In this paper, we introduce a notion of backdoors to Reiter's propositional default logic and study structural properties of it. Also we consider the problems of backdoor detection (parameterised by the solution size) as well as backdoor evaluation (parameterised by the size of the given backdoor), for various kinds of target classes (CNF, KROM, MONOTONE) and all SCHAEFER classes. Also, we study generalisations of HORN-formulas, namely, QHORN, RHORN, as well as DUALHORN. For these classes, we also classify the computational complexity of the implication problem. We show that backdoor detection is fixed-parameter tractable for the considered target classes, and prove a complete trichotomy for backdoor evaluation. The problems are either fixed-parameter tractable, para-DeltaP2-complete, or para-NP-complete, depending on the target class.

CCS Concepts: · Theory of computation → Problems, reductions and completeness; Complexity theory and logic; Parameterized complexity and exact algorithms; · Computing methodologies → Nonmonotonic, default reasoning and belief revision.

Additional Key Words and Phrases: Parameterised Complexity, Default Logic, Backdoors

1 INTRODUCTION

In the area of non-monotonic logic one aims to find formalisms that model human-sense reasoning. It turned out that this kind of reasoning is quite different from classical deductive reasoning as in the classical approach the addition of information always leads to an increase of derivable knowledge. Yet, intuitively, human-sense reasoning does not work in that way: the addition of further facts may violate previous assumptions and can consequently significantly decrease the amount of derivable conclusions. Consequently, in contrast to the classical process the behaviour of human-sense reasoning is non-monotonic. In the 1980s, several kinds of formalisms have been introduced, most notably, circumscription [64], default logic [83], autoepistemic logic [74], and non-monotonic logic [66]. A good overview of this field is given by Marek and Truszczyński [63]. One might wrongly expect that these logics are rather rusted and not really focussed in recent works [1, 2, 52, 56, 82, 92, 97].

In this paper, we focus on Reiter’s default logic (DL), which has been introduced in 1980 [83] and is one of the most fundamental formalism to model human-sense reasoning. DL extends the usual logical derivations by rules of default assumptions (default rules). Informally, default rules follow the format “in the absence of contrary information, assume . . .”. Technically, these patterns are taken up in triples of formulas $\alpha \rightarrow \beta \gamma$, which express “if prerequisite $\alpha$ can be deduced and justification $\beta$ is never violated then assume conclusion $\gamma$”. Default rules can be used to enrich calculi of different kinds of logics. Here, we consider a variant of propositional formulas, namely, formulas in conjunctive normal form (cnf). A key concept of DL is that an application of default rules must not lead to an inconsistency if conflicting rules are present, instead such rules should be avoided if possible. This concept results in the notion of stable extensions, which can be seen as a maximally consistent view of an
agent with respect to his knowledge base together in combination with its set of default rules. The corresponding decision problem, i.e., the extension existence problem, then asks whether a given default theory has a *consistent stable extension*, and is the problem of our interest. The computationally hard part of this problem lies in the detection of the order and "applicability" of default rules, which is a quite challenging task as witnessed by its $\Sigma_2^p$-completeness. In 1992, Gottlob showed that many important decision problems, beyond the extension existence problem, of non-monotonic logics are complete for the second level of the polynomial hierarchy [46] and accordingly are of high intractability.

A prominent approach to understand the intractability of a problem is to use the framework of parameterised complexity, which was introduced by Downey and Fellows [18, 19]. The main idea is to fix a certain structural property (the parameter) of a problem instance and to consider the computational complexity of the problem in dependency of the parameter. Then ideally, the complexity drops and the problem becomes solvable in polynomial time when the parameter is fixed. Notice that there are two kinds of such a polynomial runtime depending on how the parameter is related to the input length: $f(k) \cdot n^O(1)$ versus $n^{f(k)}$. In both kinds, the runtime becomes polynomial, whenever the parameter is assumed to be constant, but the quality of these two runtimes differs dramatically. Problems allowing for runtimes of the first kind are called *fixed-parameter tractable* and the corresponding parameterised complexity class that contains all fixed-parameter tractable problems, is called FPT. Problems with runtimes of the second kind are situated in the complexity class XP, which is very large class containing also FPT and another entire hierarchy, the so-called W-hierarchy. For example, for the satisfiability problem (Sat) asks, given a propositional formula whether the formula is satisfiable [27]. One (naive) parameter for Sat is the number of variables of the formula. Then, for a given formula $\varphi$ of size $n$ and $k$ variables its satisfiability can be decided in time $O(n \cdot 2^k)$, i.e., in polynomial runtime in $n$ if $k$ is assumed to be fixed.

The invention of new parameters can be quite challenging, however, Sat has so far been considered under many different parameters [11, 77, 85, 93]. A concept that provides a parameter and has been widely used in theoretical investigations of propositional satisfiability are backdoors [45, 50, 88, 89, 95]. The size of a backdoor can be seen as a parameter with which one tries to exploit a small distance of a formula from being tractable. Formally, there exist several notions of backdoors, e.g., weak, strong, and deletion backdoors. In this paper, we consider strong backdoors. Specifically, given a class $\mathcal{F}$ of formulas and a formula $\varphi$, a subset $B$ of the variables of $\varphi$ is a strong $\mathcal{F}$-backdoor if the formula $\varphi$ under every truth assignment over $B$ yields a formula that belongs to the class $\mathcal{F}$. Using backdoors usually consists of two phases:

(i) finding a backdoor (backdoor detection) and
(ii) using the backdoor to solve the problem (backdoor evaluation).

If $\mathcal{F}$ is a class of formulas where $\text{Sat}$ is tractable and the backdoor detection problem is in FPT when parameterised by the size of the backdoor, we can find such a backdoor in polynomial time in the input size and exponential in the size of the backdoor. For example, this is the case for the class that contains all Horn formulas or the class that contains all Krom formulas. Then, we use the backdoor to solve the problem, where the evaluation problem is also parameterised in the size of the backdoor. Hence, we can immediately conclude that Sat is fixed-parameter tractable when parameterised by the size of a smallest strong $\mathcal{F}$-backdoor.

Related Work. Backdoors for propositional satisfiability have been introduced by Williams, Gomes, and Selman [95, 96]. The concept of backdoors has recently been lifted to some non-monotonic formalisms as abduction [81], answer set programming [37, 38], and argumentation [21]. Beyond the classification of Gottlob [46], the complexity of fragments, in the sense of Post’s lattice, has been considered by Beyersdorff et al. extensively for default logic [7], and for autoepistemic logic by Creignou et al. [16]. Also parameterised analyses of non-monotonic logics in the spirit of Courcelle’s Theorem [12, 13] have recently been considered by Meier et al. [72]. Furthermore, Gottlob et al. studied treewidth as a parameter for various non-monotonic logics [48] and also considered a more CSP focused non-monotonic context within the parameterised complexity setting [49]. Gaspers, Ordyniak, and

ACM Trans. Comput. Logic
Szeider [44] surveyed results on CSP and backdoors. Gasper et al. [42] investigated on heterogeneous target classes, which are formed by taking the union of existing target classes. Ganian, Ramanujan, and Szeider [41] established a parameter that combines treewidth and backdoors for SAT. Recently, Meier et al. [71] considered the concept of backdoors in linear temporal logic. Fichte, Hecher, and Schindler [33] considered treewidth for default logic, which is an orthogonal parameter to backdoors. There exists several recent research directions into the parameterised complexity of nonmonotonic formalisms [29, 30, 35, 60]. Default logic for propositional logic, as studied in this paper, has been of interest not only in the recent years [3–5, 58, 59].

**Contribution.** In this paper, we introduce a notion of backdoors to propositional default logic and study structural properties therein. Then we investigate the parameterised complexity of the problems of backdoor detection (parameterised by the size of the expected backdoor) and evaluation (parameterised by the size of the given backdoor), with respect to the most important classes of CNF-formulas, e.g., CNF, KROM, HORN, MONOTONE, and POSITIVE-UNIT. Informally, given a formula $\varphi$ and an integer $k$, the detection problem asks whether there exists a backdoor of size $k$ for $\varphi$. Backdoor evaluation then exploits the distance $k$ for a target formula class to solve the problem for the starting formula class with a “simpler” complexity. Our classification shows that detection is fixed-parameter tractable for all considered target classes. However, for backdoor evaluation starting at CNF the parameterised complexity depends, as expected, on the target class: the parameterised complexity then varies between para-$\Delta^p_2$ (MONOTONE), para-NP (KROM, HORN), and FPT (POSITIVE-UNIT).

**Prior Work.** A preliminary version of this article appeared in the proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing 2016 [36]. The present paper provides a higher level of detail, in particular full proofs and more examples. The new material includes additional results on the target classes HORN and lower bounds. Furthermore, we present a full classification of the implication problem regarding generalisations of HORN-classes, namely, $q$HORN, $r$HORN, as well as DUALHORN (see Sect. 2.2). These results are used in turn to provide a complete study of these classes with respect to our backdoor problems (detection as well as evaluation). Finally, we classify all SCHAFFER-cases.

## 2 Preliminaries

We assume familiarity with standard notions in computational complexity, the complexity classes $P$ and $NP$ as well as the polynomial hierarchy. For more detailed information, we refer to other standard sources [19, 40, 78].

**Parameterised Complexity.** We follow the notion by Flum and Grohe [39]. A parameterised (decision) problem $L$ is a subset of $\Sigma^* \times \mathbb{N}$ for some finite alphabet $\Sigma$.

**Definition 2.1 (Precomputation on the parameter).** Let $C$ be a classical complexity class, then para-$C$ consists of all parameterised problems $L \subseteq \Sigma^* \times \mathbb{N}$, for which there exists an alphabet $\Sigma'$, a computable function $f: \mathbb{N} \to \Sigma'^*$, and a (classical) problem $L' \subseteq \Sigma' \times \Sigma'^*$ such that

(i) $L' \in C$, and

(ii) for all instances $(x, k) \in \Sigma^* \times \mathbb{N}$ of $L$ we have $(x, k) \in L$ if and only if $(x, f(k)) \in L'$.

For the complexity class P, we write FPT instead of para-P. We call a problem in FPT fixed-parameter tractable and the runtime $f(k) \cdot |x|^{O(1)}$ also fpt-time. Additionally, the parameterised counterparts of NP and $\Delta^p_2 = P^{NP}$, which are denoted by para-NP and para-$\Delta^p_2$, are relevant in this paper.

**Example 2.2.** Let us consider the parameter "number of variables" for the satisfiability problem SAT. Then, SAT is in FPT, by the obvious brute-force algorithm which runs in time $2^k \cdot |\varphi|$, for a given formula $\varphi$ with $k$ variables.

Now, we will briefly introduce the concept of fpt-approximability, which is a generalisation of the notion of approximation algorithms to the parameterised setting. As this concept is only relevant for Corollary 5.2 and
only implicitly, we will not go into much details here. Let \( r : \mathbb{N} \to \mathbb{R}_{\geq 1} \) be a function, called approximation ratio. An fpt-approximation algorithm is an fpt-algorithm that, given an instance \((x, k)\), either determines that there is no solution of size at most \(k\) or computes a solution of size at most \(k \cdot r(k)\) [20]. Intuitively, if \(k\) is small, then \(k \cdot r(k)\) can be still considered small. We say that a problem is fpt-approximable if it has an fpt-approximation algorithm for some function \(r\). By abuse of notation, we use fpt-approximable also for decision problems, where we are simply interested in deciding whether a solution of size at most \(k \cdot r(k)\) exists.

**Propositional Logic.** Next, we provide some notions from propositional logic. We consider a finite set of propositional variables and use the symbols \(\top\) (true) and \(\bot\) (false). A literal is a variable \(x\) (positive literal) or its negation \(\neg x\) (negative literal). A clause is a finite set of literals, interpreted as the disjunction of these literals. A propositional formula in conjunctive normal form (CNF) is a finite set of clauses, interpreted as the conjunction of its clauses. We denote the class of all CNF-formulas by \(\text{cnf}\). Notice that constants per se do not appear in clauses, but constants can be seen as either atomic tautological/contradictory CNFs, i.e., \(\top\) and \(\bot\), respectively. For a given formula \(\varphi\), we denote by \(\text{Vars}(\varphi)\) the set of its variables. If \(X\) is a set of variables, then \(\text{Lits}(X)\) is the set of all literals of \(X\), that is, it contains \(x, \neg x\) for every \(x \in X\). A clause is \((\text{Dual})\text{Horn}\) [53] if it contains at most one positive (negative) literal, \(\text{Krom}\) if it contains two literals, \(\text{monotone}\) if it contains only positive literals, and \(\text{positive-unit}\) if it contains exactly one positive literal. Furthermore, a clause is \(c\text{-valid}\), for \(c \in \{0, 1\}\), if it contains at least one positive (negative) literal, if \(c = 0\) (if \(c = 1\)). We also consider generalisations of Horn-formulas, namely, \(\text{QHorn}\) [8], and \(\text{RHorn}\) [10] that are defined as follows. A CNF-formula \(\varphi = \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} \ell_{ij}\) is \(\text{QHorn}\) if there exists a certifying function \(\beta: \text{Lits}(\text{Vars}(\varphi)) \to \{0, \frac{1}{2}, 1\}\) with \(\beta(x) = 1 - \beta(\neg x)\) for every \(x \in \text{Vars}(\varphi)\) such that for every clause \(C \in \varphi\) we have that \(\sum_{\ell \in C} \beta(\ell) \leq 1\). (Notice that this function definition does not introduce any kind of three valued logic.)

**Example 2.3 (QHorn).** The formula \(\varphi = \{\{x, y, \neg z\}, \{y, \neg z\}\}\) is \(\text{QHorn}\), as we can choose \(\beta(y) = \beta(z) = \frac{1}{2}\) and \(\beta(x) = 1\). Note that \(\varphi\) is not Horn, as the first clause contains two positive literals.

Given a CNF-formula \(\varphi\) and a subset of its variables \(X \subseteq \text{Vars}(\varphi)\), we say a renaming \(\varphi_X\) is the formula \(\varphi\), where every literal \(\ell\) in \(\varphi\) of a variable in \(X\) is flipped, that is, if \(\ell = x\) (resp., \(\ell = \neg x\)) then the flipped version is \(\neg x\) (resp., \(x\)). We say that \(\varphi\) is \(\text{RHorn}\) (renamable Horn) if there exists a set \(X \subseteq \text{Vars}(\varphi)\) such that \(\varphi_X\) is Horn.

**Example 2.4 (RHorn).** The formula \(\varphi = \{\{x, y\}, \{\neg x, \neg y\}\}\) is \(\text{RHorn}\), as we can choose \(X = \{x\}\) and then \(\varphi_X = \{\{\neg x, y\}, \{x, \neg y\}\}\) is Horn. Note that \(\varphi\) is not Horn, as it contains a clause with two positive literals.

Formally, a formula is \((\text{Dual})\text{Horn}, \text{Krom}, \text{monotone}, \text{positive-unit}, \text{Qhorn}, \text{RHorn}, \text{0-valid}\) and \(1\text{-valid}\) if all its clauses have the respective property. Furthermore, we denote by \(\text{horn}, \text{dualhorn}, \text{krom}, \text{monotone}, \text{positive-unit}, \text{qhorn}, \text{rhorn}, \text{0valid}\) and \(1\text{valid}\) the classes of all formulas that are \((\text{Dual})\text{Horn}, \text{Krom}, \text{monotone}, \text{positive-unit}, \text{Qhorn}, \text{RHorn}, \text{0valid}\) and \(1\text{valid}\) respectively.

From the previous definitions the following containment structure follows:

\[
\text{horn} \subseteq \text{rhorn} \subseteq \text{qhorn} \supseteq \text{krom}.
\]

A formula is in \(\oplus\text{-CNF}\) if it can be written in the form \(\bigwedge_{i=1}^{m} \bigoplus_{j=1}^{n} \ell_{ij}\), that is, the clauses of the CNF are not disjunctions but exclusive-ors. By this, we define the class \text{affine-formulas} as those which can be expressed in \(\oplus\text{-CNF}\). We say that a CNF-formula has a certain property if all its clauses have the property. Finally, we say that a formula is \text{schaefer} if it is either \text{0valid}, \text{1valid}, \text{horn}, \text{dualhorn}, \text{krom}, or \text{affine} (in honor of Thomas J. Schaefer [86]). Table 1 gives an overview on these classes and defines clause forms for some of these classes.

A formula \(\varphi'\) is a subformula of a CNF-formula \(\varphi\) (in symbols \(\varphi' \subseteq \varphi\)) if for each clause \(C' \in \varphi'\) there is some clause \(C \in \varphi\) such that \(C' \subseteq C\). We call a class \(\mathcal{F}\) of CNF-formulas clause-induced if whenever \(F \in \mathcal{F}\), all subformulas \(F' \subseteq F\) belong to \(\mathcal{F}\). Note that all considered target classes in this paper are clause-induced.
Given a formula $\varphi \in \text{cnf}$, and a subset $X \subseteq \text{Vars}(\varphi)$, then a (truth) assignment is a mapping $\theta : X \rightarrow \{0, 1\}$. The truth (evaluation) of propositional formulas is defined in the standard way, in particular, $\theta(\bot) = 0$ and $\theta(\top) = 1$. We extend $\theta$ to literals by setting $\theta(\neg x) = 1 - \theta(x)$ for $x \in X$. By $\mathcal{A}(X)$ we denote the set of all assignments $\theta : X \rightarrow \{0, 1\}$. For simplicity of presentation, we sometimes identify the set of all assignments by $\mathcal{A}(X)$; we set $\theta(x) = \neg \theta(\neg x)$.

Note that any assignment $\theta$ satisfies $\varphi$ if $\theta[\varphi] = \top$, $\varphi$ is satisfiable if there exists an assignment that satisfies $\varphi$, and $\varphi$ is tautological if all assignments $\theta \in \mathcal{A}(X)$ satisfy $\varphi$. Let $\varphi, \psi \in \text{cnf}$ and $X = \text{Vars}(\varphi) \cup \text{Vars}(\psi)$. We write $\varphi \vdash \psi$ if and only if for all assignments $\theta \in \mathcal{A}(X)$ we have that $\theta$ satisfying $\varphi$ implies $\theta$ satisfying $\psi$. Furthermore, we define the deductive closure of $\varphi$ as $\text{Th}(\varphi) := \{ \psi \in \text{cnf} \mid \varphi \vdash \psi \}$.

We denote with $\text{Sat}(\mathcal{F})$ the problem, given a propositional formula $\varphi \in \mathcal{F}$ asking whether $\varphi$ is satisfiable. The problem $\text{Taut}(\mathcal{F})$ is defined over a given formula $\varphi \in \mathcal{F}$ asking whether $\varphi$ tautological.

### 2.1 Default Logic

We follow notions by Reiter [83] and define a default rule $\delta$ as a triple $\frac{\alpha \beta}{\gamma}$; $\alpha$ is called the prerequisite, $\beta$ is called the justification, and $\gamma$ is called the conclusion; we set $\text{prereq}(\delta) := \alpha$, $\text{just}(\delta) := \beta$, and $\text{concl}(\delta) := \gamma$. If $\mathcal{F}$ is a class of formulas, then $\frac{\alpha \beta}{\gamma}$ is an $\mathcal{F}$-default rule if $\alpha, \beta, \gamma \in \mathcal{F}$. An $\mathcal{F}$-default theory $\langle W, D \rangle$ consists of a set of propositional formulas $W \subseteq \mathcal{F}$ and a set $D$ of $\mathcal{F}$-default rules. We sometimes call $W$ the knowledge base of $\langle W, D \rangle$. Whenever we do not explicitly state the class $\mathcal{F}$, we assume it to be $\text{cnf}$.

**Definition 2.5 (Fixed point semantics, [83])**. Let $\langle W, D \rangle$ be a default theory and $E$ be a set of formulas. Then $\Gamma(E)$ is the smallest set of formulas such that:

1. $W \subseteq \Gamma(E)$,
2. $\Gamma(E) = \text{Th}(\Gamma(E))$, and
(3) for each \( \frac{\alpha \beta}{\gamma} \in D \) with \( \alpha \in \Gamma(E) \) and \( \neg \beta \not\in E \), we have that \( \gamma \in \Gamma(E) \).

E is a stable extension of \( \langle W, D \rangle \), if \( E = \Gamma(E) \). An extension is inconsistent if it contains \( \bot \), otherwise it is called consistent.

A definition of stable extensions beyond fixed point semantics, which has been introduced by Reiter [83] as well, uses the principle of a stage construction.

**Proposition 2.6 (Stage construction, [83]).** Let \( \langle W, D \rangle \) be a default theory and \( E \) be a set of formulas. Then define \( E_0 := W \) and

\[
E_{i+1} := \text{Th}(E_i) \cup \left\{ \frac{\alpha \beta}{\gamma} \mid \alpha \in D, \alpha \in E_i \text{ and } \neg \beta \not\in E \right\}.
\]

E is a stable extension of \( \langle W, D \rangle \) if and only if \( E = \bigcup_{i \in \mathbb{N}} E_i \). The set

\[
G = \left\{ \frac{\alpha \beta}{\gamma} \in D \mid \alpha \in E \land \neg \beta \not\in E \right\}
\]

is called the set of generating defaults. If \( E \) is a stable extension of \( \langle W, D \rangle \), then \( E = \text{Th}(W \cup \{ \text{conc}(\delta) \mid \delta \in G \}) \).

**Example 2.7.** Let \( W = \emptyset, W' = \{ x \} \), \( D_1 = \{ \frac{x \vee y}{x \neg y}, \frac{x \neg y}{x \vee y} \} \), and \( D_2 = \{ \frac{x \vee y}{x \neg y}, \frac{x \neg y}{x \vee y} \} \). The default theory \( \langle W, D_1 \rangle \) has only the stable extension \( \text{Th}(W) \). The default theory \( \langle W', D_1 \rangle \) has no stable extension. The default theory \( \langle W', D_2 \rangle \) has the stable extensions \( \text{Th}(\{ x, \neg y \}) \) and \( \text{Th}(\{ x, \neg z \}) \).

The following example illustrates that a default theory might contain "contradicting" default rules that cannot be avoided in the process of determining extension existence. Informally, such default rules prohibit stable extensions. Note that there are also less obvious situations where "chains" of such default rules interact with each other.

**Example 2.8.** Consider \( W' \) and \( D_2 \) from Example 2.7 and let \( D'_2 = D_2 \cup \{ \frac{\neg \beta}{x \vee y} \} \) for some formula \( \beta \). The default theory \( \langle W', D'_2 \rangle \) has no stable extension unless \( W' \cup \{ \neg y \} \models \neg \beta \) or \( W' \cup \{ \neg z \} \models \neg \beta \).

Technically, the definition of stable extensions allows inconsistent stable extensions. However, Marek and Truszczynski showed that inconsistent extensions only occur if the set \( W \) is already inconsistent where \( \langle W, D \rangle \) is the theory of interest [63, Corollary 3.60]. An immediate consequence of this result explains why the interplay between consistency and stability of extensions is more subtle:

(i) if \( W \) is consistent, then every stable extension of \( \langle W, D \rangle \) is consistent, and

(ii) if \( W \) is inconsistent, then \( \langle W, D \rangle \) has a stable extension.

In Case (ii) the stable extension consists of the set \( L \) of all formulas. Consequently, it makes sense to consider only consistent stable extensions as the relevant ones. Moreover, we refer by \( \text{SE}(\langle W, D \rangle) \) to the set of all consistent stable extensions of \( \langle W, D \rangle \).

A main computational problem for DL is the extension existence problem, defined as follows where \( \mathcal{F} \) is a class of propositional formulas:

**Problem:** \( \text{Ext}(\mathcal{F}) \), also called Extension Existence.

**Input:** An \( \mathcal{F} \)-default theory \( \langle W, D \rangle \).

**Question:** Does \( \langle W, D \rangle \) have a consistent stable extension?

The following proposition summarises relevant results for the extension existence problem for certain classes of formulas.

**Proposition 2.9.** (1) \( \text{Ext}(\text{cnf}) \) is \( \Sigma_2^p \)-complete [46].
(2) Ext(horn) is NP-complete [90, 91].

(3) Ext(positive-unit) ∈ P [7].

2.2 The Implication Problem for Horn Variants

The implication problem is an important (sub-)problem when reasoning with default theories. In the following, we first formally introduce the implication problem for classes of propositional formulas, and then state its (classical) computational complexity for the classes horn, krom, as well as the generalisations dualhorn, qhorn, rhorn.

**Problem:** Imp(\(\mathcal{F}\)), also called Implication.

**Input:** A set \(\Phi\) of \(\mathcal{F}\)-formulas and a formula \(\psi\) ∈ \(\mathcal{F}\).

**Question:** Does \(\Phi \models \psi\) hold?

Beyersdorff et al. [7] have considered all Boolean fragments of Imp(\(\mathcal{F}\)) and completely classified its computational complexity concerning the framework of Post’s lattice. To make this concept a bit clearer to the reader, we give some intuition. Post’s lattice refers to so-called clones. Intuitively, a clone is a set of Boolean functions with some properties. For example, the clone of all monotone Boolean functions is the set of all Boolean functions that are monotone. On that level, Beyersdorff et al. have considered restrictions of default theories to certain clones. As a result, such restrictions are on the level of allowed Boolean functions. That is, no particular normal forms of functions are enforced. Accordingly, these results are not fully applicable to our setting. However, since several subclasses of cnf, like horn or krom, use the Boolean functions “\(\land\)”, “\(\neg\)”, and “\(\lor\)”, such classes are unrestricted from the perspective of Post’s lattice. Still, efficient algorithms are known for such classes from propositional satisfiability. The next results state a similar behaviour for the implication problem. Intuitively, we are able to reduce the implication question to separate non-tautology questions which, in turn, can be reduced to satisfiability questions.

The following results will yield several P results for fragments of the implication problem that will be used in the sequel. Beyond the obvious use in our setting, these results are also interesting from a theoretical perspective and might find applications in other contexts that are relevant to the implication problem.

**Lemma 2.10.** Imp(qhorn) ∈ P.

**Proof.** Given a set \(\Phi\) of qhorn-formulas and a formula \(\psi\) ∈ qhorn. Without loss of generality assume that \(\bigwedge_{\varphi \in \Phi} \varphi = \bigwedge_{i=1}^{m} C_i\) and \(\psi = \bigwedge_{j=1}^{n} C'_j\). Then we have that

\[
\langle \Phi, \psi \rangle \in \text{Imp(qhorn)} \iff \left( \bigwedge_{i=1}^{m} C_i, \bigwedge_{j=1}^{n} C'_j \right) \in \text{Imp(qhorn)} \quad (\text{1})
\]

\[
\iff \left( \bigwedge_{i=1}^{m} C_i \right) \rightarrow \left( \bigwedge_{j=1}^{n} C'_j \right) \in \text{Taut} \quad (\text{2})
\]

\[
\iff \left( \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m} C_j \rightarrow C'_i \right) \in \text{Taut} \quad (\text{3})
\]

\[
\forall 1 \leq i \leq n \left( \bigwedge_{j=1}^{m} C_j \rightarrow C'_i \right) \in \text{Taut} \quad (\text{4})
\]

\[
\Rightarrow \exists 1 \leq i \leq n \left( \bigwedge_{j=1}^{m} C_j \rightarrow C'_i \right) \notin \text{Taut} \quad (\text{5})
\]
(1) definition of the implication problem.
(2) expressing implication through the propositional function $\rightarrow$.
(3) $\alpha \rightarrow \beta \land \gamma$ is a tautology if and only if $(\alpha \rightarrow \beta) \land (\alpha \rightarrow \gamma)$ is a tautology.
(4) separated to parallel tautology questions.
(5) $\alpha \land \beta$ is a tautology iff neither $\alpha$ nor $\beta$ is not a tautology.

Now, we can check the last $n$ problems separately by

$$\left( \bigwedge_{j=1}^{m} C_j \rightarrow (\ell_{i,1} \lor \cdots \lor \ell_{i,k_i}) \right) \notin \text{Taut} \iff \left( \bigwedge_{j=1}^{m} C_j \right) [\theta_0] \in \text{Sat(qhorn)},$$

where $\theta_0$ is the assignment such that $\theta_0(\ell) := 0$ for all $\ell \in \{\ell_{i,1}, \ldots, \ell_{i,k_i}\}$. Observe that, if $\ell_{\mu,v} \equiv \neg \ell_{i,v}$ for some $1 \leq \mu \neq v \leq k$, then the implication on the left part of the equivalence is always a tautology. Further note that applying an assignment to a qhorn-formula, that is, considering the reduct of the formula under an assignment, maintains the qhorn-property. As satisfiability of qhorn-formulas is in P [8, 9], the claimed result follows. \qed

The previous proof also works for krom- and rhorn-formulas.

**Corollary 2.11.** $\text{Imp}(C) \in \text{P}$ for $C \in \{\text{krom}, \text{rhorn}, \text{horn}\}$.

**Proof.** Use the same reduction as in the proof of Lemma 2.10. Furthermore, it is known that satisfiability for krom- [55], for rhorn- [10] as well as for horn-formulas [53] is in P. \qed

Also the dualhorn-variant exhibits membership in P.

**Lemma 2.12.** $\text{Imp(}d\text{ualhorn}) \in \text{P}$.

**Proof.** W.l.o.g., we can assume, given two dualhorn-formulas $\varphi, \psi$ that $\text{Vars}(\varphi) = \text{Vars}(\psi)$ (otherwise, extend both formulas with tautological clauses of the kind $(x \land \neg x)$ which are both, horn and dualhorn). Now, note that the classical reduction for the satisfiability problem from dualhorn- to horn-formulas works here as well as it provides a 1:1-correspondence between assignments. That means, we have

$$\langle \Phi, \Psi \rangle \in \text{Imp(dualhorn)} \iff \langle \overline{\Phi}, \overline{\Psi} \rangle \in \text{Imp(horn)},$$

where $\overline{\Phi}$ for a set of formulas $\Phi$ is defined as $\{ \varphi(x_1, \ldots, x_n) \mid \varphi(x_1, \ldots, x_n) \in \Phi \}$. This reduction is computable in constant space. As $\text{Imp(horn)} \in \text{P}$ (see Cor. 2.11) this shows the desired result. \qed

The following corollary shows that the consideration of backdoor evaluation for the extension existence problem starting from all the above investigated horn-variants for default theories is worth studying.

**Corollary 2.13.** $\text{Ext}(C)$ is NP-complete for $C \in \{\text{krom, horn, qhorn, rhorn}\}$.

**Proof.** As Corollary 2.11 shows that $\text{Imp(krom)} \in \text{P}$, we get that the extension checking problem for krom default theories is in P with the help of Proposition 4.3. In order to show the NP upper bound on input $\langle W, D \rangle$, the algorithm just guesses the set of generating defaults $\mathcal{G} \subseteq D$ and then verifies if $\mathcal{W} \land \bigwedge_{\varphi \in \mathcal{G}} \varphi Y$ is an extension with respect to $\langle W, D \rangle$.

For the lower bound observe that the default theory constructed by Beyersdorff et al. [7, Lemma 5.6] consists only of literal-formulas (so being utilisable by all four classes) settling the lower bound by a reduction from 3sat. \qed
3 STRONG BACKDOORS

In this section, we lift the concept of backdoors to the world of default logic. First, we review backdoors from the propositional setting [95, 96], where a backdoor is a subset of the variables of a given formula. Formally, for a class \( \mathcal{F} \) of formulas and a formula \( \varphi \), a strong \( \mathcal{F} \)-backdoor is a set \( B \) of variables such that for all assignments \( \theta \in \mathcal{A}(B) \), we have that \( \varphi[\theta] \in \mathcal{F} \).

Backdoors in propositional satisfiability have proven themselves useful for deciding propositional satisfiability [45]: we can brute-force all assignments to the variables in the strong \( \mathcal{F} \) backdoor and solve the remaining formula when \( \mathcal{F} \) is a tractable class. These backdoors follow the binary character of truth assignments. Each variable of a given formula is considered to be either true or false. We need an analogue in default-logic for brute-forcing all assignments to the variables in the backdoor, as reasoning in default logic has a ternary character. When we consider consistent stable extensions of a given default theory then one of the following three cases is true for some formula \( \varphi \) with respect to an extension \( E \):

(i) \( \varphi \) is contained in \( E \),
(ii) the negation \( \neg \varphi \) is contained in \( E \), or
(iii) neither \( \varphi \) nor \( \neg \varphi \) is contained in \( E \) (e.g., for the theory \( \langle \{x\}, D_2 \rangle \), from Example 2.7, neither \( b \) nor \( \neg b \) is contained in any of the two stable extensions, and \( b \) is a variable).

Since we need to weave this trichotomous point of view into a backdoor definition for default logic, the original definition of backdoors cannot immediately be transferred (from the SAT setting) to the scene of default logic. The first step is a notion of extended literals and reducts. The next step can be seen as a generalisation of assignment functions to our setting.

**Definition 3.1 (Extended literals and reducts).** An extended literal is a literal or a fresh variable \( x_\ell \). For convenience, we further define \( \neg x = x \) if \( \ell = \neg x \) and \( \neg x = \neg x \) if \( \ell = x \). Given a formula \( \varphi \) and an extended literal \( \ell \), the reduct \( \rho_\ell(\varphi) \) is obtained from \( \varphi \) such that the following is true:

1. If \( \ell \) is a literal, then all clauses of \( \varphi \) that contain \( \ell \) are deleted and all literals \( \neg \ell \) are deleted from all clauses of \( \varphi \). Note that an empty clause corresponds to false.
2. If \( \ell = x_\ell \), then all occurrences of literals \( \neg x, x \) are deleted from all clauses of \( \varphi \).

Let \( \langle W, D \rangle \) be a default theory, with \( D = \{ \delta_1, \ldots, \delta_n \} \) for some \( n \in \mathbb{N} \), and \( \ell \) be an extended literal. Then, we define the reduct of \( \langle W, D \rangle \) with respect to \( \ell \) as

\[
\rho_\ell(W, D) := \left( \rho_\ell(W), \left\{ \rho_\ell(\alpha) : \rho_\ell(\beta) \mid \delta_i = \frac{\alpha : \beta}{\gamma} \in D \right\} \right),
\]

where \( y_i \) is a fresh proposition, and \( \rho_\ell(W) = \bigcup_{w \in W} \rho_\ell(w) \).

Let \( \delta = \frac{\alpha : \beta}{\gamma} \) be a default rule. Let \( C \) be a CNF-property such that \( \gamma \in C \). Then it can happen that the reduct of \( \gamma \) is no anymore in \( C \). While at first sight this might seem to be a problem, it is not. The reason is, that the conclusions are used to derive new formulas via deduction. For deduction, \( \varphi \models \gamma' \land y_i \) is equivalent to \( \varphi \models \gamma' \) and \( \varphi \models y_i \). For convenience, we will still call such default theories then \( C \)-default theories. Although, as explained before, formally for the conclusions \( \gamma' \land y_i \) this might not be the case, but \( \gamma' \) has still to be in \( C \).

Later (in the proof of Lemma 3.9), we need the \( y_i \)s for technical reasons, namely, we use them keep track of which defaults are applied when constructing a stable extension.

**Example 3.2.** Given the default theory

\[
\langle W, D \rangle = \left( \{x\}, \left\{ \frac{x : y}{\neg y \lor x} \right\} \right)
\]

we will exemplify the notion of reductions for \( x, \neg x \), and \( x \).
An application of Definition 3.1 leads to
\[ \rho_x(W, D) = \left\{ \top, \left\{ \frac{\top}{y}, \frac{\top}{y_1} \right\} \right\}, \quad \rho_{\neg x}(W, D) = \left\{ \bot, \left\{ \frac{\bot}{\neg y}, \frac{\bot}{y_1} \right\} \right\} = \rho_x(W, D). \]

In the next step, we incorporate the notion of reducts into sets of assignments. Accordingly, we introduce \textit{ternary assignment sets}. Let \( X \) be a set of variables, then we define
\[ T(X) := \{ \{a_1, \ldots, a_{|X|}\} \mid x \in X \text{ and } a_i \in \{x, \neg x, x_i\} \}. \]

Technically, \( A(X) \subseteq T(X) \) is true. That is justified by \( T(X) \) additionally containing variables \( x_i \) that will behave as “do not care” variables encompassing the trichotomous reasoning approach explained above. For \( Y \in T(X) \) the reduct \( \rho_Y(W, D) \) is the consecutive application of all \( \rho_y(y) \) for \( y \in Y \) to \( (W, D) \). Observe that the order in which we apply the reducts to \( (W, D) \) is not important.

The following proposition ensures that the addition of the \( y_i \)s from Definition 3.1 does not influence negatively our reasoning process, i.e., implication of formulas is invariant under adding conjuncts of fresh variables to the premise.

**Proposition 3.3.** Let \( \varphi, \psi \in \text{cnf} \) be two formulas and \( y \notin \text{Vars}(\varphi) \cup \text{Vars}(\psi) \). Then, \( \varphi \models \psi \) if and only if \( \varphi \land y \models \psi \).

Now we show that implication for \( \text{cnf} \)-formulas that do not contain tautological clauses is invariant under the application of “deletion reducts” \( \rho_x(\cdot) \). This will allow us, together with another property (shown in Lemma 3.5) to lift implication questions of formulas to implication questions of their reducts (Cor. 3.7). This will be a central property for applying backdoors.

**Lemma 3.4.** Let \( \varphi, \psi \in \text{cnf} \) be two formulas that do not contain tautological clauses. If \( \psi \models \varphi \), then \( \rho_x(\psi) \models \rho_x(\varphi) \) for every variable \( x \in \text{Vars}(\varphi) \cup \text{Vars}(\psi) \).

**Proof.** Assume for contradiction that \( \rho_x(\psi) \notmodels \rho_x(\varphi) \). Then, there exists an assignment \( \theta : \text{Vars}(\rho_x(\psi)) \cup \text{Vars}(\rho_x(\varphi)) \to \{0, 1\} \) such that \( \theta \models \rho_x(\psi) \) but \( \theta \notmodels \rho_x(\varphi) \). As \( \varphi \models \psi \) every arbitrary extension of \( \theta \) satisfies \( \psi \), in particular, any extension on \( \{x\} \cup \text{Vars}(\rho_x(\psi)) \cup \text{Vars}(\rho_x(\varphi)) \). Denote such an extension by \( \theta_x \). Yet, by \( \psi \models \varphi \) we get \( \theta_x \models \varphi \). As this is true for any arbitrary such \( \theta_x \) the satisfiability of \( \varphi \) is independent of setting \( x \) wherefore \( \theta_x \models \rho_x(\varphi) \) as well. (Note that here it is crucial that we require \( \varphi \) to contain no tautological clauses.) As \( x \notin \text{Vars}(\rho_x(\varphi)) \) is true, we get \( \theta \models \rho_x(\varphi) \), which is a contradiction. As a result, \( \rho_x(\psi) \models \rho_x(\varphi) \). \( \square \)

The following lemma shows that implication for \( \text{cnf} \)-formulas is invariant under the application of reducts over \( A \), i.e., the usual assignments. Note that an assignment \( \theta \) can be interpreted as a set of literals that are set to true.

**Lemma 3.5.** Let \( \varphi, \psi \) be two \( \text{cnf} \)-formulas, and \( X \subseteq \text{Vars}(\psi) \cup \text{Vars}(\varphi) \). If \( \psi \models \varphi \) is true, then \( \rho_Y(\psi) \models \rho_Y(\varphi) \) is true for every set \( Y \in A(X) \).

**Proof.** Let \( \psi, \varphi \), and \( X \) be as in the formulation of the lemma and assume that \( \psi \models \varphi \) is true. Now, fix an arbitrary \( Y \in A(X) \) and consider every assignment \( \tau_Y : \text{Vars}(\rho_Y(\psi)) \cup \text{Vars}(\rho_Y(\varphi)) \to \{0, 1\} \). Note that \( \tau_Y \) is defined on \( (\text{Vars}(\psi) \cup \text{Vars}(\varphi)) \setminus Y \). Define \( \tau \upharpoonright Y \) as the assignment \( \tau \) extended by setting \( \tau(x) := 1 \) if \( x \in X \), and \( \tau(x) := 0 \) if \( \neg x \in Y \). Consequently, \( \tau \upharpoonright Y \) completely agrees with \( \tau \) on the variables in \( Y \).

Then \( \tau \upharpoonright Y \models \neg \psi \lor \varphi \) is true by assumption as \( \psi \models \varphi \). Then, by an easy induction, we get \( \tau \upharpoonright Y \models \varphi \) if and only if \( \tau \models \rho_Y(\varphi) \), and \( \tau \upharpoonright Y \models \psi \) if and only if \( \tau \models \rho_Y(\psi) \). On that account, we get
\[
\tau \models \rho_Y(\psi) \iff \tau \upharpoonright Y \models \psi \iff \tau \upharpoonright Y \models \varphi \iff \tau \models \rho_Y(\varphi)
\]
and the lemma follows. \( \square \)
We denote by BD-Imp(cnf → F) the parameterised version of the problem Imp(cnf) where additionally a strong F-backdoor is given and the parameter is the size of the strong F-backdoor.

**Corollary 3.6.** BD-Imp(cnf → F) ∈ FPT, for F ∈ \{positive-unit, horn, krom, qhorn, rhorn, dualhorn\}.

**Proof.** Let W, φ, X be the given input instance. Then the following FPT algorithm decides the problem BD-Imp(cnf → F). For every assignment Y ∈ A(X) check if ρY(W) ⊨ ρY(φ). For the corresponding classes F these implication problems are all decidable in polynomial time (see Sect. 2.2), and positive-unit is a special case of horn. The correctness follows from Lemma 3.5. Accordingly, the corollary is proven. □

A combination of Lemma 3.4 and Lemma 3.5 now yields a generalisation for CNF-formulas that do not contain tautological clauses. Note that the crucial difference is the use of T instead of A in the claim of the result.

**Corollary 3.7.** Let ψ, φ be two cnf-formulas that do not contain tautological clauses, and X ⊆ Vars(E) ∪ Vars(φ) be a set of variables. If ψ ⊨ φ then for every set Y ∈ T(X) we have that ρY(ψ) ⊨ ρY(φ).

The following lemma states that we do not lose any stable extensions under the application of reducts. Before we can start with the lemma, we need to introduce a bit of notion. The following concept, is utilised in later identifying the set of stable extensions of a default theory. To some extend it can be seen as a syntactic sugar which otherwise could be algorithmically identified on a syntactic level. However, we will use it also on the algorithmic level for having simpler implication questions as will be pointed out in the proof of Lemma 4.5.

**Definition 3.8 (y conclusions).** For a set D = {δ₁, . . . , δₙ} of default rules and a set E of formulas, we define y-concl(D, E) := \{concl(δᵢ) | 1 ≤ i ≤ n, δᵢ ∈ D, E ⊨ yᵢ\}, that is, the set of conclusions of default rules δᵢ such that yᵢ is implied by formulas in E.

Furthermore, for a set X of variables, we will extend the notion for SE(·) as follows:

\[ SE((W, D), X) := \bigcup_{Y ∈ T(X)} \{Th(W ∪ y-concl(D, E)) | E ∈ SE(ρY(W, D))\}.\]

**Lemma 3.9.** Let (W, D) be a cnf default theory with formulas that do not contain tautological clauses, and X be a set of variables from Vars(W, D). Then, we have that SE((W, D), X) ⊆ SE((W, D) ∪ X).

**Proof.** Let (W, D) be the given default theory, X ⊆ Vars(W, D), and E ∈ SE((W, D)) be a consistent stable extension of (W, D).

We will show that E ∈ SE((W, D), X). Furthermore, let G be the set of generating defaults of E by Proposition 2.6. W.l.o.g., let G := {δ₁, . . . , δₖ} also denote the order in which these defaults are applied. That is, the ordering with respect to i refers to in which order the default rules are applied. We have E = Th(W ∪ {concl(δ) | δ ∈ G}). Accordingly, W ⊨ prereq(δ₁) is true and further fix a Y ∈ T(X) which agrees with E on the implied literals from Vars(W, D), i.e., x ∈ Y if E ⊨ x for x ∈ Vars(W, D), ¬x ∈ Y if E ⊨ ¬x, and x ∈ Y otherwise. Then, by Corollary 3.7 we know that also \( \bigwedge_{w ∈ W} ρY(w) \vdash ρY(prereq(δᵢ)) \) is true. Eventually, we get that

\[ \bigwedge_{w ∈ W} ρY(w) \land \bigwedge_{1 ≤ j ≤ i} ρY(concl(δⱼ)) \vdash ρY(prereq(δᵢ+1)) \]

is true for i < k. By definition of ρY(W, D), the reducts of the knowledge base W and the derived conclusions together trivially imply the yᵢs, i.e., we have

\[ \bigwedge_{w ∈ W} ρY(w) \land \bigwedge_{1 ≤ i ≤ k} ρY(concl(δᵢ)) \vdash \bigwedge_{1 ≤ i ≤ k} yᵢ.\]
We say that $$\exists \delta \in D \setminus G, \exists \delta' \in D \setminus G, \text{ extension } E \text{ is a consistent set, and } Y \text{ agrees with } E \text{ on the implied variables from } Vars(W, D), \text{ we get that no further default rule } \delta \text{ is implied by } \rho_Y(W) \text{ or } \rho_Y(W \cup \{ \text{concl} | \delta \in D \setminus G \}).$$

Furthermore, we have that no justification is violated as $$E \models \neg \beta \text{ for some } \beta \in \bigcup_{\delta \in \delta} \text{just}(\delta) \text{ would imply that } \rho_Y(E) \models \neg \rho_Y(\beta) \text{ is true by Corollary 3.7. Eventually } E' = \text{Th}(\rho_Y(W) \cup \{ \rho_Y(\text{concl}) | \delta \in G \}) \text{ is a stable extension with respect to } \rho_Y(W, D). \text{ But, the set of conclusions of } G \text{ coincides with } y\text{-concl}(D, E') \text{ wherefore}

$$E = \text{Th}(W \cup \{ \text{concl} | \delta \in G \})$$

$$= \text{Th}(W \cup y\text{-concl}(D, E')) \in \text{SE}(\langle\langle W, D, X \rangle\rangle)$$

is true. \hfill \Box

We have seen that it is important to disallow tautological clauses. However, the detection of this kind of clauses is possible in polynomial time. As a consequence, we assume in the following that a given theory contains no tautological clauses. This is a very weak restriction as (i) $$\varphi \wedge C \equiv \varphi$$ for any tautological clause $$C$$ and (ii) $$C \equiv \top$$ for any tautological clause $$C$$.

The following example illustrates how reducts maintain existence of stable extensions.

**Example 3.10.** The default theory $$\langle W, D \rangle = \{ \{x\}, \{ \frac{T}{T}, \frac{x}{x} \} \}$$ has the extension $$E = \text{Th}(x, \neg z \vee x)$$ and yields the following cases for the set $$B = \{x\}$$: $$\rho_\top(W, D) = \{ \top \}, \{ \frac{T}{T}, \frac{x}{x} \} \}$$, yields $$\text{SE}(\rho_\top(W, D)) = \{ \text{Th}(\neg z \wedge y_1) \}$$, and, both, $$\rho_{\neg z}(W, D)$$ and $$\rho_{z}(W, D)$$ yield an empty set of stable extensions. Then, with $$y\text{-concl}(D, \text{Th}(\neg z \wedge y_1)) = \{ \neg z \vee x \}$$ we get $$\text{Th}(\langle\{\neg z \vee x \} \cup \{x\} \rangle)$$, which is equivalent to the extension $$E$$ of $$\langle W, D \rangle$$.

As explained before, reducts of conclusions in default rules might formally not be in the class of formulas we are interested in. For that reason we introduce the following definition.

**Definition 3.11 (Formula class membership for reducts of default theories).** Let $$\mathcal{F}$$ be a class of formulas. Let $$\langle W, D \rangle$$ be a default theory. Let $$\ell$$ be an extended literal and $$\rho_\ell(W, D)$$ be the reduct of $$\langle W, D \rangle$$ with respect to $$\ell$$. We say that $$\rho_\ell(W, D) = \langle W', D' \rangle$$ is an $$\mathcal{F}$$-default theory if

- for all $$\varphi \in W'$$ we have that $$\varphi \in \mathcal{F}$$,
- for all $$\frac{\alpha \beta}{\gamma} \in D'$$ we have that $$\alpha, \beta \in \mathcal{F}$$ and
- for all $$\frac{\alpha \beta}{\gamma} \in D'$$ such that $$\gamma = \gamma' \wedge y_i$$ for some $$i \in \mathbb{N}$$, we have that $$\gamma' \in \mathcal{F}$$.

Now, we are in the position to present a definition of strong backdoors for default logic.

**Definition 3.12 (Strong Backdoors for Default Logic).** Given a cnf default theory $$\langle W, D \rangle$$, a set $$B \subseteq Vars(W, D)$$ of variables, and a class $$\mathcal{F}$$ of formulas. We say that $$B$$ is a strong $$\mathcal{F}$$-backdoor if for each $$Y \in \mathbb{T}(B)$$ the reduct $$\rho_Y(W, D)$$ is an $$\mathcal{F}$$-default theory.

4 BACKDOOR EVALUATION

In this section, we investigate the evaluation of strong backdoors for the extension existence problem in default logic with respect to different classes of CNF-formulas. Formally, the problem of strong backdoor evaluation for extension existence is defined as follows.

**Problem:** EvalExt($$\mathcal{F} \rightarrow \mathcal{F}'$$), also called Backdoor Evaluation.

**Input:** An $$\mathcal{F}$$-default theory $$\langle W, D \rangle$$, and a strong $$\mathcal{F}'$$-backdoor $$B \subseteq Vars(W, D)$$.

**Parameter:** The size of the backdoor $$B$$.

**Question:** Does $$\langle W, D \rangle$$ have a stable extension?
Example 4.1 (Continuation of Example 2.7). Notice that \( B = \{u\} \) is a strong horn-backdoor for the default theory \( \{x \lor u, \{\frac{x}{\neg y}, \frac{x \lor y}{z}\}\} \) as follows:

\[
\rho_u(W, D) = \{x\}, \left\{ \begin{array}{l}
x : z \\ \neg y \land y_1 \\
\neg z \land y_2 
\end{array} \right\},
\]

\[
\rho_{\neg u}(W, D) = \{x\}, \left\{ \begin{array}{l}
x : z \\ \neg y \land y_1 \\
\neg z \land y_2 
\end{array} \right\},
\]

\[
\rho_{u}(W, D) = \{x\}, \left\{ \begin{array}{l}
x : z \\ \neg y \land y_1 \\
\neg z \land y_2 
\end{array} \right\},
\]

while before it was not Horn because of the knowledge base formula \( x \lor u \). All three reduct default theories have the stable extensions \( \text{Th}(\{x, \neg y \land y_1\}) \) and \( \text{Th}(\{x, \neg z \land y_2\}) \). Notice that either extension nicely shows via the respective \( \frac{\beta}{\gamma} \) which default rule was applied.

First, we study the complexity of the “extension checking problem”, which is a main task we need to accomplish when using backdoors as our approach following Lemma 3.9 yields only “stable extension candidates”. The problem is defined as follows:

**Problem:** EC, also called Extension Checking.

**Input:** A default theory \( \langle W, D \rangle \) and a finite set \( \Phi \) of formulas.

**Question:** Does \( \text{Th}(\Phi) \in \text{SE}(\langle W, D \rangle) \) hold?

Example 4.2. Consider the default theory \( \{x, \{\frac{x}{y}\}\} \) and the set of formulas \( \Phi = \{x, y\} \). Then, \( \text{Th}(\Phi) \) is a stable extension of \( \{x, \{\frac{x}{y}\}\} \), as the only existing default rule is applicable and the corresponding conclusion is \( y \).

Rosati [84] classified the extension checking problem as complete for the complexity class \( \Theta^2_P = \Delta^P_2[\log] \), which allows only logarithmically many oracle questions to an NP oracle. For further information on the complexity class \( \Theta^2_P \), we refer the reader to the survey article of Eiter and Gottlob [23]. Later, we will see that a simpler version suffices for our complexity analysis. As a consequence, we state in Algorithm 1 an adaption of Rosati’s algorithm [84, Figure 1] to our notation showing containment (only) in \( \Delta^P_2 \). From an algorithmic point of view it is important to note that the input to the algorithm is, of course, a finite set of formulas. Comparing Definition 2.5 and Proposition 2.6, it is clear that the input set \( E \) to the algorithm must be finite and the extension itself \( \text{Th}(E) \) contains the (infinitely large sets of) tautologies as well as the implied formulas of \( E \). However, the sets \( E \) which serve as “extension candidates” are finitely represented by the help of generating defaults.

Observe that a justification-free default theory has exactly one extension. That is why the justifications \( \beta \) in the default rules processed in Line 2–3 of Algorithm 1 are removed. Intuitively, the complexity of the extension checking problem EC heavily relies on the difficulty of the implication problem. If the complexity of the implication question drops to \( P \), then the complexity of EC also drops to \( P \) (as used in the proof of Corollary 2.13).

**Proposition 4.3 ([84, Figure 1, Theorem 4]).** EC \( \in \Delta^P_2 \).

In a way, extension checking can be compared to model checking in logic. In default logic the complexity of the extension existence problem Ext is twofold: using the approach of Proposition 2.6

(i) one has to non-deterministically guess the set (and ordering) of the generating defaults, and

(ii) one has to verify whether the generating defaults lead to an extension.
Algorithm 1: Extension checking algorithm [84, Theorem 4]

Input: Set $E$ of formulas and a default theory $(W, D)$
Output: True if and only if $E$ is a stable extension of $(W, D)$

1. $D' := \emptyset$
2. foreach $\frac{\beta}{\gamma} \in D$ do // (1) Classify unviolated justifications.
   3. if $E \nvdash \neg \beta$ then $D' := D' \cup \{ \frac{\beta}{\gamma} \}$
   // (2) Compute extension candidate of justification-free theory.
4. $E' := W$
5. while $E'$ did change in the last iteration do
6.   foreach $\frac{\alpha}{\gamma} \in D'$ do
7.     if $E' \models \alpha$ then $E' := E' \cup \{ \gamma \}$
   // (3) Does the candidate match the extension?
8. if $E \models E'$ and $E' \models E$ then return true else return false

For (ii), one needs to answer quadratically many implication questions. As a result, the problem is in $NP^{NP}$. Accordingly, a straightforward approach for EC omits the non-determinism in (i), by going through the default set $D$ and sorting out those default rules whose justifications are violated, and achieves the result in $P^{NP}$.

The main theorem of this section is proven in the following lemmas and corollaries.

**Theorem 4.4.**

1. $EvalExt(cnf \rightarrow monotone)$ is para-$\Delta_2^P$-complete. (Lemma 4.7)
2. $EvalExt(cnf \rightarrow horn)$ is para-NP-complete. (Lemma 4.5)
3. $EvalExt(cnf \rightarrow qhorn)$ is para-NP-complete. (Corollary 4.6)
4. $EvalExt(cnf \rightarrow rhorn)$ is para-NP-complete. (Corollary 4.6)
5. $EvalExt(cnf \rightarrow dualhorn)$ is para-NP-complete. (Corollary 4.6)
6. $EvalExt(cnf \rightarrow krom)$ is para-NP-complete. (Corollary 4.6)
7. $EvalExt(cnf \rightarrow affine)$ is para-NP-complete. (Corollary 4.6)
8. $EvalExt(cnf \rightarrow positive-unit) \in FPT$. (Corollary 4.8)
9. $EvalExt(cnf \rightarrow valid) \in FPT$ for $c \in \{0, 1\}$. (Corollary 4.9)

**Lemma 4.5.** $EvalExt(cnf \rightarrow horn)$ is para-NP-complete.

**Proof.** We start with the upper bound. Let $(W, D)$ be a given $cnf$ default theory and $B \subseteq Vars(W, D)$ be the given backdoor. In order to evaluate the backdoor, we have to consider the $|T(X)| = 3^{|B|}$ many different reducts to $horn$ default theories. For each of them we have to non-deterministically guess a possible “extension candidate” $E$. Then, we use Algorithm 1 to verify whether $E$ is indeed a valid extension. Note that, $Imp(horn) \in P$ by Cor. 2.11. Consequently, stable extension checking is in $P$ for $horn$ formulas. Then, after finding an extension $E$ with respect to the reduct default theory $p_Y(W, D)$, we need to compute the corresponding extension $E'$ with respect to the original default theory. Here we just need to verify simple implication questions of the form $E \models y_i$ for $1 \leq i \leq |D|$. Next, we need to verify whether $E'$ is a valid extension for $(W, D)$ using Algorithm 1. Note that Corollary 3.6 shows that the implication problem of propositional $horn$ formulas parameterised by the size of the backdoor is in $FPT$, accordingly, we can compute the implication questions without the need for oracle calls already within the computation. As the length of the used formulas is bounded by the input size and the relevant parameter is the same as for the input this step runs in fpt-time. Together this yields a para-NP algorithm. Algorithm 2 depicts a generic algorithm in pseudocode.
Algorithm 2: Generic algorithm for $\text{EvalExt}(F \to F')$

\begin{algorithmic}[1]
\State \textbf{Input:} $F$-default theory $(W, D)$, backdoor $B \subseteq \text{Vars}(W, D)$
\For {$Y \in T(B)$}
\State guess an extension candidate $E$ for $F'$ default theory $\rho_Y(W, D)$
\If {$E$ is an extension for $\rho_Y(W, D)$}
\State $E' := \bigcup_{y \in W \wedge (D, E) \in \text{cond}(D, E)} y$ \Comment{always in $P$ by construction}
\EndIf
\EndFor
\If {$E'$ is an extension for $(W, D)$} \textbf{return} true \EndIf
\textbf{return} false
\end{algorithmic}

For the para-NP lower bound, $\text{Ext}$ restricted to literal-formulas only is NP-complete [7, Lemma 5.6] which implies the lower bound for backdoor-size 0 already. \hfill $\square$

**Corollary 4.6.** $\text{EvalExt}(\text{cnf} \to F)$ is para-NP-complete for $F \in \{\text{krom, rhorn, dualhorn, qhorn, affine}\}$.

**Proof.** The implication problem of all these formulas (except for affine) is in $P$ due to Sect. 2.2. Regarding sets of affine formulas (we can split the $\oplus$-CNF-formulas at the conjunctions) the implication problem is in $\oplus L \subseteq NC^2 \subseteq P$ due to Beyersdorff et al. [6, Thm. 4.1 (2)]. Eventually, under a similar argumentation as in the proof of Lemma 4.5, we can construct a para-NP algorithm.

Regarding the para-NP lower bound, $\text{Ext}$ restricted to literal-formulas only is NP-complete [7, Lemma 5.6] and literals can be expressed by all these classes of formulas. This implies the lower bound for backdoor-size 0 already. \hfill $\square$

**Lemma 4.7.** $\text{EvalExt}(\text{cnf} \to \text{monotone})$ is para-$\Delta^P_2$-complete.

**Proof.** Start with the upper bound. For a monotone formula $\varphi$, its negation is not any longer monotone unless $\varphi \in \{\top, \bot\}$. This observation is important for such $\varphi$ occurring as justifications. If $\varphi \notin \{\top, \bot\}$, the justification can be deleted because its negation cannot be derived, so the default rule applies whenever its prerequisite is met. If $\varphi \in \{\top, \bot\}$, either it is only applicable in an inconsistent case or always. Accordingly, we can distinguish between these cases in polynomial time. Note that from the previous argument we can conclude that there is a unique stable extension, if there is one at all. Furthermore, the construction of the set of generating defaults and also the extension itself is achievable in para-$\Delta^P_2$ as follows: we have to do quadratically many implication questions, and the implication problem for monotone formulas has the same upper bound as the unrestricted one, that is, coNP. Step (5) of Algorithm 2 just checks for implications, which is solved via the para-$\Delta^P_2$ algorithm.

With respect to the lower bound, note that Beyersdorff et al. [7, Lemma 5.5] have shown $\Delta^P_2$-completeness for $\text{Ext}$ restricted to monotone-formulas, which are formulas containing only $\lor, \land, 0$ as connectors. However, in their proof [7, Lemma 5.5], the authors construct a default theory consisting of a knowledge base with pure CNF-formulas (they reduce from sequentially nested SAT, SNSAT for short, which can be restricted to CNF-formulas) and default rules of the following form

\[
D := \left\{ \begin{array}{l}
\forall_{i=1}^m \left( z_{ij} \land z'_{ij} \right) \lor \forall_{j=1}^{i-1} \left( x_j \land x'_j \right) : \top \\
\forall_{j=1}^{n-1} \left( x_j \land x'_j \right) : \bot \\
\end{array} \right\} 1 \leq i < n \right\} \cup \right.
\]
Clearly, the formulas in the prerequisites of the default rules are not in CNF but in DNF. In our reduction, we need CNF-
default rules though. A naive translation of the DNF formulas from the default rule prerequisites in \( D \)
to an equivalent CNF-formula is not possible, as it would go beyond the runtime requirement. Yet, it is possible to "split" each prerequisite into new default rules having only CNFs as component formulas by declaring auxiliary
variables. To achieve this translation, the default rules from above are replaced in two steps. First, we substitute each rule from above for each \( 1 \leq i \leq n \) by

\[
D' := \left\{ \frac{d_i \colon \top}{x_i'} \mid 1 < n \right\} \cup \left\{ \frac{d_n \colon \bot}{\perp} \right\}.
\]

Then, the still not considered prerequisites from the rules in \( D \) are mimicked by sets of default rules from the set \( D'' \) giving the possibility that one of the DNF terms is implied and by this triggering the rules from \( D' \):

\[
D'' := \left\{ \frac{z_{ij} \land z'_{ij} : \top}{d_i}, \frac{x_j \land x'_j : \top}{d_i} \bigg| 1 \leq j \leq m_i, 1 \leq i \leq n \right\}.
\]

So, for each previous default rule we split the DNF of the prerequisite into a bunch of default rules that trigger the previous DNF-default (now mimicked via a fresh proposition \( d_i \)) whenever one of the DNF-terms is implied. By construction, we then observe that

\[
\left\{ \bigvee_{j=1}^{m_i} (z_{ij} \land z'_{ij}) \vee \bigvee_{j=1}^{m_i} (x_j \land x'_j) \colon \top \bigg| 1 \leq i \leq n \right\}
\]

is applicable if and only if the prerequisite \( \bigvee_{j=1}^{m_i} (z_{ij} \land z'_{ij}) \vee \bigvee_{j=1}^{m_i} (x_j \land x'_j) \) is implied if and only if one of the DNF-terms \( (z_{ij} \land z'_{ij}) \) or \( (x_j \land x'_j) \) is implied if and only if one element from \( D'' \) is applicable if and only if the corresponding rule in \( D' \) is applicable. This establishes the lower bound. \( \square \)

**Corollary 4.8.** EvalExt\((\text{cnf} \rightarrow \text{positive-unit})\) \( \in \) \( \text{FPT} \).

**Proof.** The implication problem for positive-unit formulas is in \( \text{AC}^0 \) by Beyersdorff et al. who showed this result for formulas using only conjunctions \( [6, \text{Theorem 4.1(4)}] \). That being so, Algorithm 1 runs in polynomial time. Accordingly, we achieve the FPT upper bound by a similar argumentation as in the proof of Lemma 4.5. \( \square \)

**Corollary 4.9.** EvalExt\((\text{cnf} \rightarrow \text{cvalid})\) \( \in \) \( \text{FPT} \) for \( c \in \{0, 1\} \).

**Proof.** Let \( \varphi \) be a cvalid formula for a fixed \( c \in \{0, 1\} \). Then the formula \( \neg \varphi \) is no longer cvalid. As a result, justifications in default rules are irrelevant, since they cannot be inferred. Accordingly, there is always a unique extension. \( \square \)

### 5 Backdoor Detection

In this section, we study the problem of finding backdoors, formalised in terms of the following parameterised problem. In the previous section, we considered all classes of formulas as defined by Schaefer for evaluation. Here, we restrict ourselves to CNF as input, meaning, that we disregard affine formulas.

**Problem:** BdDetect\((\text{cnf} \rightarrow F)\) also called Backdoor Detection

**Input:** A CNF default theory \( T \) and an integer \( k \).

**Parameter:** The integer \( k \).

**Question:** Does \( T \) have a strong \( F \)-backdoor of size at most \( k \)?

The following statement allows us to lift interesting target classes from propositional satisiability to default logic and thereby determine backdoors also for default logic in fpt-time. Therefore, we consider deletion backdoors.
of formulas. Here, \( F - X \) denotes the formula obtained from \( F \) by removing all literals \( x \), \( \neg x \) for \( x \in X \) from the clauses of \( F \). Then, a set \( X \) of atoms is a deletion \( \mathcal{F} \)-backdoor of \( F \) if \( F - X \in \mathcal{F} \).

Before we proceed, we need to introduce a requirement for target classes that can be directly used. In fact, the requirement is quite weak as all useful target classes for satisfiability on \( \text{cnf} \)-formulas fulfil it. We say that a class \( \mathcal{F} \subseteq \text{cnf} \) of formulas is \( \text{clause-restricting} \) if it restricts each clause to a certain form, but, does not impose any restrictions between two clauses. Note that all target classes considered in this work are clause-restricting, i.e., \text{horn}, \text{dualhorn}, \text{qhorn}, \text{rhorn}, \text{krom}, \text{monotone}, and \text{positive-unit}.

Intuitively, the following lemma states that we can use deletion backdoors of a specific formula instead of strong backdoors for a default theory, and vice versa.

**Lemma 5.1.** Let \( \langle W, D \rangle \) be a \( \text{cnf} \)-default theory, \( B \subseteq \text{Vars}(W, D) \) be a set of variables, and \( \mathcal{F} \subseteq \text{cnf} \) be a clause-induced and clause-restricting class of formulas. The set \( B \) is a \( \mathcal{F} \)-backdoor of \( \langle W, D \rangle \) if and only if \( B \) is a deletion \( \mathcal{F} \)-backdoor of the formula

\[
F = \bigwedge_{\ell \in W} \ell \land \bigwedge_{\delta \in D} \left( \bigwedge_{\alpha \in \text{prereq}(\delta)} \alpha \land \bigwedge_{\beta \in \text{just}(\delta)} \beta \land \bigwedge_{\gamma \in \text{concl}(\delta)} (\gamma \land y_{\gamma}) \right).
\]

**Proof.** Given a \( \text{cnf} \)-default theory \( \langle W, D \rangle \), a set \( B \subseteq \text{Vars}(W, D) \) of variables, and a class \( \mathcal{F} \subseteq \text{cnf} \) of clause-induced formulas. Furthermore, let \( X \) consist of the variables occurring in \( \langle W, D \rangle \).

\( \Rightarrow \): Assume that the set \( B \) is a \( \mathcal{F} \)-backdoor of the default theory \( \langle W, D \rangle \), i.e., for each \( Y \in \mathcal{T}(B) \) the reduct \( \rho_Y(W, D) \) is an \( \mathcal{F} \)-default theory. Since \( \mathcal{A}(X) \subseteq \mathcal{T}(X) \) holds, we have in particular, for every \( Y \in \mathcal{A}(B) \), that \( \rho_Y(w) \in \mathcal{F} \) for each \( w \in W \) and \( \rho_Y(\alpha) \in \mathcal{F} \), \( \rho_Y(\beta) \in \mathcal{F} \), and \( \rho_Y(\gamma \land y_{\gamma}) \in \mathcal{F} \) for each \( \delta \in D \) with \( \delta = \frac{\alpha - \beta}{\gamma} \) and the fresh propositions \( y_{\gamma} \). Since the class \( \mathcal{F} \) is clause-restricting, each class requires only restrictions on the clauses and hence the set \( B \) is a \( \mathcal{F} \)-backdoor of a conjunction of these formulas. Since the class \( \mathcal{F} \) is clause-induced, the set \( B \) is also a deletion \( \mathcal{F} \)-backdoor of \( F \) \cite{55}.

\( \Leftarrow \): Conversely, assume that the set \( B \) is a deletion \( \mathcal{F} \)-backdoor of the formula \( F \) as defined above. Hence, we have that \( \bigwedge_{\ell \in W} \ell \land \bigwedge_{\delta \in D} \left( \bigwedge_{\alpha \in \text{prereq}(\delta)} \alpha \land \bigwedge_{\beta \in \text{just}(\delta)} \beta \land \bigwedge_{\gamma \in \text{concl}(\delta)} (\gamma \land y_{\gamma}) \right) - B \in \mathcal{F} \). Since \( \mathcal{F} \) is clause-restricting, for every \( w \in W \) it is true that \( w - B \in \mathcal{F} \) and for every \( \delta \in D \) it holds that \( \text{prereq}(\delta) - B \in \mathcal{F} \), \( \text{just}(\delta) - B \in \mathcal{F} \), and \( \text{concl}(\delta) - y_{\gamma} \in \mathcal{F} \). Next, we show that \( B \) is also a strong \( \mathcal{F} \)-backdoor of \( \langle W, D \rangle \) according to Definitions 3.1 and 3.12, i.e., for every assignment \( Y \in \mathcal{T}(B) \) it holds that \( \rho_Y(W, D) \) is an \( \mathcal{F} \)-default theory. By assumption, we know that for every \( \ell \in W \), \( \kappa - B \in \mathcal{F} \), and \( \text{prereq}(\delta) - B \in \mathcal{F} \), \( \beta(\delta) - B \in \mathcal{F} \), and \( \gamma(\delta) - B \in \mathcal{F} \) for every \( \delta \in D \). Assume that \( \text{cnf} \)-formulas are represented as sets of clauses in the context of the \( \subseteq \)-relation. We show that \( \rho_B(G) \subseteq G - B \) for some formula \( G \subseteq F \), which is sufficient to combine with the assumption that \( B \) is a deletion \( \mathcal{F} \)-backdoor of \( G \), and \( \mathcal{F} \) is clause-induced, meaning, whenever \( F \in \mathcal{F} \), all subformulas \( F' \subseteq F \) belong to \( \mathcal{F} \). Let \( c \in G \) be an arbitrary clause of the form \( c = \{ \ell_1, \ldots, \ell_k \} \) of \( G \). We distinguish three cases for \( \ell_k \in Y \):

(i) \( \ell_k \in \mathcal{A}(B) \) and \( \ell \) is a positive literal,

(ii) \( \ell_k \in \mathcal{T}(B) \) and \( \ell \) is a negative literal, and

(iii) \( \ell_k \in \mathcal{A}(B) \) and \( \ell \) is a negative literal.

If Case (i) holds and \( \ell \) occurs positively in \( Y \), then \( c \) is removed when constructing \( \rho_B(G) \) whereas \( \ell \) is just removed from \( c \) when constructing \( G - B \). If Case (ii) is true and \( \ell \) occurs negatively, \( \ell \) is removed from both constructions. If Case (iii) holds, \( \ell \) is removed in both constructions from \( c \). From applying the three cases to all literals and clauses, we obtain \( \rho_B(G) \subseteq G - B \). By the above statement, we conclude that \( \rho_B(\kappa) \subseteq \kappa - B \), and \( \text{prereq}(\delta) - B \subseteq \text{prereq}(\delta) \), \( \beta(\delta) - B \in \mathcal{F} \), and \( \gamma(\delta) - B \in \mathcal{F} \) for every \( \delta \in D \). This establishes the if-direction.

Lemma 5.1 establishes the connection between deletion backdoors for SAT and strong backdoors for default logic. In a way, it also tells us that notions differ in default logic and explains why we do not provide the concept.
of deletion backdoors for default logic. Following Definition 3.1 of extended literals and reducts, the intuition is easy. Case (ii) already includes a "deletion case" for the literals that have to be evaluated as undecided. In the following statement and proof, we lift results from the SAT case to default logic applying the lemma above.

Corollary 5.2. \( \text{BdDetect}(\text{cnf} \rightarrow C) \in \text{FPT}, \) for \( C \in \{ \text{horn}, \text{positive-unit}, \text{krom}, \text{monotone}, \text{qhorn}, \text{dualhorn}, \text{positive-unit}, 0\text{valid}, 1\text{valid} \} \) and \( \text{BdDetect}(\text{cnf} \rightarrow C) \) is fpt-approximable if \( C = \text{rhorn} \).

Proof. Using Lemma 5.1 above, we obtain the result directly from known results and constructions in the propositional setting. Case \( C = \text{monotone} \): A cnf-formula \( \varphi \) is monotone if every literal appears only positively in any clause \( C \in \varphi \) where \( \varphi \in C \). We can trivially construct a smallest strong monotone-backdoor by taking all negative literals of clauses in formulas of \( C \) in linear time. Consequently, the claim is true. We apply results from the propositional setting as follows \( C \in \{ \text{horn}, \text{positive-unit}, \text{krom} \} \) [85], \( C = \text{rhorn} \) [45], \( C = \{ \text{dualhorn}, 0\text{valid}, 1\text{valid} \} \) [76], and for \( C = \text{qhorn} \) [43].

If the target class \( F' \) is clause-induced, we can use a decision algorithm for \( \text{BdDetect}(F \rightarrow F') \) to find the backdoor using self-reduction [19, 87].

Lemma 5.3. Let \( F \) be a clause-induced class of cnf-formulas. If the problem \( \text{BdDetect}(\text{cnf} \rightarrow F) \) is fixed-parameter tractable, then also computing a strong \( F \)-backdoor of size at most \( k \) of a given default theory \( T \) is fixed-parameter tractable (for parameter \( k \)).

Proof. Let \( T = (W, D) \) be a default theory. We proceed by induction on \( k \). If \( k = 0 \) the statement is clearly true. Let \( k > 0 \). Given \( (T, k) \) we check for all \( v \in \text{Vars}(W) \cup \text{Vars}(D) \) whether \( \rho_Y(W, D) \), \( \rho_{Y'}(W, D) \), and \( \rho_{Y''}(W, D) \) have a strong \( F \)-backdoor of size at most \( k - 1 \) where \( Y = \{ v \} \), \( Y' = \{ \neg v \} \), and \( Y'' = \{ v \} \). If the answer is No for all \( v \), then \( T \) has no strong \( F \)-backdoor of size \( k \). If the answer is Yes for \( v \), then by induction hypothesis we can compute a strong \( F \)-backdoor \( B \) of size at most \( k - 1 \) of \( \rho_Y(W, D) \), \( \rho_{Y'}(W, D) \), and \( \rho_{Y''}(W, D) \) and \( B \cup \{ v \} \) is a strong \( F \)-backdoor of \( T \).

Now, we can use Corollary 5.2 to strengthen the results of Theorem 4.5 and Corollary 4.6 and Lemma 4.7 by dropping the assumption that the backdoor is given.

Theorem 5.4. The problem \( \text{EvalExt}(\text{cnf} \rightarrow C) \) parameterised by the size of a smallest strong \( C \)-backdoor of the given theory is

1. para-\( \Delta^P_2 \)-complete if \( C = \text{monotone} \),
2. para-NP-complete if \( C \in \{ \text{horn}, \text{krom}, \text{qhorn}, \text{rhorn}, \text{dualhorn} \} \), and
3. in FPT if \( C = \{ \text{positive-unit}, 0\text{valid}, 1\text{valid} \} \).

6 CONCLUSION
We layed out the foundations of backdoors for propositional default logic. Then, we used this concept to study the parameterised complexity of backdoor set evaluation and detection for all central cnf-based classes of propositional formulas. We established that backdoor detection for the classes cnf, horn, qhorn, dualhorn, krom, monotone, 0valid, 1valid and positive-unit are fixed-parameter tractable whereas for rhorn we achieved only fpt-approximation status. Regarding the evaluation problem the classification is more complex. For monotone default theories, we achieved para-\( \Delta^P_2 \)-completeness (improving from N PNP- completeness in the classical setting). For horn, krom, qhorn, rhorn, and dualhorn, we showed that the problem is para-NP-completeness. Finally, for positive-unit, 0valid, and 1valid, we showed that the problem is fixed-parameter tractable. Consequently, this problem can be solved by an fpt-algorithm that can query a SAT-solver multiple times [17]. Notice that this strengthening of Theorem 5.4 does not apply to affine default theories due to its lack of clause-inducedness (here, it would be

ACM Trans. Comput. Logic
interesting to find a corresponding result). As a result, if one considers solely the Schaefer-classes, then this would yield a dichotomy theorem for the evaluation problem distinguishing between \( \text{FPT} \) and para-NP cases.

Furthermore, we would like to mention that fruitful combinations of backdoors and the well-established parameter treewidth have been pointed out [34]. Also recent reduction techniques, so-called DP reductions, and structural investigations could be of interest [25, 26, 51].

Immediate future work includes other nonmonotonic logics such as autoepistemic logic [75], abduction [79, 80] and circumscription [65]. Particularly, there exist the concept of direct translations [24, 47, 54, 57, 62, 67, 68, 94] and it would be interesting to investigate whether our results and concepts can be transferred to these logics. Furthermore, as there exist recent parameterised complexity investigations of nonmonotonic formalisms [61, 73], the combination of these results with backdoors might yield new insights.

Moreover, a systematic study of the (parameterised) enumeration complexity [14, 15, 69, 70] in this setting is planned for future work. In addition, quantitative aspects for understanding solutions [28] or more precise reasoning [31, 32], might be interesting topics for future research. Finally, a direct application of quantified Boolean formulas in the context of propositional default logic, for instance, via the work of Egly et al. [22] or exploiting backdoors similar to results by Fichte and Szeider [37], might yield new insights.

ACKNOWLEDGMENTS

The authors thank Jonni Virtema for pointing out ideas of proving Lemma 2.10 and Sebastian Ordyniak for discussions on Lemma 3.4. The authors also thank the anonymous reviewers for their valuable comments and suggestions. The last two authors appreciate funding by the German Research Foundation (DFG) under grant ME 4279/1-1.

REFERENCES

[1] Mario Alviano. 2017. Model enumeration in propositional circumscription via unsatisfiable core analysis. *Theory Pract. Log. Program.* 17, 5-6 (2017), 708–725.

[2] Mario Alviano. 2018. Query Answering in Propositional Circumscription. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI’18)*, Jérôme Lang (Ed.). ijcai.org, 1669–1675.

[3] Robert Baumgartner and Georg Gottlob. 2002. Propositional default logics made easier: computational complexity of model checking. *Theor. Comput. Sci.* 289, 1 (2002), 591–627.

[4] Dritan Berzati, Bernhard Anrig, and Jürg Kohlas. 2002. Embedding default logic in propositional argumentation systems. In *Proceedings of the 9th International Workshop on Non-Monotonic Reasoning (NMR’02)*, Salem Benferhat and Enrico Giunchiglia (Eds.). 302–310.

[5] Olaf Beyersdorf, Arne Meier, Sebastian Müller, Michael Thomas, and Heribert Vollmer. 2011. Proof complexity of propositional default logic. *Arch. Math. Log.* 50, 7-8 (2011), 727–742.

[6] Olaf Beyersdorf, Arne Meier, Michael Thomas, and Heribert Vollmer. 2009. The Complexity of Propositional Implication. *Inform. Process. Lett.* 109, 18 (2009), 1071–1077.

[7] Olaf Beyersdorf, Arne Meier, Michael Thomas, and Heribert Vollmer. 2012. The complexity of reasoning for fragments of default logic. *Journal of Logic and Computation* 22, 3 (2012), 587–604. https://doi.org/10.1093/logcom/exq061

[8] Endre Boros, Yves Crama, and Peter L. Hammer. 1990. Polynomial-Time Inference of All Valid Implications for Horn and Related Formulae. *Ann. Math. Artif. Intell.* 1 (1990), 21–32.

[9] Endre Boros, Peter L. Hammer, and Xiaorong Sun. 1994. Recognition of q-Horn Formulae in Linear Time. *Discr. Appl. Math.* 55, 1 (1994), 1–13.

[10] V. Chandru, Collette R. Coullard, Peter L. Hammer, M. Montanuz, and Xiaorong Sun. 1990. On Renamable Horn and Generalized Horn Functions. *Ann. Math. Artif. Intell.* 1 (1990), 33–47.

[11] Jianer Chen, Benny Chor, Michael R. Fellows, XiuZhen Huang, David W. Juedes, Iyad A. Kanji, and Ge Xia. 2005. Tight Lower Bounds for Certain Parameterized NP-Hard Problems. *Information and Computation* 201, 2 (2005), 216–231.

[12] Bruno Courcelle. 1990. Graph Rewriting: An Algebraic and Logic Approach. In *Handbook of theoretical computer science*, Jan van Leeuwen (Ed.), Vol. Formal Models and Semantics. Elsevier Science Publishers, North-Holland, Cambridge, MA, USA, 193–242.

[13] Bruno Courcelle and Joost Engelfriet. 2012. *Graph structure and monadic second-order logic, a language theoretic approach*. Cambridge University Press, Cambridge.
[34] Johannes K. Fichte, Markus Hecher, and Ira Schindler. 2022. Default logic and bounded treewidth. Information and Computation 283 (2022), 104675. https://doi.org/10.1016/J.ICA.2020.104675

[35] Johannes K. Fichte, Martin Kronegger, and Stefan Woltran. 2019. A multiparametric view on answer set programming. Ann. Math. Artif. Intell. 86, 1-3 (2019), 121–147. https://doi.org/10.1007/s10472-019-09633-x

[36] Johannes K. Fichte, Arne Meier, and Ira Schindler. 2016. Strong Backdoors for Default Logic. In Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing (SAT’16) (Lecture Notes in Computer Science, Vol. 9710), Nadia Creignou and Daniel Le Berre (Eds.). Springer Verlag, Bordeaux, France, 45–59. https://doi.org/10.1007/978-3-319-40970-2_4

[37] Johannes K. Fichte and Stefan Szeider. 2015. Backdoors to Normality for Disjunctive Logic Programs. ACM Trans. Comput. Log. 17, 1 (2015), 7.

[38] Johannes K. Fichte and Stefan Szeider. 2015. Backdoors to Tractable Answer-Set Programming. Artificial Intelligence 220 (2015), 64–103. https://doi.org/10.1016/j.artint.2014.12.001

[39] Jörg Flum and Martin Grohe. 2003. Describing parameterized complexity classes. and Approximability. In 187, 2 (2003), 291–319.

[40] Jörg Flum and Martin Grohe. 2006. Parameterized Complexity Theory. Theoretical Computer Science, Vol. XIV. Springer Verlag, Berlin. 495 pages. https://doi.org/10.1016/3-540-29953-X

[41] Robert Ganian, M. S. Ramanujan, and Stefan Szeider. 2017. Backdoor Treewidth for SAT. In Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing (SAT’17) (Lecture Notes in Computer Science, Vol. 10491), Serge Gaspers and Toby Walsh (Eds.). Springer Verlag, Melbourne, VIC, Australia, 20–37. https://doi.org/10.1007/978-3-319-66263-3_2

[42] Serge Gaspers, Neeldhara Misra, Sebastian Ordyniak, Stefan Szeider, and Stanislav Zivny. 2017. Backdoors into heterogeneous classes of SAT and CSP. J. of Computer and System Sciences 85 (2017), 38–56. https://doi.org/10.1016/j.jcss.2016.10.007

[43] Serge Gaspers, Sebastian Ordyniak, M. S. Ramanujan, Saket Saurabh, and Stefan Szeider. 2016. Backdoors to q-Horn. Algorithmica 74, 1 (2016), 540–557. https://doi.org/10.1007/s00453-014-9958-9

[44] Serge Gaspers, Sebastian Ordyniak, and Stefan Szeider. 2017. Backdoor Sets for CSP. In The Constraint Satisfaction Problem: Complexity and Approximability. Andrei Krokhin and Stanislav Zivny (Eds.), Dagstuhl Follow-Ups, Vol. 7. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 137–157. https://doi.org/10.4230/DFU.Vol7.15301.137

[45] Serge Gaspers and Stefan Szeider. 2012. Backdoors to Satisfaction. In The Multivariate Algorithmic Revolution and Beyond - Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday (Lecture Notes in Computer Science, Vol. 7370), Hans L. Bodlaender, Rod Downey, Fedor V. Fomin, and Daniël Marx (Eds.). Springer Verlag, 287–317.

[46] Georg Gottlob. 1992. Complexity Results for Nonmonotonic Logics. J. Logic Comput. 2, 3 (1992), 397–425. https://doi.org/10.1093/logcom/2.3.397

[47] Georg Gottlob. 1995. Translating Default Logic into Standard Autoepistemic Logic. J. of the ACM 42, 4 (1995), 711–740.

[48] Georg Gottlob, Reinhard Pichler, and Fang Wei. 2010. Bounded treewidth as a key to tractability of knowledge representation and reasoning. Artificial Intelligence 174, 1 (2010), 105–132. https://doi.org/10.1016/j.artint.2009.10.003

[49] Georg Gottlob, Francesco Scarcello, and Martha Sideri. 2002. Fixed-parameter complexity in AI and nonmonotonic reasoning. Artificial Intelligence 138, 1-2 (2002), 55–86. https://doi.org/10.1016/S0004-3702(02)00182-0

[50] Georg Gottlob and Stefan Szeider. 2006. Parameterized algorithms for artificial intelligence, constraint satisfaction, and database problems. Comput. J. 51, 3 (2008), 303–325. Survey paper.

[51] Markus Hecher. 2022. Treewidth-aware reductions of normal ASP to SAT – Is normal ASP harder than SAT after all? Artificial Intelligence 387, 1-2 (2022), 103651. https://doi.org/10.1016/j.artint.2021.103651

[52] Pieter Van Hertum, Marcus Kramer, Bart Bogaerts, and Marc Denecker. 2016. Distributed Autoepistemic Logic and its Application to Access Control. In Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI’16), Gerhard Brewka (Ed.). IJCAI/AAAI Press, New York, NY, USA, 1286–1292.

[53] Alfred Horn. 1951. On Sentences Which are True of Direct Unions of Algebras. J. Symbol. Log., 16, 1 (1951), 14–21.

[54] Tomi Janhunen. 1999. On the intertranslatability of non-monotonic logics. Ann. Math. Artif. Intell. 27, 1-4 (1999), 79–128.

[55] Neil D. Jones, Y. Edmund Lien, and William T. Laaser. 1976. New Problems Complete for Nondeterministic Loc Space. Math. Syst. Theory 10 (1976), 1–17.

[56] Rafael Kiesel, Ernan Acar, and Stefan Schlobach. 2018. Preferential Default Reasoning on the Semantic Web. In Description Logics (CEUR Workshop Proceedings, Vol. 2211). CEUR-WS.org.

[57] Kurt Konolige. 1988. On the Relation Between Default and Autoepistemic Logic. Artificial Intelligence 35, 3 (1988), 343–382.

[58] Wei Li, Yuefei Sui, and Yuhui Wang. 2017. The propositional normal default logic and the infinite/infinite injury priority method. Sci. China Inf. Sci. 60, 9 (2017), 092107:1–092107:10.

[59] Paolo Liberatore and Marco Schaerf. 2005. The complexity of model checking for propositional default logics. Data Knowl. Eng. 55, 2 (2005), 189–202.

[60] Yasir Mahmood, Arne Meier, and Johannes Schmidt. 2020. Parameterised Complexity of Abduction in Schaefer’s Framework. In Logical Foundations of Computer Science - International Symposium, LFCS 2020, Deerfield Beach, FL, USA, January 4-7, 2020, Proceedings (Lecture Notes in Computer Science, Vol. 11972), Sergei N. Artemov and Anil Nerode (Eds.). Springer, 195–213. https://doi.org/10.1007/978-3-030-
36755-8_13

[61] Yasir Mahmood, Arne Meier, and Johannes Schmidt. 2021. Parameterized complexity of abduction in Schaefer’s framework. J. Logic Comput. 31, 1 (2021), 266–296. https://doi.org/10.1093/LOGCOM/EXAA079

[62] V. Wiktor Marek and Miroslaw Truszczynski. 1989. Relating Autoepistemic and Default Logics. In KR. Morgan Kaufmann, 276–288.

[63] Victor W. Marek and Miroslaw Truszczynski. 1993. Nonmonotonic Logic: context-dependent reasoning. Springer Verlag, Berlin, Germany. https://doi.org/10.1007/978-3-662-02906-0

[64] John McCarthy. 1980. Circumscription – A form of non-monotonic reasoning. Artificial Intelligence 13 (April 1980), 27–39. Issue 1-2. https://doi.org/10.1016/0004-3702(80)90011-9

[65] J. McCarthy. 1980. Circumscription – a form of non-monotonic reasoning. Artificial Intelligence 13, 1-2 (1980), 27–39.

[66] Drew McDermott and Jon Doyle. 1980. Non-monotonic logic I. Artificial Intelligence 13, 41-72 (1980). https://doi.org/10.1016/0004-3702(80)90012-0

[67] Drew V. McDermott. 1982. Nonmonotonic Logic II: Nonmonotonic Modal Theories. J. of the ACM 29, 1 (1982), 35–57.

[68] Drew V. McDermott and Jon Doyle. 1980. Non-Monotonic Logic I. Artificial Intelligence 13, 1-2 (1980), 41–72.

[69] Arne Meier. 2020. Incremental FPT Delay. Algorithms 13, 5 (2020), 122. https://doi.org/10.3390/a13050122

[70] Arne Meier. 2020. Parametrised enumeration. https://doi.org/10.15488/9427 Habilitation thesis, Leibniz Universität Hannover..

[71] Arne Meier, Sebastian Ordyniak, M. S. Ramanujan, and Irena Schindler. 2018. Backdoors for Linear Temporal Logic. Algorithmica (18 Sep 2018). https://doi.org/10.1007/s00453-018-0515-5

[72] Arne Meier, Irina Schindler, Johannes Schmidt, Michael Thomas, and Heribert Vollmer. 2015. On the parameterized complexity of non-monotonic logics. Archive for Mathematical Logic 54, 5-6 (2015), 685–710. https://doi.org/10.1007/s00153-015-0455-x

[73] Arne Meier, Irina Schindler, Johannes Schmidt, Michael Thomas, and Heribert Vollmer. 2015. On the parameterized complexity of non-monotonic logics. Archive for Mathematical Logic 54, 5-6 (2015), 685–710. https://doi.org/10.1007/s00153-015-0455-x

[74] Robert C. Moore. 1985. Semantical considerations on modal logic. Artificial Intelligence 25 (Jan. 1985), 75–94. Issue 1.

[75] Robert C. Moore. 1985. Semantical Considerations on Nonmonotonic Logic. Artificial Intelligence 25, 1 (1985), 75–94. https://doi.org/10.1016/0004-3702(85)90042-6

[76] Naomi Nishimura, Prabhakar Ragde, and Stefan Szeider. 2004. Detecting Backdoor Sets with Respect to Horn and Binary Clauses. In Proceedings of the 7th International Conference on Theory and Applications of Satisfiability Testing (SAT’04) (Lecture Notes in Computer Science, Vol. 3542), Holger H. Hoos and David G. Mitchell (Eds.). Springer Verlag, Vancouver, BC, Canada, 96–103.

[77] Sebastian Ordyniak, Daniel Paulusma, and Stefan Szeider. 2013. Satisfiability of acyclic and almost acyclic CNF formulas. Theoretical Computer Science 481 (2013), 85–99. https://doi.org/10.1016/j.tcs.2012.05.059

[78] Christos H. Papadimitriou. 1994. Computational Complexity. Addison-Wesley. 523 pages.

[79] Charles Sanders Peirce. 1935. Collected Papers of Charles Sanders Peirce. Oxford University Press.

[80] C. S. Peirce, C. Hartshorne, and P. Weiss. 1932. The Collected Papers of Charles Sanders Peirce. Cambridge University Press, Cambridge.

[81] Andreas Pfeilner, Stefan Rümmele, and Stefan Szeider. 2013. Backdoors to Abduction. In Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI’13), Francesca Rossi (Ed.). The AAAI Press, Beijing, China, 1046–1052.

[82] Sopo Pkhakadze and Hans Tompits. 2020. Sequent-Type Calculi for Three-Valued and Disjunctive Default Logic. Axioms 9, 3 (2020), 84.

[83] Raymond Reiter. 1980. A logic for default reasoning. Artificial Intelligence 13 (April 1980), 81–132. Issue 1. https://doi.org/10.1016/0004-3702(80)90014-4

[84] Riccardo Rosati. 1999. Model Checking for Nonmonotonic Logics: Algorithms and Complexity. In Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI’99), Thomas Dean (Ed.). The AAAI Press, Stockholm, Sweden.

[85] Marko Samer and Stefan Szeider. 2009. Fixed-Parameter Tractability. In Handbook of Satisfiability, Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh (Eds.). IOS Press, Chapter 13, 425–454.

[86] Thomas J. Schaefer. 1978. The complexity of satisfiability problems. In Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC’78), Richard J. Lipton, Walter A. Burkhard, Walter J. Savitch, Emily P. Friedman, and Alfred V. Aho (Eds.). Assoc. Comput. Mach., New York, San Diego, CA, USA, 216–226. https://doi.org/10.1145/800133.804350

[87] Claus-Peter Schnorr. 1981. On Self-Transformable Combinatorial Problems. In Mathematical Programming at Oberwolfach, H. König, B. Korte, and K. Ritter (Eds.). Mathematical Programming Studies, Vol. 14. Springer Verlag, 225–243. https://doi.org/10.1007/BFB0120931

[88] Alexander Semenov, Daniil Chivilikhin, Stepan Kochemazov, and Irgragim Dzhiblavi. 2023. Probabilistic Generalization of Backdoor Trees with Application to SAT. In Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAA1’23). The AAAI Press, Article 456, 9 pages. https://doi.org/10.1609/aaai.v37i04.25525

[89] Alexander Semenov, Artem Pavlenko, Daniil Chivilikhin, and Stepan Kochemazov. 2022. On Probabilistic Generalization of Backdoors in Boolean Satisfiability. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI’22), Vasant Honavar and Matthijs Spaan (Eds.), Vol. 36. The AAAI Press, 10353–10361. https://doi.org/10.1609/aaai.v36i09.21277

[90] Jonathan P. Stillman. 1990. It’s not my default: The complexity of membership problems in restricted propositional default logics. In Proceedings of the 8th National conference on Artificial Intelligence (AAAI’90), Thomas Dietterich and William Swartout (Eds.), Vol. 1. The AAAI Press, Boston, MA, USA, 571–578.
[91] Jonathan P. Stillman. 1990. The Complexity of Horn Theories with Normal Unary Defaults. In Proceedings of the 8th Canadian Artificial Intelligence Conference (AI’90).

[92] Ezgi Iraz Su. 2020. A Unifying Approach for Nonmonotonic S4F, (Reflexive) Autoepistemic Logic, and Answer Set Programming. Fundam. Informaticae 176, 3-4 (2020), 205–234.

[93] Stefan Szeider. 2004. On fixed-parameter tractable parameterizations of SAT. In Proceedings of the 6th International Conference Theory and Applications of Satisfiability (SAT’03) (Lecture Notes in Computer Science, Vol. 2919), Enrico Giunchiglia and Armando Tacchella (Eds.). Springer Verlag, 188–202. Selected and Revised Papers.

[94] Michael Thomas. 2010. On the complexity of fragments of nonmonotonic logics. Ph. D. Dissertation. University of Hanover.

[95] Ryan Williams, Carla Gomes, and Bart Selman. 2003. Backdoors To Typical Case Complexity. In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI’03). Georg Gottlob and Toby Walsh (Eds.). Morgan Kaufmann, Acapulco, Mexico, 1173–1178.

[96] Ryan Williams, Carla Gomes, and Bart Selman. 2003. On the Connections Between Backdoors, Restarts, and Heavy-tailedness in Combinatorial Search. In Informal Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing (SAT’03). Portofino, Italy, 222–230.

[97] Anthony P. Young, Sanjay Modgil, and Odinaldo Rodrigues. 2016. Prioritised Default Logic as Rational Argumentation. In Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems (AAMAS’16), John Thangarajah, Karl Tuyls, Catholijn M. Jonker, and Stacy C. Marsella (Eds.). Assoc. Comput. Mach., New York, 626–634.