Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (2008)

Backdoor Trees

Marko Samer and **Stefan Szeider**
Department of Computer Science
Durham University, UK
{marko.samer,stefan.szeider}@durham.ac.uk

Abstract

The surprisingly good performance of modern satisfiability (SAT) solvers is usually explained by the existence of a certain "hidden structure" in real-world instances. We introduce the notion of backdoor trees as an indicator for the presence of a hidden structure. Backdoor trees refine the notion of strong backdoor sets, taking into account the relationship between backdoor variables. We present theoretical and empirical results. Our theoretical results are concerned with the computational complexity of detecting small backdoor trees. With our empirical results we compare the size of backdoor trees against the size of backdoor sets for real-world SAT instances and random 3SAT instances of various density. The results indicate that backdoor trees amplify the properties that have been observed for backdoor sets.

Introduction

Today's state-of-the-art satisfiability (SAT) solvers routinely solve huge real-world instances with hundreds of thousands of variables. This successful performance of solvers is in strong contrast to theoretical worst-case upper bounds: no algorithm is known that solves n-variable SAT instances in time $2^{o(n)}$ and it is believed that no such algorithm exits (Impagliazzo, Paturi, & Zane 2001). This wide gap between theoretical upper bounds and empirical data is usually explained by the presence of a certain "hidden structure" in real-world instances. The notion of backdoor sets, introduced by Williams, Gomes, and Selman (2003a; 2003b) allows to make the rather vague notion of a hidden structure precise and explicit. Basically, a backdoor set of a SAT instance F is a (small) set B of variables that provide the key for solving the instance. Backdoor sets are defined with respect to a fixed base class of tractable instances. The base class can be defined algorithmically (say, the class of instances that can be decided by the polynomial-time simplification and propagation methods of a SAT solver) or syntactically (say, the class of all Horn instances). A set B of variables is a weak backdoor set if F is satisfiable and there exists a truth assignment to the variables in B that puts the instance into the base class; B is a strong backdoor set if all possible truth assignments to the variables in B put the instance into the base class. We give more exact definitions below.

There is empirical evidence that real-world instances actually have small backdoor sets (see related work below). Note that if we know a strong backdoor set of size k, then we can decide the given instance by solving 2^k tractable instances that belong to the base class.

In this paper we propose a more refined view that takes the interaction of variables that form a backdoor set into account. We shift the focus from the *size* k of a backdoor set to the actual *number of tractable instances the strong backdoor set produces*, a number that can be considerably smaller than 2^k. We claim that this number gives a better indication for the presence of structure than the plain backdoor set size.

As formal basis for our investigations we introduce the concept of *backdoor trees*. Backdoor trees are binary decision trees on backdoor variables whose leaves correspond to instances of the base class. Every strong backdoor set of size k gives rise to a backdoor tree with at least $k+1$ and at most 2^k leaves. It is reasonable to rank the hardness of instances in terms of the number of leaves of backdoor trees, thus gaining a more refined view than by just comparing the size of backdoor sets. An appealing feature of strong backdoor sets and backdoor trees is that they apply in the same way to satisfiable and unsatisfiable instances (in contrast to some hardness parameters defined in terms of resolution refutations).

In this paper we focus on the following fundamental base classes: 2CNF, the class of CNF formulas with binary clauses (each clause contains at most two literals); HORN, the class of Horn formulas (each clause contains at most one positive literal); RHORN, the class of renamable Horn formulas (formulas that can be made Horn by "flipping" some variables, i.e., by replacing all occurrences of x by $\neg x$ and all occurrences of $\neg x$ by x, for all variables x belonging to some fixed set X).

Our theoretical results are concerned with the computational complexity of *backdoor tree detection:* Given a CNF formula F and an integer $k > 0$, decide whether F has a backdoor tree with at most k leaves. We show that the problem is NP-hard for the three fundamental base classes mentioned above. Further we investigate whether the problem becomes easier if k is assumed to be a small integer; i.e., we investigate the *fixed-parameter tractability* of the problem, where k is considered as the parameter (see the next section for basic definitions). In fact, we show that backdoor tree detection is fixed-parameter tractable for

the base classes 2CNF and HORN, extending the fixed-parameter tractability results of Nishimura et al. (2004) for backdoor *set* detection. We establish this result by means of a polynomial-time preprocessing algorithm.

For our empirical investigations we use a search algorithm that computes upper bounds on the number of leaves of backdoor trees. We consider benchmark sets for logistic planning as well as random instances of various density. The search algorithm is parameterized by the depth of the search tree explored. For depth 0 the computed upper bound is just the trivial 2^k bound obtained from the size k of a backdoor set. Gradually increasing the depth reveals how backdoor trees improve over backdoor sets.

We also compare theoretically and empirically the number of leaves of backdoor trees with the primal treewidth of instances (an indicator for 'tree likeness', see, e.g., (Samer & Szeider 2007b)).

Related Work. Backdoor sets were introduced by William, Gomes, and Selman (2003a; 2003b) as an analytic tool for gaining insight into the heavy-tailed behavior of complete backtrack-search methods. With different terminology and context, this concept was also studied by Crama, Elkin, and Hammer (1997). The dependency among the variables of a minimal weak backdoor set was studied by Ruan, Kautz, and Horvitz (2004). The size of backdoor sets was empirically investigated by several authors, e.g., Interian (2003) (weak backdoor sets for random formulas), Lynce and Marques-Silva (2004) (strong backdoor sets vs. minimal unsatisfiable cores), Kilby, Slaney, Thiebaux, and Walsh (2005) (weak backdoor sets vs. backbones), Dilkina, Gomes, and Sabharwal (2007) (strong backdoor sets and empty clause detection). The computational complexity of finding small backdoor sets for various base classes was studied by Nishimura, Ragde, and Szeider (2004; 2007) (strong backdoor with respect to the base classes HORN and 2CNF, hitting formulas) and Szeider (2005) (weak and strong backdoor sets with respect to the base classes defined via unit propagation and pure literal elimination). Samer and Szeider (2007a) extended the notion of backdoor sets to quantified Boolean formulas (QBFs).

Backdoor Sets and Backdoor Trees

We consider propositional formulas in conjunctive normal form (CNF) represented by sets of clauses, e.g., $F = \{\{x, \neg y\}, \{\neg x, z\}\}$ represents the CNF formula $(x \vee \neg y) \wedge (\neg x \vee z)$. For a CNF formula F, $var(F)$ denotes the set of variables occurring negated or unnegated in F. A (partial truth) *assignment* is a mapping $\tau : X \to \{0, 1\}$ (0 representing false, 1 representing true) defined on a set X of variables. We write $var(\tau) = X$. If $var(\tau) = \{x\}$ then we denote τ simply by '$x = 1$' or '$x = 0$'. An assignment τ extends in the obvious way to literals over $var(\tau)$ via $\tau(\neg x) = 1 - \tau(x)$. $F[\tau]$ denotes the *restriction* of F to τ (i.e., $F[\tau]$ is the CNF formula obtained from F by removing all clauses that contain a literal that is true under τ, and by removing from the remaining clauses all literals that are false under τ). A CNF formula F is *satisfiable* if $F[\tau] = \emptyset$ for some assignment τ. We also consider *variable deletion* in the following form: If X is a set of variables and F a CNF formula, then $F - X$ denotes the CNF formula obtained from F by removing from all clauses literals of the form x or $\neg x$ for $x \in X$.

Base Classes. A *base class* is a class of CNF formulas for which both membership and satisfiability can be decided in polynomial time. Throughout this paper we also assume that *self-reducibility* holds for the considered base classes $\mathcal{C}$: For every $F \in \mathcal{C}$ and $x \in var(F)$ also $F[x = 0], F[x = 1] \in \mathcal{C}$. A base class $\mathcal{C}$ is *clause-induced* if for every $F \in \mathcal{C}$ and $F' \subseteq F$ also $F' \in \mathcal{C}$. Note that 2CNF, HORN, and RHORN are all clause-induced.

Backdoor Sets. Let $\mathcal{C}$ be a base class, F a CNF formula, and $B \subseteq var(F)$. Then B is a *strong $\mathcal{C}$-backdoor set of F* if $F[\tau] \in \mathcal{C}$ for every truth assignment $\tau : B \to \{0, 1\}$ (there is also the notion of a "weak backdoor set" which, however, we will not consider in this paper). For each base class $\mathcal{C}$ we consider the following problem:

STRONG $\mathcal{C}$-BACKDOOR SET. *Instance:* A CNF formula F and a non-negative integer k. *Parameter:* The integer k. *Question:* Has F a $\mathcal{C}$-backdoor set of cardinality at most k?

Let B be a strong $\mathcal{C}$-backdoor set of a CNF formula F. B is *smallest* if F has no strong $\mathcal{C}$-backdoor set that is smaller than B; B is *minimal* if F has no strong $\mathcal{C}$-backdoor set that is a proper subset of B.

Note that if $\mathcal{C}$ is clause-induced, then every $B \subseteq var(F)$ with $F - B \in \mathcal{C}$ is a strong $\mathcal{C}$-backdoor set of F. Let us call such a set B a *deletion* $\mathcal{C}$-backdoor set. It is not difficult to see that for $\mathcal{C} \in \{\text{2CNF}, \text{HORN}\}$ every strong $\mathcal{C}$-backdoor set is also a deletion $\mathcal{C}$-backdoor set (Crama, Ekin, & Hammer 1997; Nishimura, Ragde, & Szeider 2004). This is not the case for $\mathcal{C} = \text{RHORN}$ where the size of a smallest strong RHORN-backdoor set can be exponentially smaller than the size of a smallest deletion HORN-backdoor set (Dilkina, Gomes, & Sabharwal 2007). The problem DELETION $\mathcal{C}$-BACKDOOR SET is defined similarly as STRONG $\mathcal{C}$-BACKDOOR SET.

From general results of Crama, Ekin, and Hammer (1997) it follows that the problems STRONG $\mathcal{C}$-BACKDOOR SET and DELETION $\mathcal{C}$-BACKDOOR SET are NP-hard for $\mathcal{C} \in \{\text{2CNF}, \text{HORN}, \text{RHORN}\}$.

Parameterized Complexity. Some base classes allow an efficient detection of strong backdoor sets if the size of the backdoor sets is assumed to be small. Nishimura et al. (2004) show that STRONG 2CNF-BACKDOOR SET and STRONG HORN-BACKDOOR SET are *fixed-parameter tractable* in the sense that one can detect a backdoor set of size at most k in time $\mathcal{O}(f(k)\, n^c)$, where f denotes a computable function, n denotes the input size of the given CNF formula, and c is a constant (independent of k and n). For background on fixed-parameter tractability see, e.g., (Downey & Fellows 1999; Flum & Grohe 2006; Niedermeier 2006).

Until recently the parameterized complexities of STRONG/DELETION RHORN-BACKDOOR SET were open. Razgon and O'Sullivan (2008) show that MAX-2-SAT

parameterized by the number of clauses that remain unsatisfied is fixed-parameter tractable, a problem known to be equivalent to DELETION RHORN-BACKDOOR SET under fpt-reductions. On the other hand, we recently found an fpt-reduction from CLIQUE to STRONG RHORN-BACKDOOR SET which makes fixed-parameter tractability of the latter problem unlikely.

Decision Trees. A *binary decision tree* is a rooted binary tree T. Every node of T is either a leaf or has exactly two children. The nodes of T, except for the root, are labeled with literals such that the following conditions are satisfied: (i) Two nodes v_0 and v_1 with the same parent are labeled with complimentary literals x and $\neg x$, respectively; and (ii) the labels of nodes on a path from the root to a leaf do not contain the same literal twice, nor a complementary pair of literals.

We write $|T|$ for the number of leaves of T and define its *size* s as the binary logarithm of its number of leaves, i.e., $s = \log_2 |T|$ (this definition is convenient for comparing the sizes of backdoor trees and backdoor sets).

For a binary decision tree T we write $var(T)$ for the set of variables occurring negated or unnegated as labels of T. For a node v of T, $var_T(v)$ denotes the set of variables occurring negated or unnegated as labels on the path from the root to v; we will omit the subscript T if the context allows it. We associate with each node v of a binary decision tree T a truth assignment $\tau_v : var(v) \to \{0,1\}$ which sets exactly those literals to true that appear as labels on the path from the root to v.

Lemma 1. *For every binary decision tree T, it holds that $|var(T)| \leq |T| - 1$.*

Backdoor Trees. Let $\mathcal{C}$ be a base class, F a CNF formula, and T a binary decision tree with $var(T) \subseteq var(F)$. Then T is a *$\mathcal{C}$-backdoor tree of F* if $F[\tau_v] \in \mathcal{C}$ for every leaf v of T. A $\mathcal{C}$-backdoor tree of F with the smallest number of leaves is a *smallest* $\mathcal{C}$-backdoor tree of F.

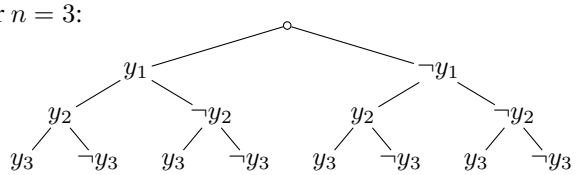
For every base class $\mathcal{C}$ we consider the following parameterized problem:

$\mathcal{C}$-BACKDOOR TREE. *Instance:* A CNF formula F and a non-negative integer k. *Parameter:* The integer k. *Question:* Has F a $\mathcal{C}$-backdoor tree with at most k leaves?

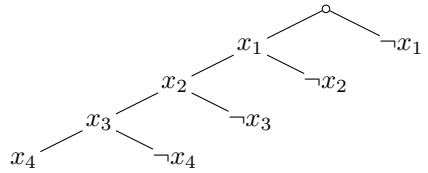
Example 1. Consider the CNF formula F with variables $x_1, \ldots, x_{2n}$, $y_1, \ldots, y_n$, and $z_1, \ldots, z_n$, consisting of all clauses of the form

$$\{y_i, \neg x_1, \ldots, \neg x_{2i-2}, x_{2i-1}, \neg x_{2i}, \ldots, \neg x_{2n}\},$$
$$\{y_i, \neg x_1, \ldots, \neg x_{2i-1}, x_{2i}, \neg x_{2i+1}, \ldots, \neg x_{2n}\},$$
$$\{\neg y_i, z_i\},$$

for $1 \leq i \leq n$. The set $B = \{y_1, \ldots, y_n\}$ is a strong HORN-backdoor set (in fact, B is the smallest possible). However, every HORN-backdoor tree T with $var(T) = \{y_1, \ldots, y_n\}$ has 2^n leaves; the following figure shows a tree for $n = 3$:



On the other hand, the formula F has a HORN-backdoor tree T' with only $2n + 1$ leaves where $var(T') = \{x_1, \ldots, x_{2n}\}$; the following figure shows a tree for $n = 2$:



Note that $B = var(T)$ and $B' = var(T')$ are strong HORN-backdoor sets of F (in fact, B is the smallest possible). On the other hand, T' is a smallest HORN-backdoor tree, having exponentially fewer leaves than T. Thus, when we want to minimize the number of leaves of backdoor trees, we must not restrict ourselves to variables of a smallest strong backdoor set.

Complexity Results

Our first result gives a fundamental link between strong backdoor sets and backdoor trees.

Lemma 2. *Let $\mathcal{C}$ be a base class, F a CNF formula and T a $\mathcal{C}$-backdoor tree of F. Then $var(T)$ is a strong $\mathcal{C}$-backdoor set of F.*

Proof. Let $\tau : var(T) \to \{0,1\}$ be an arbitrarily chosen truth assignment. Starting from the root we trace out a path in T containing only nodes labeled with literals that are true under τ; let v be the leaf at the end of the path. Evidently τ_v is the restriction of τ to $var(v)$. Since T a $\mathcal{C}$-backdoor tree of F, $F[\tau_v] \in \mathcal{C}$, and since $\mathcal{C}$ is self-reducible, also $F[\tau] \in \mathcal{C}$. Hence $var(T)$ is a strong $\mathcal{C}$-backdoor set of F. □

The next result is an immediate consequence of Lemma 1 and Lemma 2.

Proposition 3. *Let $\mathcal{C}$ be a base class and F a CNF formula. If B is a smallest strong $\mathcal{C}$-backdoor set of F and T is a smallest $\mathcal{C}$-backdoor tree of F, then*

$$|B| + 1 \leq |T| \leq 2^{|B|}.$$

Proposition 4. *For $\mathcal{C} \in \{2\text{CNF}, \text{HORN}, \text{RHORN}\}$, the (non-parameterized) problem $\mathcal{C}$-BACKDOOR TREE is* NP*-hard.*

Proof. Case $\mathcal{C} =$ HORN. We reduce from the NP-hard vertex cover problem. From a given graph $G = (V, E)$ we construct a monotone CNF formula $F = \{\, \{u, v\} : uv \in E \,\}$. Nishimura et al. (2004) show that G has a vertex cover of size k if and only if F has a strong HORN-backdoor set of size k. Now we put $F' = \{\, C' : C \in F \,\}$, where $C' = C \cup \{\, \neg x : x \in V \setminus C \,\}$. It is easy to see that F and F' have exactly the same strong HORN-backdoor sets. However, every strong HORN-backdoor set B of F' gives rise to a HORN-backdoor tree with $|B| + 1$ leaves. In view of Lemma 2, it follows that G has a vertex cover of size k if and only if F' has a HORN-backdoor tree with $k + 1$ leaves.

Case $\mathcal{C} =$ RHORN. We proceed similarly as above and obtain from F' the CNF formula F'' by adding for every $v \in V$ the clauses $\{\neg v, x_v\}$ and $\{\neg v, \neg x_v\}$, where x_v is a new variable. Now flipping the polarity of any variable v

introduces a non-Horn clause and we need to put v or x_v into the backdoor set. It follows that G has a vertex cover of size k if and only if F'' has an RHORN-backdoor tree with $k+1$ leaves.

Case $\mathcal{C} = 2\text{CNF}$. Let F^* denote the CNF formula obtained from F by adding to every clause $C = \{u, v\}$ a third variable x_C. Nishimura et al. (2004) show that G has a vertex cover of size k if and only if F^* has a strong 2CNF-backdoor set of size k. We can use the same construction as in the first case to extend this reduction to backdoor trees. □

The above hardness results are contrasted if the number of leaves of the backdoor tree is assumed to be small. We show that the problem is fixed-parameter tractable for the classes 2CNF and HORN. We need the following definitions and lemmas. We say that a base class $\mathcal{C}$ admits a *loss-free kernelization*[1] if there exists a polynomial-time algorithm that, given a CNF formula F and an integer k, either correctly decides that F has no strong $\mathcal{C}$-backdoor set of size at most k, or computes a set $X \subseteq var(F)$ such that the following conditions hold: (i) X contains all minimal strong $\mathcal{C}$-backdoor sets of F of size at most k; and (ii) the size of X is bounded by a computable function that depends on k only.

Proposition 5. *$\mathcal{C}$-BACKDOOR TREE is fixed-parameter tractable for every base class $\mathcal{C}$ that admits a loss-free kernelization.*

Proof. Let (F, k) be an instance of $\mathcal{C}$-BACKDOOR TREE. We may assume that $k \geq 2$ since otherwise we only need to check whether $F \in \mathcal{C}$ which can be done in polynomial time by the definition of a base class. We apply loss-free kernelization to $(F, k-1)$ and compute (in polynomial time) the set X. If F has a $\mathcal{C}$-backdoor tree T with at most k leaves, then, by Lemma 2, $var(T)$ is a strong $\mathcal{C}$-backdoor set. In that case we have $|var(T)| \leq k-1$ by Lemma 1, hence $var(T) \subseteq X$. The number of binary decision trees T that satisfy $var(T) \subseteq X$ is bounded by a computable function f of k since by definition of a loss-free kernelization $|X|$ is bounded in terms of k. Hence we need to check for at most $f(k)$ binary decision trees T whether T is a $\mathcal{C}$-backdoor tree of F with at most k leaves. For each T this check involves at most k times testing for membership in $\mathcal{C}$ and is therefore feasible in polynomial time. □

Next we show that the base classes HORN and 2CNF admit a loss free kernelization, and consequently BACKDOOR TREE is fixed-parameter tractable for these two classes by Proposition 5. For the loss-free kernelization we utilize a general algorithm that kernelizes instances of the hitting set problem. For that we need the following notions.

Let $S = \{X_1, \ldots, X_m\}$ be a set of finite sets over a universe $V(S) = \bigcup_{i=1}^{m} X_i$ of elements. We will refer to S as a *set system*. A set $H \subseteq V(S)$ is a *hitting set* of S if $H \cap X_i \neq \emptyset$ for all $1 \leq i \leq m$.

[1] We call the kernelization "loss-free" to distinguish it from kernelizations usually considered in parameterized complexity where one would have the following strictly weaker condition: (i') If there exists a strong $\mathcal{C}$-backdoor set of F of size at most k, then there exists one that is a subset of X.

Algorithm $\mathsf{A}(S, d, k)$
if $d > 1$ then for all $x \in V(S)$ do
  $T(x) := \{ X \in S : x \in X \}$
  $S(x) := \{ X \setminus \{x\} : X \in T(x) \}$
  $S'(x) := \mathsf{A}(S(x), d-1, k)$
  $T'(x) := \{ X \cup \{x\} : X \in S'(x) \}$
  $S := (S \setminus T(x)) \cup T'(x)$
if $|V(S)| \leq \sum_{i=1}^{d} k^i$ then return S
else return $\{\emptyset\}$

Figure 1: Algorithm for Lemma 6.

Lemma 6. *Let $d, k \geq 1$ be integers. Given a set system S with $\max_{X \in S} |X| \leq d$ and $|V(S)| = n$, then algorithm* A *of Figure 1 computes in time $\mathcal{O}(n^d)$ a set system S' such that (i) $|V(S')| \leq \sum_{i=1}^{d} k^i$, and (ii) every minimal hitting set H of S with $|H| \leq k$ is a subset of $V(S')$.*

Owing to space limitations we omit the proof.

Lemma 7. *The base classes* HORN *and* 2CNF *admit loss-free kernelizations (with loss-free kernels of size $k^2 + k$ and $k^3 + k^2 + k$, respectively).*

Proof. (Sketch.) Algorithm A of Figure 1 can be used to compute loss-free kernelizations for HORN and 2CNF by applying it to the set systems $\{ X : X \subseteq var(C) \text{ with } |X| = 3 \text{ for some } C \in F \}$ and $\{ \{u, v\} : u, v \in C \text{ with } u \neq v \text{ for some } C \in F \}$, respectively. □

By combining Proposition 5 and Lemma 7 we get the main result of this section.

Theorem 8. *The problems* HORN-BACKDOOR TREE *and* 2CNF-BACKDOOR TREE *are fixed-parameter tractable.*

Treewidth is a graph parameter that indicates in a certain sense the 'tree likeness' of a graph. Bounding the treewidth of instances is a general method for achieving tractability and is used in many different areas, see, e.g., (Gottlob, Pichler, & Wei 2006; Gottlob & Szeider 2006). Treewidth can be applied to SAT via certain graphs associated with CNF formulas, for example via the *primal graph* obtained by considering variables as vertices and by making two variables adjacent if they occur (negated or unnegated) together in a clause. Time and space requirements for solving instances of primal treewidth k are similar to the respective requirements for instances with backdoor trees of size k (Samer & Szeider 2007b).

The next result indicates that treewidth and the size of backdoor sets/trees are orthogonal to each other.

Proposition 9. *There are CNF formulas with arbitrarily large primal treewidth that have* HORN*-backdoor sets of constant size. Conversely, there are CNF formulas with constant primal treewidth that require arbitrarily large* RHORN*-backdoor trees.*

Proof. (Sketch.) The first part follows, for example, by considering negative CNF formulas (all literals are negative). For the second part consider the variable-disjoint union of n formulas that are not renamable Horn and have constant treewidth k. This gives a CNF formula of treewidth k that has no RHORN-backdoor tree with fewer than n leaves. □

Algorithms and Experiments

We have computed upper bounds for the minimal number of leaves in backdoor trees for various benchmark sets using a variant of algorithm $\mathsf{H}_\mathcal{C}$ given in Figure 2.

The algorithm calls as subroutine an algorithm $\mathsf{S}_\mathcal{C}$ that computes an upper bound on the size of a smallest strong $\mathcal{C}$-backdoor set. With the parameter d one can bound the *recursion depth* for $\mathsf{H}_\mathcal{C}$; if depth d is reached the generic upper bound provided by Proposition 3 is used (thus, if the algorithm is called with $d = 0$, it returns $2^{\mathsf{S}_\mathcal{C}(F)}$). *Empty clause detection* (Dilkina, Gomes, & Sabharwal 2007) can be included by adding the line "if $\emptyset \in F$ then return 1" on top. *Pure literal detection* can be included similarly by the line "if F contains a pure literal ℓ then return $\mathsf{H}_\mathcal{C}(F[\ell = 1])$."

We obtain S_{HORN} and $\mathsf{S}_{\text{RHORN}}$ by computing vertex covers in certain graphs that we associate with the given CNF formula. With standard greedy and approximation techniques one can compute quickly vertex covers that are not far from optimal (Vazirani 2001); in fact, the graph associated with RHORN contains a perfect matching and admits therefore better vertex cover approximations (Chen & Kanj 2005). Therefore, the base classes HORN and RHORN are particularly suited for the algorithmic framework provided by $\mathsf{H}_\mathcal{C}$.

G_F is the graph on the variables of F where two variables u, v are adjacent if and only if $u, v \in var(C)$ for some clause $C \in F$ (G_F is related to the set system S_{HORN} as considered in the previous section). G'_F is the graph on the literals of F where two literals u, v are adjacent if and only if they are complementary or $u, v \in C$ for some clause $C \in F$. The following two lemmas can be easily verified.

Lemma 10. *A set $B \subseteq var(F)$ is a strong* HORN*-backdoor set of F if and only if B is a vertex cover of G_F.*

Lemma 11. *A set $B \subseteq var(F)$ is a deletion* RHORN*-backdoor set of F if and only if there exists a vertex cover VC of G'_F such that $B = \{\, x \in var(F) : x, \neg x \in VC \,\}$.*

Empirical Results. We have applied a variant of algorithm $\mathsf{H}_\mathcal{C}$ to real-world SAT instances from car configuration (Sinz, Kaiser, & Küchlin 2003) (these classes have also been used by Dilkina et al. (2007)) and random 3SAT instances of various clause/variable densities. In our implementation we use empty clause and pure literal detection and we chose decision variables with preference to variables that belong to small backdoor sets. The results of our experiments are given in Figure 3.

Considering HORN as the base class we observe a difference of up to over 6% (relative to the total number of instance variables) between the size of backdoor sets and the size of backdoor trees, which is remarkable in view of the logarithmic scale used for measuring the size of backdoor trees. For random instances this set/tree difference ranges from 1.55% for low density instances to 0.43% for high density instances. Considering RHORN as the base class, where already backdoor sets are significantly smaller than for HORN, also the set/tree differences are smaller: between 0.66% and 1.62% for structured instances, and between 0.57% and virtually 0% for random instances.

Algorithm $\mathsf{H}_\mathcal{C}(F, d)$
if $\mathsf{S}_\mathcal{C}(F) = 0$ or $d = 0$ then return $2^{\mathsf{S}_\mathcal{C}(F)}$
else for all $x \in var(F)$ do
  $f(x) := \mathsf{S}_\mathcal{C}(F[x = 0]) + \mathsf{S}_\mathcal{C}(F[x = 1])$
pick $x \in var(F)$ with minimal $f(x)$
return $\mathsf{H}_\mathcal{C}(F[x = 0], d - 1) + \mathsf{H}_\mathcal{C}(F[x = 1], d - 1)$

Figure 2: Algorithm that computes an upper bound on the number of leaves of a smallest $\mathcal{C}$-backdoor tree.

More specifically, for random instances the size of backdoor sets and the size of backdoor trees appear to be both monotonically increasing with the density, the latter faster than the former; we cannot observe a threshold phenomenon. For both structured as well as random instances, we observe a significantly larger set/tree difference for base class HORN than for base class RHORN.

For comparison we have also included upper bounds on the *primal treewidth* (Samer & Szeider 2007b) (see the rightmost column of the table in Figure 3). In view of the theoretical orthogonality of the parameters as expressed in Proposition 9, it is interesting to observe a strong positive correlation between treewidth and the size of backdoor sets/trees in the empirical data. This indicates that problem hardness is reflected in several parametric dimensions.

Acknowledgments

We thank Daniele Pretolani for the inspiring discussions. Research supported by the EPSRC, project EP/E001394/1.

References

Chen, J.; and Kanj, I. A. 2005. On approximating minimum vertex cover for graphs with perfect matching. *Theoret. Comput. Sci.* 337(1-3):305–318.

Crama, Y.; Ekin, O.; and Hammer, P. L. 1997. Variable and term removal from Boolean formulae. *Discr. Appl. Math.* 75(3):217–230.

Dilkina, B. N.; Gomes, C. P.; and Sabharwal, A. 2007. Tradeoffs in the complexity of backdoor detection. In *Proc. CP'07*, LNCS 4741, 256–270. Springer-Verlag.

Downey, R. G.; and Fellows, M. R. 1999. *Parameterized Complexity*. Springer-Verlag.

Flum, J.; and Grohe, M. 2006. *Parameterized Complexity Theory*. Springer-Verlag.

Gottlob, G.; Pichler, R.; and Wei, F. 2006. Bounded treewidth as a key to tractability of knowledge representation and reasoning. In *Proc. AAAI'06*, 250–256. AAAI Press.

Gottlob, G.; and Szeider, S. 2006. Fixed-parameter algorithms for artificial intelligence, constraint satisfaction, and database problems. *The Computer Journal*. doi:10.1093/comjnl/bxm056.

Impagliazzo, R.; Paturi, R.; and Zane, F. 2001. Which problems have strongly exponential complexity. *J. of Computer and System Sciences* 63(4):512–530.

instance set	number of variables	number of clauses	HORN bd set (%)	HORN bd tree (%)	RHORN bd set (%)	RHORN bd tree (%)	treewidth (%)
S1: C168_FW_SZ	1583	5646.80	15.36	9.21(-6.15)	3.06	2.40(-0.66)	6.25
S2: C168_FW_UT	1804	7489.25	25.00	19.57(-5.43)	5.82	4.84(-0.98)	6.10
S3: C170_FR_SZ	1528	4998.25	10.95	7.91(-3.03)	3.89	3.13(-0.76)	3.14
S4: C202_FS_SZ	1556	6231.86	14.08	12.00(-2.09)	5.21	4.45(-0.76)	4.85
S5: C202_FW_SZ	1561	8929.10	16.94	14.34(-2.61)	7.19	6.23(-0.96)	7.56
S6: C202_FW_UT	1820	11352	23.79	21.37(-2.42)	8.52	7.57(-0.94)	5.38
S7: C208_FA_SZ	1516	5285	11.21	9.89(-1.33)	4.86	4.18(-0.67)	3.17
S8: C208_FA_UT	1805	7335.50	24.04	22.14(-1.90)	7.76	6.13(-1.62)	10.80
S9: C208_FC_RZ	1513	5567	11.24	9.70(-1.54)	5.02	4.33(-0.69)	3.17
S10: C208_FC_SZ	1513	5575	11.59	9.92(-1.66)	5.18	4.47(-0.71)	3.17
S11: C210_FS_RZ	1607.33	5764.33	12.71	10.22(-2.49)	4.62	3.80(-0.82)	3.17
S12: C210_FS_SZ	1607.14	5810.71	12.91	10.36(-2.54)	4.81	3.98(-0.83)	3.30
S13: C210_FW_RZ	1628.33	7408.33	13.78	11.28(-2.50)	5.28	4.53(-0.75)	4.05
S14: C210_FW_SZ	1628	7546.22	15.54	12.90(-2.64)	6.14	5.27(-0.87)	6.21
S15: C210_FW_UT	1891	9720	22.18	19.62(-2.56)	7.88	6.88(-1.00)	5.00
S16: C220_FV_SZ	1530	4882.50	11.63	10.33(-1.31)	3.62	2.79(-0.82)	6.88
R1: $\rho = 2.0$	800	1600	46.50	44.95(-1.55)	24.54	23.97(-0.57)	48.13
R2: $\rho = 2.5$	800	2000	49.75	48.54(-1.21)	30.25	29.81(-0.44)	54.29
R3: $\rho = 3.0$	800	2400	53.38	52.41(-0.97)	35.38	35.01(-0.37)	59.29
R4: $\rho = 3.5$	800	2800	56.58	55.90(-0.69)	40.92	40.72(-0.20)	63.46
R5: $\rho = 4.0$	800	3200	58.63	58.05(-0.57)	44.67	44.41(-0.25)	66.13
R6: $\rho = 4.5$	800	3600	60.17	59.71(-0.45)	48.00	47.78(-0.22)	69.13
R7: $\rho = 5.0$	800	4000	62.29	61.77(-0.52)	51.33	51.09(-0.24)	71.54
R8: $\rho = 5.5$	800	4400	63.96	63.49(-0.47)	54.21	54.00(-0.21)	73.75
R9: $\rho = 6.0$	800	4800	66.00	65.57(-0.43)	56.34	56.34(-0.00)	75.21

Figure 3: Experimental results for backdoor sets and backdoor trees of various classes of benchmark sets. S1-S16 are benchmark sets with structured instances from logistics planning, R1-R9 are benchmark sets with random 3CNF instances of fixed clause/variable density ρ. Computed upper bounds on the size of backdoor sets and backdoor trees are shown as average percentage of the number of variables of the instance, the set/tree difference is given in parenthesis. For example, class S1 has, on average, strong HORN-backdoor sets with 243 variables ($\approx 15.36\%$), which reduces an instance (theoretically) to 2^{243} HORN instances. Class S1 has, on average, HORN-backdoor trees with 2^{146} leaves ($s = 146 \approx 9.21\%$), which reduces an instance (theoretically) to 2^{146} HORN instances. Backdoor trees are computed with a variant of algorithm $\mathsf{H}_{\mathcal{C}}$ and depth $d = 10$. The last column provides upper bounds on the primal treewidth as average percentage of the number of variables of the instance.

Interian, Y. 2003. Backdoor sets for random 3-SAT. In *Proc. SAT'03*, 231–238. Informal Proc.

Kilby, P.; Slaney, J. K.; Thiébaux, S.; and Walsh, T. 2005. Backbones and backdoors in satisfiability. In *Proc. AAAI'05*, 1368–1373. AAAI Press.

Lynce, I.; and Marques-Silva, J. P. 2004. Hidden structure in unsatisfiable random 3-SAT: An empirical study. In *Proc. ICTAI'04*, 246–251. IEEE Computer Society.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford Univ. Press.

Nishimura, N.; Ragde, P.; and Szeider, S. 2004. Detecting backdoor sets with respect to Horn and binary clauses. In *Proc. SAT'04*, 96–103. Informal Proc.

Nishimura, N.; Ragde, P.; and Szeider, S. 2007. Solving #SAT using vertex covers. *Acta Informatica* 44(7-8):509–523.

Razgon, I.; and O'Sullivan, B. 2008. Almost 2-SAT is fixed-parameter tractable. arXiv:0801.1300.

Ruan, Y.; Kautz, H. A.; and Horvitz, E. 2004. The backdoor key: A path to understanding problem hardness. In *Proc. AAAI'04*, 124–130. AAAI Press.

Samer, M.; and Szeider, S. 2007a. Backdoor sets of quantified Boolean formulas. In *Proc. SAT'07*, LNCS 4501, 230–243. Springer-Verlag.

Samer, M.; and Szeider, S. 2007b. Algorithms for propositional model counting. In *Proc. LPAR'07*, LNCS 4790, 484–498. Springer-Verlag.

Sinz, C.; Kaiser, A.; and Küchlin, W. 2003. Formal methods for the validation of automotive product configuration data. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 17(1):75–97.

Szeider, S. 2005. Backdoor sets for DLL subsolvers. *J. of Automated Reasoning* 35(1-3):73–88.

Vazirani, V. V. 2001. *Approximation Algorithms*. Springer-Verlag.

Williams, R.; Gomes, C.; and Selman, B. 2003a. Backdoors to typical case complexity. In *Proc. IJCAI'03*, 1173–1178. Morgan Kaufmann.

Williams, R.; Gomes, C.; and Selman, B. 2003b. On the connections between backdoors, restarts, and heavy-tailedness in combinatorial search. In *Proc. SAT'03*, 222–230. Informal Proc.