tau : B' \to D$ either $I[\tau]$ is not closed under ϕ or $I[\tau]$ has no solution. Let $\tau' : B' \to D$ be the restriction of τ to B'. Because $I[\tau]$ and hence also $I[\tau']$ has a solution, we obtain that $I[\tau']$ is not closed under the operation ϕ. By the definition of the search tree T_ϕ, it holds that t has a child t' whose first label is $B' \cup \{v\}$, for every variable v in the scope of some constraint $c \in I[\tau']$ that is not closed under ϕ. We claim that $V(c) \cap B \neq \emptyset$ and hence t has a child, whose first label is a subset of B, as required. Indeed, suppose not. Then $c \in I[\tau]$ a contradiction to our assumption that $I[\tau]$ is closed under ϕ. This concludes our proof concerning the correctness of the algorithm.

The running time of the algorithm is obtained as follows. Let T_ϕ be a search tree obtained by the algorithm. Then the running time of the depth-bounded search tree algorithm is $O(|V(T_\phi)|)$ times the maximum time that is spent on any node of T_ϕ. Since the number of children of any node of T_ϕ is bounded by $|D|^k \delta(I)$ (recall that $\delta(I)$ denotes the maximum arity of any constraint of I) and the longest path from the root of T_ϕ to some leaf of T_ϕ is bounded by $k+1$, we obtain that $|V(T_\phi)| \leq O((|D|^k \delta(I))^{k+1})$. Furthermore, the time required for any node t of T is at most $O(|D|^k \mathrm{comp\_rest}(I,\tau)|C(I[\tau])|\mathrm{check\_poly}(c,\phi)\mathrm{check\_sol}(I[\tau],\phi))$, where $\mathrm{comp\_rest}(I,\tau)$ is the time required to compute $I[\tau]$ for some assignment τ of at most k variables, $\mathrm{check\_poly}(c,\phi)$ is the time required to check whether a constraint c of $I[\tau]$ preserves the polymorphism ϕ, and $\mathrm{check\_sol}(I[\tau],\phi)$ is the time required to solve $I[\tau]$ given that it is closed under the operation ϕ. Observe that $\mathrm{comp\_rest}(I,\tau)$ and $|C(I[\tau])|$ are polynomial in the input size. The same holds for $\mathrm{check\_poly}(c,\phi)$ and $\mathrm{check\_sol}(I[\tau],\phi)$, because ϕ is a $c_\mathbb{P}$-ary tractable polymorphism. Now, the total running time required by the algorithm is the time required to compute the set P plus the time required to compute T_ϕ for every $\phi \in P$. Putting everything together, we obtain $O(|P|(|D|^k\delta(I))^{k+1}|D|^k n^{O(1)}) = O(|P|(|D|^k\delta(I))^{k+1}n^{O(1)}) = O(|D|^{|D|^{c_\mathbb{P}}}(|D|^k\delta(I))^{k+2}n^{O(1)})$, as the total running time of the algorithm, where n denotes the input size of the CSP instance. This shows that WEAK $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION is fixed-parameter tractable parameterized by k, $\delta(I)$, and $|D|$. $\qquad\square$

Because of Proposition 1, we obtain the following.

Corollary 2. *Let $\mathcal{C}$ be a base class consisting of the union of some of the classes MINMAX, MAJ, MIN, and MAL. Then WEAK $\mathcal{C}$-BACKDOOR DETECTION is fixed-parameter tractable for the combined parameter size of the backdoor set, size of the domain, and the maximum arity of the given CSP instance.*

6. Hardness of Backdoor Detection for CSP

In this section we show our parameterized hardness results for STRONG $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION and WEAK $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION. In particular, we show that STRONG $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION and WEAK $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION are W[2]-hard parameterized by the size of the backdoor set even for CSP instances of Boolean domain and for CSP instances with arity two. We start by showing hardness for CSP instances with Boolean domain.

Theorem 12. *Let $\mathbb{P}$ be a tractable polymorphism property such that all operations ϕ with $\mathbb{P}(\phi)$ are idempotent. Then, STRONG $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION and WEAK $\mathcal{C}_\mathbb{P}$-BACKDOOR DETECTION are W[2]-hard parameterized by the size of the backdoor set, even for CSP instances over the Boolean domain.*

Proof. We start by introducing what we call "Boolean barriers" of operations since they form the basis of the proof. Let $\phi : D^n \to D$ be an n-ary operation over D. We say a set λ of $r(\lambda)$-ary tuples over $\{0,1\}$ is a *Boolean barrier* for ϕ if there is a sequence $\langle t_1, \ldots, t_n \rangle$ of (not necessarily distinct) tuples in λ such that $\phi(t_1, \ldots, t_n) \notin \lambda$. We call a Boolean barrier λ of ϕ *minimal* if $|\lambda|$ is minimal over all Boolean barriers of ϕ. For an operation ϕ, we denote by $\lambda(\phi)$ a minimal Boolean barrier of ϕ. For our reduction below, we will employ the fact that every operation ϕ with $\mathbb{P}(\phi)$ has a non-empty Boolean barrier of finite size. The reason that a Boolean barrier must exist is simply because ϕ is tractable and if ϕ would not have a Boolean barrier then every Boolean CSP instance would be closed under ϕ and thus tractable, which unless P = NP is not possible. To see that $\lambda(\phi)$ is finite first note that $|\lambda(\phi)|$ is at most as large as the arity $c_\mathbb{P}$ of ϕ. Moreover, $r(\lambda) \leq 2^{c_\mathbb{P}}$ because there are at most $2^{c_\mathbb{P}}$ distinct $c_\mathbb{P}$-ary tuples over $\{0,1\}$. Hence the size of $\lambda(\phi)$ is at most $c_\mathbb{P} \cdot 2^{c_\mathbb{P}}$.

We show the theorem via an fpt-reduction from HITTING SET. Given an instance $(\mathcal{Q}, U, k)$ of HITTING SET, we construct a CSP instance $I = \langle V, \{0,1\}, C \rangle$ such that $\mathcal{Q}$ has a hitting set of size at most k if and only if I has a strong $\mathcal{C}_\mathbb{P}$-backdoor set of size at most k. Let Λ be the set of all Boolean barriers of polymorphisms with property $\mathbb{P}$, i.e., $\Lambda = \{\lambda(\phi) : \phi \text{ has property } \mathbb{P}\}$. Note that the cardinality of Λ only depends on $c_\mathbb{P}$ and can thus be considered to be finite. The variables of I are $\{x_u : u \in U\} \cup \{o_1(\lambda, Q), \ldots, o_{r(\lambda)}(\lambda, Q) : \lambda \in \Lambda \text{ and } Q \in \mathcal{Q}\}$. Furthermore, for every Boolean barrier $\lambda \in \Lambda$ with

$\lambda = \{t_1, \ldots, t_n\}$ for some natural number n, and $Q \in \mathcal{Q}$ with $Q = \{u_1, \ldots, u_{|Q|}\}$, C contains a constraint $R(\lambda, Q)$ with scope $\langle o_1(\lambda, Q), \ldots, o_{r(\lambda)}(\lambda, Q), x_{u_1}, \ldots, x_{u_{|Q|}} \rangle$ whose relation contains the row

$$t_i[1], \ldots, t_i[r(\lambda)], \underbrace{\langle i \mod 2, \ldots, i \mod 2 \rangle}_{|Q| \text{times}}$$

for every i in $1 \leq i \leq n$. This completes the construction of I. Observe that I is satisfiable, because the first rows of every constraint of I are pairwise compatible. In particular, set all the variables of the form $\{x_u : u \in U\}$ to zero, and set all remaining variables according to their first-row constraints — note that these rows are the same for different $Q \neq Q'$ associated with the same Boolean barrier λ, while on the other hand, for different Boolean barriers $\lambda \neq \lambda'$, the associated variables are different and there is no cause for conflict. Suppose that $\mathcal{Q}$ has a hitting set B of size at most k. We claim that $B_u = \{x_u : u \in B\}$ is a strong $\mathcal{C}_\mathbb{P}$-backdoor set of I. Let $\tau : B_u \to D$ be an assignment of the variables in B and $\lambda \in \Lambda$ be a Boolean barrier of maximum cardinality over all Boolean barriers in Λ. We claim that $I[\tau]$ is closed under any operation ϕ with property $\mathbb{P}$ with $\lambda = \lambda(phi)$ and hence B_u is a strong $\mathcal{C}_\mathbb{P}$-backdoor set of I. Because B is a hitting set of $\mathcal{Q}$, it follows that every relation of $I[\tau]$ contains at most half of the tuples of the corresponding relation in I. Furthermore, because ϕ is idempotent (and hence every tuple is mapped to itself), it holds that $|lambda'| > 1$ for every $\lambda' \in \Lambda$. It follows that every relation of $I[\tau]$ contains at least one tuple less than the corresponding relation in I. Because of the choice of λ, we obtain that every constraint $c \in C[\tau]$ contains less than $|\lambda|$ tuples, and hence $I[\tau]$ is closed under any operation ϕ with property $\mathbb{P}$ and $\lambda = \lambda(\phi)$. Hence, $B_u = \{x_u : u \in B\}$ is a strong $\mathcal{C}_\mathbb{P}$-backdoor set of I, as required. Towards showing that B_u is also a weak $\mathcal{C}_\mathbb{P}$-backdoor set of I consider the assignment $\tau : B_u \to \{0\}$. It follows from the above argumentation that $C[\tau]$ is closed under any operation ϕ with property $\mathbb{P}$ and $\lambda = \lambda(\phi)$. Furthermore, $I[\tau]$ is satisfiable, because the first row of every constraint in C is also contained in every constraint of $I[\tau]$ and these rows are pairwise compatible.

For the reverse direction, suppose that I has a strong $\mathcal{C}_\mathbb{P}$-backdoor set B of size at most k. Because for every $Q \in \mathcal{Q}$ it holds that the set of constraints $\{R(\lambda, Q) : \lambda \in \Lambda\}$ is not closed under any operation ϕ with property $\mathbb{P}$, we obtain that B has to contain at least one variable from $\bigcup_{\lambda \in \Lambda} V(R(\lambda, Q))$ for every $Q \in \mathcal{Q}$. Since the only variables that are shared between $\bigcup_{\lambda \in \Lambda} V(R(\lambda, Q))$ and $\bigcup_{\lambda \in \Lambda} V(R(\lambda, Q'))$ for distinct $Q, Q' \in \mathcal{Q}$ are the variables in $\{x_u : u \in U\}$, it follows that $\mathcal{Q}$ has a hitting set of size at most $|B| \leq k$, as required. $\qquad\square$

Because all min/max, majority, minority and Mal'cev operations can be defined via tractable properties and are idempotent, we obtain.

Corollary 3. *For every $\mathcal{C} \in \{\text{MINMAX, MAJ, MIN, MAL}\}$,* STRONG $\mathcal{C}$-BACKDOOR DETECTION *and* WEAK $\mathcal{C}$-BACKDOOR DETECTION *are* W[2]-*hard parameterized by the size of the backdoor set, even for CSP instances over the Boolean domain.*

In the following we show that hardness also holds if we drop the restriction on the domain of the CSP instance but instead consider only CSP instances of arity 2. To do so we need the following lemma.

Lemma 1. *For every $\mathcal{C} \in \{\text{MINMAX, MAJ, MIN, MAL}\}$ and every $k \geq 3$, there is a 2-ary CSP instance $I(\mathcal{C}, k)$ with k constraints, each containing the all 0 tuple, such that $I(\mathcal{C}, k) \notin \mathcal{C}$ but for every assignment τ of at least one variable of $I(\mathcal{C}, k)$, it holds that $I(\mathcal{C}, k)[\tau] \in \mathcal{C}$.*

Proof. The proof of this lemma is inspired by the proof of [35, Lemma 3]. For every $\mathcal{C}$ as in the statement of the lemma, $I(\mathcal{C}, k)$ has variables $v_1, \ldots, v_{2k}$ and contains a constraint c_i with scope $\langle v_{2i-1}, v_{2i} \rangle$ for every $1 \leq i \leq k$.

If $\mathcal{C} = \text{MAJ}$, the domain of $I(\mathcal{C}, k)$ is $\{1, \ldots, 3k - 4\}$, and for every $1 \leq i \leq k$ the relations $R(c_i)$ are defined as follows: $R(c_1) = \{(0,0), (1,3), (1,4), (2,5)\}$, $R(c_2) = \{(0,0), (1,3), (2,4), (1,5)\}$, $R(c_3) = \{(0,0), (2,3(k-2)), (1,3(k-2)+1), (2,3(k-2)+2)\}$, and for every $3 < i \leq k$, $R(c_i) = \{(0,0), (3(i-3), 3(i-2)), (3(i-3)+1, 3(i-2)+1), (3(i-3)+2, 3(i-2)+2)\}$. Clearly, the instance is binary and every relation contains the all 0 tuple. We first show that $I(\mathcal{C}, k)$ is not closed under any majority operation ϕ. From

20

$R(c_1)$ and $R(c_2)$, we obtain $\phi(3,4,5) = 3$ and from $R(c_3)$, we obtain $\phi(3(k-2), 3(k-2)+1, 3(k-2)+2) \in \{3(k-2)+1, 3(k-2)+2\}$. Finally, from $R(c_i)$ with $3 < i \leq k$, we obtain $\phi(3(i-2), 3(i-2)+1, 3(i-2)+2) = j$ if and only if $\phi(3(i-1), 3(i-1)+1, 3(i-1)+2) = j+3$. This chain implies $f(3,4,5) \in \{4,5\}$ in contradiction to $f(3,4,5) = 3$. On the other hand, for any assignment τ that sets at least 1 variable of $I(\mathcal{C}, k)$, it holds that at least one relation of $I(\mathcal{C}, k)[\tau]$ contains at most 2 tuples. This relation is then closed under any majority operation and breaks the above chain of implications.

If $\mathcal{C} = $ MINMAX, the domain of $I(\mathcal{C}, k)$ is $\{1, \ldots, k+1\}$, and for every $1 \leq i \leq k$ the relations $R(c_i)$ are defined as follows: $R_i = \{(0,0), (i, i+1), (i+1, i), (i+1, i+1)\}$ for every $1 \leq i < k$ and $R_k = \{(0,0), (1, k+1), (k+1, 1), (1,1)\}$. Clearly, the instance is binary and every relation contains the all 0 tuple. Let ϕ be a max operation (the proof remains the same if ϕ is a min operation). Then, for every $1 \leq i < k$, the relation R_i implies $i < i+1$, but the relation R_k implies $1 > k+1$, a contradiction. Hence, the set of all relations $R_1, \ldots, R_k$ is not closed under any max operation. On the other hand, for any assignment τ that sets at least 1 variable of $I(\mathcal{C}, k)$, it holds that at least one relation of $I(\mathcal{C}, k)[\tau]$ contains at most 2 tuples that agree on (at least) one coordinate. This relation is then closed under any max operation and breaks the above chain of implications.

If $\mathcal{C} = $ MIN, the domain of $I(\mathcal{C}, k)$ is $\{1, \ldots, 3k+2\}$, and for every $1 \leq i \leq k$ the relations $R(c_i)$ are defined as follows: $R_1 = \{(0,0), (1,3), (1,4), (2,5)\}$, $R_i = \{(0,0), (3(i-1), 3i), (3(i-1)+1, 3i+1), (3(i-1)+2, 3i+2)\}$ for every $1 < i < k$, and $R_k = \{(0,0), (1, 3(k-1)), (2, 3(k-1)+1), (1, 3(k-1)+1)\}$. Clearly, the instance is binary and every relation contains the all 0 tuple. Let ϕ be a minority operation. Then, R_1 implies $f(3,4,5) = 5$, which together with the relations R_i, for every $1 < i < k$ implies $f(3i, 3i+1, 3i+2) = 3i+2$. In particular, we obtain $f(3(k-1), 3(k-1)+1, 3(k-1)+2) = 3(k-1)+2$, which contradicts relation R_k. Hence, the set of all relations $R_1, \ldots, R_k$ is not closed under any minority operation. On the other hand, for any assignment τ that sets at least 1 variable of $I(\mathcal{C}, k)$, it holds that at least one relation of $I(\mathcal{C}, k)[\tau]$ contains at most 2 tuples. This relation is then closed under any minority operation and breaks the above chain of implications. If $\mathcal{C} = $ MAL, we can use the same set of relations. $\square$

Theorem 13. *For every $\mathcal{C} \in \{$MINMAX, MAJ, MIN, MAL$\}$, STRONG $\mathcal{C}$-BACKDOOR DETECTION and WEAK $\mathcal{C}$-BACKDOOR DETECTION are W[2]-hard parameterized by the size of the backdoor set even for CSP instances with arity 2.*

Proof. The proof of the theorem is inspired by the proof of [35, Theorem 6]. We show the theorem via a fpt-reduction from HITTING SET. Let $H = \langle \mathcal{U}, \mathcal{F}, k \rangle$ be an instance of HITTING SET.

We construct a CSP instance $I = \langle V, D, C \rangle$ of arity 2 such that H has a hitting set of size at most k if and only if I has a strong $\mathcal{C}$-backdoor set of size at most k. The variables of I are $\{ x_u : u \in \mathcal{U} \} \cup \{ y_F^i : F \in \mathcal{F}$ and $1 \leq i \leq |F| \}$. Furthermore, for every $F \in \mathcal{F}$ with $F = \{u_1, \ldots, u_{|F|}\}$, let $I(F)$ be the CSP instance obtained from $I(\mathcal{C}, |F|)$ (see Lemma 1), where for every i with $1 \leq i \leq |F|$, we replace the scope of the constraint c_i with $\langle x_{u_i}, y_F^i \rangle$ and adapt $I(\mathcal{C}, |F|)$ in such a way that for distinct $F, F' \in \mathcal{F}$, $I(\mathcal{C}, |F|)$ and $I(\mathcal{C}, |F'|)$ have no common domain value. Then, the constraints of I is the union of the constraints of $I(F)$ over every $F \in \mathcal{F}$. This completes the construction of I. Clearly, I has arity 2 and is satisfiable, e.g., by the all 0 assignment.

Suppose that H has a hitting set B of size at most k. We claim that $B_u = \{ x_u : u \in B \}$ is a strong $\mathcal{C}$-backdoor set of I. We start by showing that B_u is a strong $\mathcal{C}$-backdoor set of I. Let $\tau : B_u \to D$ be an assignment of the variables in B. Then, it follows from Lemma 1 that the constraints of $I(F)[\tau]$ are in $\mathcal{C}$. In particular, there is an operation ϕ (which is either min, max, majority, minority or Mal'cev, depending on $\mathcal{C}$) such that the constraints of $I(F)[\tau]$ are closed under ϕ. Because for every distinct $F, F' \in \mathcal{F}$, $I(F)[\tau]$ and $I(F')[\tau]$ do not have a common domain value, it follows that there is an operation (which is a combination of the operations for each $I(F)[\tau]$) such that $I[\tau]$ is closed under ϕ and ϕ is a min/max, majority, minority, or Mal'cev operation (depending on $\mathcal{C}$). Hence, $I[\tau] \in \mathcal{C}$, as required. Towards showing that B_u is also a weak $\mathcal{C}$-backdoor set of C consider the assignment $\tau : B_u \to \{0\}$. It follows from the above argumentation that $I[\tau] \in \mathcal{C}$. Furthermore, $I[\tau]$ is satisfiable, because every constraint of $I[\tau]$ still contains the all 0 tuple.

For the reverse direction, suppose that I has a strong $\mathcal{C}$-backdoor set B of size at most k. Because for every $F \in \mathcal{F}$ it holds that the set of constraints of $I(F)$ is not in $\mathcal{C}$, we obtain that B has to contain at least

1 variable from $V(I(F))$ for every $F \in \mathcal{F}$. Since the only variables that are shared between $V(I(F))$ and $V(I(F'))$ for distinct $F, F' \in \mathcal{F}$ are the variables in $\{\, x_u : u \in \mathcal{U} \,\}$, it follows that H has a hitting set of size at most $|B| \leq k$, as required. $\qquad\square$

7. Comparison to Partition-Backdoors

In this section we draw a comparison between our notion of backdoor sets, and the approach recently introduced by Bessiere et al. (2013), which we will call the *partition backdoor set* approach. The main difference is that the partition backdoor set approach considers a backdoor with respect to a subset of constraints that is tractable (because it is closed under some tractable polymorphism). In particular, Bessiere et al. (2013) define partition backdoor sets as follows.

Definition 1. *Let $\mathbb{P}$ be a tractable polymorphism property. A $\mathbb{P}$-partition backdoor of a CSP instance $I = \langle V, D, C \rangle$ is a set B of variables, such that there is a partition of C into C_1 and C_2 for which the following holds:*

- *$\langle V, D, C_2 \rangle \in \mathcal{C}_\mathbb{P}$, and*

- *If $\mathbb{P}$ is characterized using only idempotent polymorphisms, then B is the set of all variables in the scope of the constraints of C_2, while if $\mathbb{P}$ is characterized using only conservative polymorphisms, then the variables of B are a vertex cover of the primal graph of C_1.*

Note that the primal graph of a set C of constraints is the graph, whose vertices are all the variables appearing in the scope of the constraints in C and that has an edge between two variables if and only if they appear together in the scope of at least one constraint in C.

Bessiere et al. (2013) show that given a CSP instance and a partition (C_1, C_2) as defined above, then the detection and evaluation of $\mathbb{P}$-partition backdoor sets are fixed-parameter tractable

- in the size of the domain and the number of variables in C_1, if $\mathbb{P}$ is characterized using only idempotent polymorphisms;

- in the size of the domain and the vertex cover of the primal graph of C_1, if $\mathbb{P}$ is characterized using only conservative polymorphisms.

Observe that the above tractability results only hold when the partition (C_1, C_2) is given as a part of the input, if it is not then the detection of partition backdoors is fixed-parameter intractable (for most combinations of parameters).

Apart from the complexity of detecting and evaluating backdoors it is also important that the backdoor sets are as small as possible over a wide variety of CSP instances. In the following we will observe that our backdoor sets are always at most as large as partition backdoors and we exhibit CSP instances where our backdoors have size one but the size of a smallest partition backdoor set can be arbitrary large.

Proposition 2. *Let $\mathbb{P}$ be a tractable polymorphism property. Then, for every CSP instance I the size of a smallest strong $\mathbb{P}$-backdoor set of I is at most the size of a smallest $\mathbb{P}$-partition backdoor set of I.*

Proof. This follows from the observation that any $\mathbb{P}$-partition backdoor set is also a strong $\mathbb{P}$-backdoor set. $\qquad\square$

Proposition 3. *For any tractable property $\mathbb{P}$ and any natural number n , there is a CSP instance that has a strong/weak $\mathcal{C}_\mathbb{P}$-backdoor set of size one, but the size of a smallest $\mathbb{P}$-partition backdoor set is $\Omega(n)$.*

Proof. Let $\mathbb{P}$ be a tractable property and let $I = \langle V, D, C \rangle$ be a CSP instance such that:

- no constraint of I is closed under a polymorphism with property $\mathbb{P}$;

- all constraints of I share a common variable, say x, such that for any assignment τ of x, the CSP instance $I[\tau]$ is closed under some polymorphisms with property $\mathbb{P}$.

It is straightforward to check that for any number of variables n, such a CSP instance using n variables can be easily constructed for every tractable property $\mathbb{P}$ that contains at least one operation. Furthermore, the variable x is a strong $\mathcal{C}_{\mathbb{P}}$-backdoor set of I. However, because no constraint of I is closed under any operation from $\mathbb{P}$ it follows that every partition backdoor has to contain at least all but 1 variable from every constraint of I. $\qquad\square$

8. Summary

We have introduced heterogeneous base classes and have shown that strong backdoor sets into such classes can be considerably smaller than strong backdoor sets into homogeneous base classes; nevertheless, the detection of strong backdoor sets into heterogeneous base classes is still fixed-parameter tractable in many natural cases. Hence, our results push the tractability boundary considerably further. Our theoretical evaluation entails hardness results that establish the limits for tractability.

Our main result in the context of SAT (Theorem 6) provides a complete complexity classification for strong backdoor set detection to heterogeneous base classes composed of the well-known Schaefer classes. We observe that the hardness results rely on clauses of unbounded length and show that if the maximum length of the clauses is taken as an additional parameter then strong as well as weak backdoor set detection into any combination of the Schaefer classes is fixed-parameter tractable (Theorem 7).

Our main result for CSP backdoors (Theorem 10) establishes fixed-parameter tractability for a wide range of heterogeneous base classes composed of possibly infinitely many tractable polymorphisms. In particular, we show that the detection of strong as well as weak backdoor sets is fixed-parameter tractable for the combined parameter backdoor size, domain size, and the maximum arity of constraints. In fact, this result entails *heterogeneous* base classes, as different instantiations of the backdoor variables can lead to reduced instances that are closed under different polymorphisms (even polymorphisms of different type). We complement our main result with hardness results that show that we lose fixed-parameter tractability when we omit either domain size or the maximum arity of constraints from the parameter (Theorems 12 and 13).

It would be interesting to extend our line of research to the study of backdoor sets into heterogeneous base classes composed of homogeneous classes defined by e.g., structural restrictions such as bounded treewidth [36, 37] or bounded hyper-treewidth [38], and base classes defined by "hybrid" concepts like forbidden patterns [39], in contrast to the Schaefer classes for SAT, and polymorphism-based classes for CSP. Another promising direction of future research is to parameterize backdoor sets not by their size but by structural properties of the backdoor set and how it interacts with the rest of the instance, similar to the parameters that have been considered for modulators into graph classes [40, 41]; some first results in this direction have recently been obtained [42].

References

[1] Gaspers, S., Misra, N., Ordyniak, S., Szeider, S., Živný, S.. Backdoors into heterogeneous classes of SAT and CSP. In: Brodley, C.E., Stone, P., editors. Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada. AAAI Press; 2014, p. 2652–2658.

[2] Williams, R., Gomes, C., Selman, B.. Backdoors to typical case complexity. In: Gottlob, G., Walsh, T., editors. Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, IJCAI 2003. Morgan Kaufmann; 2003, p. 1173–1178.

[3] Williams, R., Gomes, C., Selman, B.. On the connections between backdoors, restarts, and heavy-tailedness in combinatorial search. In: Informal Proc. of the Sixth International Conference on Theory and Applications of Satisfiability Testing, S. Margherita Ligure - Portofino, Italy, May 5-8, 2003 (SAT 2003). 2003, p. 222–230.

[4] Ruan, Y., Kautz, H.A., Horvitz, E.. The backdoor key: A path to understanding problem hardness. In: McGuinness, D.L., Ferguson, G., editors. Proceedings of the 19th National Conference on Artificial Intelligence, 16th Conference on Innovative Applications of Artificial Intelligence. AAAI Press / The MIT Press; 2004, p. 124–130.

[5] Dilkina, B.N., Gomes, C.P., Sabharwal, A.. Tradeoffs in the complexity of backdoor detection. In: Principles and Practice of Constraint Programming - CP 2007, 13th International Conference, CP 2007, Providence, RI, USA, September 23-27, 2007, Proceedings; vol. 4741 of *Lecture Notes in Computer Science*. Springer Verlag; 2007, p. 256–270.

[6] Samer, M., Szeider, S.. Fixed-parameter tractability. In: Biere, A., Heule, M., van Maaren, H., Walsh, T., editors. Handbook of Satisfiability; chap. 13. IOS Press; 2009, p. 425–454.

[7] Kottler, S., Kaufmann, M., Sinz, C.. A new bound for an NP-hard subclass of 3-SAT using backdoors. In: Büning, H.K., Zhao, X., editors. Theory and Applications of Satisfiability Testing - SAT 2008, 11th International Conference, SAT 2008, Guangzhou, China, May 12-15, 2008. Proceedings; vol. 4996 of *Lecture Notes in Computer Science*. Springer Verlag; 2008, p. 161–167.

[8] Dilkina, B.N., Gomes, C.P., Sabharwal, A.. Backdoors in the context of learning. In: Kullmann, O., editor. Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings; vol. 5584 of *Lecture Notes in Computer Science*. Springer Verlag; 2009, p. 73–79.

[9] Gaspers, S., Szeider, S.. Strong backdoors to bounded treewidth SAT. In: 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA. IEEE Computer Society; 2013, p. 489–498.

[10] LeBras, R., Bernstein, R., Gomes, C.P., Selman, B., van Dover, R.B.. Crowdsourcing backdoor identification for combinatorial optimization. In: Rossi, F., editor. IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013. IJCAI/AAAI; 2013, p. 2840–2847.

[11] Pfandler, A., Rümmele, S., Szeider, S.. Backdoors to abduction. In: Rossi, F., editor. IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013. IJCAI/AAAI; 2013, p. 1046–1052.

[12] Dvorák, W., Ordyniak, S., Szeider, S.. Augmenting tractable fragments of abstract argumentation. Artificial Intelligence 2012;186:157–173.

[13] Samer, M., Szeider, S.. Backdoor sets of quantified Boolean formulas. Journal of Automated Reasoning 2009;42(1):77–97.

[14] Kronegger, M., Ordyniak, S., Pfandler, A.. Backdoors to planning. In: Brodley, C.E., Stone, P., editors. Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada. AAAI Press; 2014, p. 2300–2307.

[15] Kronegger, M., Ordyniak, S., Pfandler, A.. Variable-deletion backdoors to planning. In: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25 -30, 2015, Austin Texas, Austin, USA. AAAI Press; 2014, p. 2300–2307.

[16] Downey, R.G., Fellows, M.R.. Fundamentals of Parameterized Complexity. Texts in Computer Science; Springer Verlag; 2013.

[17] Nishimura, N., Ragde, P., Szeider, S.. Detecting backdoor sets with respect to Horn and binary clauses. In: Proceedings of SAT 2004 (Seventh International Conference on Theory and Applications of Satisfiability Testing, 10–13 May, 2004, Vancouver, BC, Canada). 2004, p. 96–103.

[18] Gaspers, S., Szeider, S.. Backdoors to satisfaction. In: Bodlaender, H.L., Downey, R., Fomin, F.V., Marx, D., editors. The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday; vol. 7370 of *Lecture Notes in Computer Science*. Springer Verlag; 2012, p. 287–317.

[19] Schaefer, T.J.. The complexity of satisfiability problems. In: Conference Record of the Tenth Annual ACM Symposium on Theory of Computing (San Diego, Calif., 1978). ACM; 1978, p. 216–226.

[20] Bulatov, A.A., Dalmau, V.. A simple algorithm for mal'tsev constraints. SIAM J Comput 2006;36(1):16–27.

[21] Idziak, P.M., Markovic, P., McKenzie, R., Valeriote, M., Willard, R.. Tractability and learnability arising from algebras with few subpowers. SIAM Journal on Computing 2010;39(7):3023–3037.

[22] Barto, L., Kozik, M.. Constraint Satisfaction Problems Solvable by Local Consistency Methods. Journal of the ACM 2014;61(1). Article No. 3.

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

[24] Barto, L.. Constraint satisfaction problem and universal algebra. ACM SIGLOG News 2014;1(2):14–24.

[25] Carbonnel, C., Cooper, M.C., Hebrard, E.. On backdoors to tractable constraint languages. In: O'Sullivan, B., editor. Principles and Practice of Constraint Programming - 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings; vol. 8656 of *Lecture Notes in Computer Science*. Springer; 2014, p. 224–239.

[26] Ganian, R., Ramanujan, M.S., Szeider, S.. Discovering archipelagos of tractability for constraint satisfaction and

counting. In: Krauthgamer, R., editor. Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016. SIAM; 2016, p. 1670–1681.

[27] Carbonnel, C., Cooper, M.C.. Tractability in constraint satisfaction problems: a survey. Constraints 2016;21(2):115–144.

[28] Knuth, D.E.. The Art of Computer Programming, Volume IVa: Combinatorial Algorithms, Part 1. Addison-Wesley; 2011. ISBN 0-201-03804-8.

[29] Szeider, S.. Matched formulas and backdoor sets. J on Satisfiability, Boolean Modeling and Computation 2008;6:1–12.

[30] Jeavons, P., Cohen, D.A., Cooper, M.C.. Constraints, consistency and closure. Artificial Intelligence 1998;101(1-2):251–265.

[31] Feder, T., Vardi, M.Y.. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. SIAM J Comput 1998;28(1):57–104.

[32] Idziak, P.M., Markovic, P., McKenzie, R., Valeriote, M., Willard, R.. Tractability and learnability arising from algebras with few subpowers. SIAM J Comput 2010;39(7):3023–3037.

[33] Kozik, M., Krokhin, A., Valeriote, M., Willard, R.. Characterizations of several Maltsev Conditions. Algebra Universalis 2015;73(3-4):205–224.

[34] Barto, L., Kozik, M.. Constraint satisfaction problems solvable by local consistency methods. J of the ACM 2014;61(1):3:1–3:19.

[35] Bessiere, C., Carbonnel, C., Hebrard, E., Katsirelos, G., Walsh, T.. Detecting and exploiting subproblem tractability. In: Rossi, F., editor. IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013. IJCAI/AAAI; 2013, p. 468–474.

[36] Gaspers, S., Szeider, S.. Strong backdoors to bounded treewidth SAT. In: 54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA. IEEE Computer Society; 2013, p. 489–498.

[37] Samer, M., Szeider, S.. Constraint satisfaction with bounded treewidth revisited. J of Computer and System Sciences 2010;76(2):103–114.

[38] Gottlob, G., Leone, N., Scarcello, F.. Hypertree decompositions and tractable queries. J of Computer and System Sciences 2002;64(3):579–627.

[39] Cohen, D.A., Cooper, M.C., Creed, P., Marx, D., Salamon, A.Z.. The tractability of CSP classes defined by forbidden patterns. J Artif Intell Res 2012;45:47–78.

[40] Eiben, E., Ganian, R., Szeider, S.. Meta-kernelization using well-structured modulators. In: Husfeldt, T., Kanj, I.A., editors. Parameterized and Exact Computation - 10th International Symposium, IPEC 2014, Patras, Greece, September 16-18, 2015. Revised Selected Papers; vol. 43 of *LIPIcs*. 2015, p. 114–126. URL: `http://drops.dagstuhl.de/opus/volltexte/2015/5576/pdf/13.pdf`. doi:10.4230/LIPIcs.IPEC.2015.114.

[41] Eiben, E., Ganian, R., Szeider, S.. Solving problems on graphs of high rank-width. In: Algorithms and Data Structures Symposium (WADS 2015), August 5-7, 2015, University of Victoria, BC, Canada. LNCS; Springer Verlag; 2015, p. 314–326. URL: `http://arxiv.org/abs/1507.05463`. doi:10.1007/978-3-319-21840-3_26.

[42] Ganian, R., Ramanujan, M.S., Szeider, S.. Combining treewidth and backdoors for CSP. CoRR 2016;abs/1610.03298. URL: `http://arxiv.org/abs/1610,03298`.