Annals of Mathematics and Artificial Intelligence, 15(1995):289–323

On the Computational Cost of Disjunctive Logic Programming: Propositional Case*

THOMAS EITER AND GEORG GOTTLOB
Christian Doppler Laboratory for Expert Systems,
Institut für Informationssysteme, Technische Universität Wien
Paniglgasse 16, A-1040 Wien, Austria
E-mail: `(eiter|gottlob)@vexpert.dbai.tuwien.ac.at`

This paper addresses complexity issues for important problems arising with disjunctive logic programming. In particular, the complexity of deciding whether a disjunctive logic program is consistent is investigated for a variety of well-known semantics, as well as the complexity of deciding whether a propositional formula is satisfied by all models according to a given semantics. We concentrate on finite propositional disjunctive programs with as wells as without integrity constraints, i.e., clauses with empty heads; the problems are located in appropriate slots of the polynomial hierarchy. In particular, we show that the consistency check is Σ_2^P-complete for the disjunctive stable model semantics (in the total as well as partial version), the iterated closed world assumption, and the perfect model semantics, and we show that the inference problem for these semantics is Π_2^P-complete; analogous results are derived for the answer sets semantics of extended disjunctive logic programs. Besides, we generalize previously derived complexity results for the generalized closed world assumption and other more sophisticated variants of the closed world assumption. Furthermore, we use the close ties between the logic programming framework and other nonmonotonic formalisms to provide new complexity results for disjunctive default theories and disjunctive autoepistmic literal theories.

1 Introduction

Disjunctive logic programming is a generalization of logic programming, where the heads of clauses may consist of disjunctions of atoms. This generalization has recently attracted much interest by the research community, and several alternative semantics have been proposed. A comprehensive survey can be found in the recent book by Lobo, Minker, and Rajasekar [34].[1]

The aim of this paper is a complexity analysis of various semantics for disjunctive logic programming in the propositional case. We consider the following semantics.

*Parts of the results in this paper appeared in form of an abstract in the Proceedings of the Twelth ACM SIGACT SIGMOD-SIGART Symposium on Principles of Database Systems (PODS-93), pp. 158–167. Other parts appeared in shortened form in the Proceedings of the International Logic Programming Symposium, Vancouver, October 1993 (ILPS-93), pp. 266–278. MIT Press.

[1]We use the term *disjunctive logic program* for what is elsewhere called a *normal disjunctive logic program* with integrity constraints (which are disregarded in [34]). This terminology may differ from that used elsewhere.

- The Disjunctive Stable Model Semantics (DSM) defined by Przymusinski [49], which adapts Gelfond and Lifschitz's Stable Semantics [24] to disjunctive logic programming.
- The Answer Set Semantics (ANSW) of Gelfond and Lifschitz [25] for the extension of disjunctive logic programming by classical negation, which is an adaptation of the stable model semantics to deal with negative literals. Informally, an answer set can be seen as a kind of stable model of an extended disjunctive logic program.
- The Partial Disjunctive Stable Model Semantics (PDSM) by Przymusinski [49], which extends the Well-Founded Semantics of van Gelder, Ross, and Schlipf [65].
- The Iterated Closed World Assumption (ICWA) by Gelfond, Przymusinska, and Przymusinski [28].
- The Perfect Models Semantics (PERF) by Przymusinski [48].
- Various forms of the Closed World Assumption (CWA) including the Generalized CWA (GCWA) by Minker [43], the Extended GCWA (EGCWA) by Yahya and Henschen [67], the Careful CWA (CCWA) by Gelfond and Przymusinska [26], and the Extended CWA (ECWA) by Gelfond, Przymusinska, and Przymusinski [28], which coincides in the finite propositional case with McCarthy's circumscription (CIRC) [41, 42], and further variants such as the Disjunctive Database Rule (DDR) of Ross and Topor [55], which is equivalent to the Weak GCWA (WGCWA) of Rajasekar, Lobo, and Minker [51], and the Possible Models Semantics (PMS) of Sakama [56], which is equivalent to Chan's Possible Worlds Semantics (PWS) [13].

For each semantics $\mathcal{S}$, we consider the following problems. Given a finite propositional disjunctive logic program P, decide whether P has a model under semantics $\mathcal{S}$ ($\mathcal{S}$-**Consistency**), and deciding whether a given propositional formula F is satisfied by all legal models of P according to the semantics $\mathcal{S}$ ($\mathcal{S}$-**Entailment**).

Basic complexity results for the last group (various forms of the CWA) have already been derived in [60, 11, 13, 21]. In the present paper we sharpen some of the known results and state, for the sake of completeness of the analysis, some minor new results. All other results are novel.

This paper complements recent surveys on complexity results for nonmonotonic reasoning and logic programming [12, 59]. By the work of Apt, Blair, Cholak, Chomicki, Marek, Nerode, Remmel, Schlipf, and Subrahmanian [1, 9, 7, 36, 35, 58, 59, 16] the complexity of general logic programming is quite well-understood today. Practical methods for coping with complexity in disjunctive logic programming have been investigated in [17, 18]. The main results of the present paper state that $\mathcal{S}$-**Consistency** is Σ_2^P-complete for DSM, ANSW, PDSM and PERF while the problem is less complex for the other semantics (NP-complete or, in case of ICWA, even polynomial) and furthermore that $\mathcal{S}$-**Entailment** is Π_2^P-complete for all considered semantics except for some versions of CWA.

Σ_2^P-hardness of $\mathcal{S}$-**Consistency** in case of DSM, ANSW, and PDSM may be intuitively explained by an additional source of complexity in terms of a fixpoint condition on a model which interacts with an independent minimality criterion. PERF similarly imposes a nonmonotonic minimality condition on models.

An intuitive explanation for Π_2^P-hardness of $\mathcal{S}$-**Entailment** is that due to some minimality criterion, the problem of identifying a disjunctive model is difficult and involves an (at least) coNP-hard test. This constitutes a source of complexity "orthogonal" to the source given by the potentially exponentially many candidates for a (disjunctive) model of P. As a consequence of the results, under the widely accepted hypothesis that the polynomial hierarchy does not collapse to some class below Π_2^P, polynomial inference algorithms using an oracle for classical inference do not exist.

Inference of a formula under GCWA or CCWA is an interesting problem: the best upper bound we can provide by a nontrivial membership proof is $\Delta_3^P[O(\log n)]$, which is "mildly" harder than Π_2^P or Σ_2^P. Completeness for this class would entail Σ_2^P-hardness; however, it is not clear how to reduce a Σ_2^P-complete problem to this problem. Inference of a literal under GCWA is in Π_2^P, as it suffices in this case to check a restricted set of models.

Chan [13] shows that inference under DDR, WGCWA, PWS, and PMS is tractable if integrity clauses (i.e. clauses with empty heads) are not allowed; this in fact constitutes the only cases of tractability for the considered semantics.

In addition, based on the complexity results for disjunctive logic programming, we are able to derive new complexity results for nonmonotonic logics such as disjunctive default logic [27] and autoepistemic logic [45].

The paper is organized as follows. After introducing basic concepts and notation in Section 2, the analysis starts in Section 3 with PSM and PDSM. In Section 4, we deal with the different versions of the CWA, and in Section 5 we analyze the ICWA-semantics for stratified programs. After treating the Perfect Models Semantics in Section 6 and the Answer Set Semantics for programs extended with classical negation in Section 7, we establish in Section 8 new complexity results for disjunctive default logic and autoepistemic logic. Section 9 concludes the paper.

2 Preliminaries and Notation

A comprehensive treatment of disjunctive logic programming is given in [34], to which the reader is referred for unexplained concepts. A disjunctive logic program clause (DLP clause, or simply clause) C is a formula

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k, \mathit{not}\, b_{k+1}, \cdots, \mathit{not}\, b_m, \qquad m, n \geq 0$$

where all a_i and b_j are atoms from a (fragment of a) first-order language, "," denotes conjunction, and "*not*" is a negation-by-default operator. An extension to DLP clauses, which will be considered in Section 7, allows literals under classical negation instead of atoms.

Notice that we also allow clauses with empty head (i.e. $n = 0$), which we call *integrity clauses*.[2] Integrity clauses without "*not*" are also called *negative clauses*.

Permitting integrity clauses does not affect the complexity of inference under the considered semantics in the general case, but most likely makes deciding whether a program

[2] If some designated atom $\mathbf{f}$ representing falsity is used, clauses in which the only atom in the head is $\mathbf{f}$ are equivalent to integrity clauses.

is consistent under a certain semantics more complex. We will highlight the difference of allowing/disallowing integrity clauses in the analysis.

A *disjunctive logic program* (DLP) P is a finite collection of DLP clauses. A DLP P is *normal* if "*not*" is allowed to occur in its clauses, and *not*-free if "*not*" does not occur in P. A DLP P is *positive* if it is *not*-free and it contains no integrity clauses. A DLP in which each clause has one (resp. at most one) atom in the head ($n = 1$ resp. $n \leq 1$) is called *definite* (resp. *nondisjunctive*).

Example 1
Consider the following propositional DLP P:

$$P = \quad a \vee b \leftarrow c \quad b \leftarrow not\, a, not\, c$$

P contains no integrity clauses but is not positive, as "not" occurs in P.

In the rest of this paper, we restrict our considerations to finite *propositional* DLPs, i.e. (finite) DLPs where the atoms are propositions. We omit the phrase "finite propositional" when referring to a propositional DLP.

The various semantics of disjunctive logic programs we consider, as well as others not treated here, can be alternatively characterized proof-theoretically, by fixpoints, or in terms of models, cf. [34, 22]. We refer to model theoretic characterizations, which are most useful for the purpose of this paper.

Interpretations and models of DLPs are restricted to Herbrand interpretations resp. Herbrand models; thus "interpretation" and "model" refer to "Herbrand interpretation" and "Herbrand model" throughout the text.

For any DLP P, we denote by $A(P)$ the set of propositional atoms of P and by $\mathrm{M}(P)$ the set of all models of P if "*not*" is interpreted by classical negation. $I \models F$ resp. $\mathcal{C} \models F$ denotes satisfaction of a propositional formula F by an interpretation I resp. a collection $\mathcal{C}$ of interpretations. $Cn(\mathcal{F})$ denotes the set of all logical consequences from the set of formulae $\mathcal{F}$.

The models of a DLP P under a semantics $\mathcal{S}$ are a subset of $\mathrm{M}(P)$, which we denote by $\mathcal{S}(P)$. P is called *$\mathcal{S}$-consistent* iff $\mathcal{S}(P) \neq \emptyset$. A DLP P *entails* a propositional formula F under semantics $\mathcal{S}$ iff $\mathcal{S}(P) \models F$.

Example 2
For the above program

$$P = \quad a \vee b \leftarrow c \quad b \leftarrow not\, a, not\, c$$

we have $\mathrm{M}(P) = \{\{b\}, \{a\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\}\}$. *Hence,* $\mathrm{M}(P) \models a \vee b$ *and* $\mathrm{M}(P) \not\models \neg c$.

For every DLP P, the partial order $\leq$ on $\mathrm{M}(P)$ is defined by $M \leq M'$ iff $M \subseteq M'$, i.e., all atoms true in M are also true in M'. $M < M'$ stands for $M \leq M'$ and $M' \not\leq M$. For any partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ of $A(P)$, the preorder $\leq_{\hat{P};\hat{Z}}$ is defined by $M \leq_{\hat{P};\hat{Z}} M'$ iff $M \cap \hat{Q} = M' \cap \hat{Q}$ and $M \cap \hat{P} \subseteq M' \cap \hat{P}$. Note that $\leq_{\hat{P};\hat{Z}}$ coincides with $\leq$ if $\hat{P} = A(P)$.

Model $M \in \mathrm{M}(P)$ is *minimal* (resp. $\langle \hat{P}; \hat{Z} \rangle$*-minimal*) iff no $M' \in \mathrm{M}(P)$ satisfies $M' < M$, (resp. $M' \leq_{\hat{P};\hat{Z}} M$ and $M \not\leq_{\hat{P};\hat{Z}} M'$). The collection of all minimal (resp. $\langle \hat{P}; \hat{Z} \rangle$-minimal) models of P is denoted by $\mathrm{MM}(P)$ (resp. $\mathrm{MM}(P; \hat{P}; \hat{Z})$).

Example 3
Consider the above program

$$P = \quad a \vee b \leftarrow c \qquad b \leftarrow not\, a, not\, c$$

again. Then, $\mathrm{MM}(P) = \{\{a\}, \{b\}\}$, *and for the partition* $\langle \{a\}, \{b\}, \{c\} \rangle$, *we have* $\mathrm{MM}(P; \{a\}; \{c\}) = \{\{b\}, \{b, c\}, \{a\}, \{a, c\}\}$.

We assume that the reader has some background on the concept of NP-completeness [23, 30]. Upon the class NP, the polynomial hierarchy (PH) has been defined as a subrecursive analog to the Kleene arithmetical hierarchy. For any complexity class C, let P^C (resp. NP^C) denote the decision problems for which there exists a polynomial-time bounded deterministic (resp. nondeterministic) Turing-reduction to any problem $\pi \in C$, i.e. the decision problems solvable in polynomial time by some deterministic (resp. nondeterministic) oracle Turing machine with an oracle for any problem in C. The classes Δ_k^P, Σ_k^P, and Π_k^P of PH are defined as follows:

$$\Delta_0^P = \Sigma_0^P = \Pi_0^P = \mathrm{P}$$

and for all $k \geq 0$,

$$\Delta_{k+1}^P = \mathrm{P}^{\Sigma_k^P}, \;\; \Sigma_{k+1}^P = \mathrm{NP}^{\Sigma_k^P}, \;\; \Pi_{k+1}^P = \mathrm{co}\Sigma_{k+1}^P.$$

In particular, $\mathrm{NP} = \Sigma_1^P$, $\mathrm{coNP} = \Pi_1^P$, $\Sigma_2^P = \mathrm{NP}^{\mathrm{NP}}$, and $\Pi_2^P = \mathrm{coNP}^{\mathrm{NP}}$. The classical NP-complete problem is to decide if a collection $\mathcal{C} = \{L_{i,1} \vee \cdots \vee L_{i,n_i} : 1 \leq i \leq m\}$ of propositional clauses is simultaneously satisfiable (SAT). This problem is still NP-complete if each clause $L_{i,1} \vee \cdots \vee L_{i,n_i}$ in $\mathcal{C}$ contains only positive literals or only negative literals (MSAT) [23]. The most prominent Σ_2^P-complete problem is to decide the validity of a formula from $\mathrm{QBF}_{2,\exists}$, the set of quantified Boolean formulae of the form $\exists x_1 \cdots \exists x_n \forall y_1 \cdots \forall y_m E$, where $E = E(x_1, \ldots, x_n, y_1, \ldots, y_m)$ is a propositional formula built from atoms $x_1, \ldots, x_n, y_1, \ldots, y_m$.

The classes Δ_k^P have been refined by bounding the number of queries to the oracle [66, 30]. $\Delta_{k+1}^P[f(n)]$, $k \geq 1$, denotes the class of decision problems that are polynomially solvable with at most $f(n)$ calls to a Σ_k^P oracle, where $f(n)$ is a function in the size n of the problem instance. In particular, $\Delta_{k+1}^P[O(\log n)]$ allows logarithmically many oracle queries.

In this paper, we consider the following problems on DLPs for different semantics $\mathcal{S}$.

$\mathcal{S}$-Consistency: Given a DLP P, decide if P is $\mathcal{S}$-consistent.

$\mathcal{S}$-Entailment: Given a DLP P and a propositional formula F, decide whether $\mathcal{S}(P) \models F$.

The restriction of $\mathcal{S}$-**Entailment** to the case in which F is a literal is of particular importance. It appears that for each semantics the presented lower complexity bound applies to this case.

The notion of entailment in terms of consequence from all models under $\mathcal{S}$ is commonly called *cautious reasoning* or *skeptical reasoning*. An alternative version of entailment is in terms of consequence from some model under $\mathcal{S}$, which is known as *brave reasoning* or *credulous reasoning*. Complexity results for brave reasoning under the considered semantics can be straightforwardly derived from our results on cautious reasoning. Hence, we will not consider brave reasoning in our analysis.

3 Stable model semantics

The Stable model semantics for logic programs [24, 5] has been widely acknowledged and is one of the most important semantics for logic programs. Concerning the computational properties, Marek and Truszczyński [38] and independently Bidoit and Froidevaux [6] showed that deciding whether a definite logic program has a stable model is NP-complete. Marek and Truszczyński further showed that deciding whether an atom belongs to every stable model of a definite logic program is coNP-complete [39].

The stable model semantics was generalized in a natural way to disjunctive logic programs by Przymusinski [49] and independently by Gelfond and Lifschitz in terms of answer sets in the more general framework of extended DLPs (see Section 7). Besides the generalization of total (i.e. 2-valued) stable model semantics, Przymusinski presented in [48] also a generalization of partial (i.e. 3-valued) stable semantics from definite programs to disjunctive logic programs. Przymusinski demonstrated that the partial stable model semantics coincides for definite programs with the well-founded semantics of van Gelder et al. [65], and can be thus seen as proper generalization of the well-founded semantics to disjunctive programs. Other generalizations of the well-founded semantics which we do not examine here are presented in [54, 3]. We study in this section both total and partial stable model semantics.

We start with total stable model semantics. Given a DLP P and an interpretation I, the *Gelfond-Lifschitz transformation* P^I of P with respect to I is defined as follows.

(i) if $a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k, not\, b_{k+1}, \cdots, not\, b_m \in P$
and for all $k < i \leq m$, $b_i \notin I$, then $a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k \in P^I$

(ii) nothing else is in P^I.

Notice that P^I is *not*-free, and that P^I coincides with P if P is *not*-free.

An interpretation I is called a *(disjunctive) stable model* of P iff $I \in \mathrm{MM}(P^I)$ [49]. $\mathrm{DSM}(P)$ denotes the collection of all stable models of P, i.e.

$$\mathrm{DSM}(P) = \{I : I \in \mathrm{MM}(P^I)\}.$$

Example 4
Let

$$P = \quad a \vee b \leftarrow c \quad b \leftarrow not\, a, not\, c \quad a \vee c \leftarrow not\, b.$$

Consider $I = \{b\}$. *Then,*

$$P^I = \quad a \vee b \leftarrow c \quad b \leftarrow .$$

Check that I is a minimal model of P^I; thus, I is a stable model of P.

Each stable model is in fact a model of P and has the following property.

Proposition 1 (cf. [49][3])
Every stable model of a DLP P is a minimal model of P, i.e. $\mathrm{DSM}(P) \subseteq \mathrm{MM}(P)$.

Notice that DSM-**Consistency** is trivial if P is a positive DLP, since each such P has a stable model. ($\mathrm{M}(P) \neq \emptyset$ and $P^I = P$ for any I holds in this case, hence the existence of a stable model can be easily seen from the definition.) If the heads in the rules are allowed to be empty, we obtain the following.

Proposition 2
DSM-**Consistency** *restricted to instances of P which are not-free is* NP-*complete.*

Proof. Let $\mathcal{C}$ be an instance of SAT. Each such $\mathcal{C}$ can be trivially rewritten as a logically equivalent *not*-free DLP P and vice versa, such that $\mathcal{C}$ is satisfiable iff $\mathrm{M}(P) \neq \emptyset$. It holds that P has a stable model iff $\mathrm{MM}(P) \neq \emptyset$, which is equivalent with $\mathrm{M}(P) \neq \emptyset$. From this, the result follows. ■

The complexity of checking DSM-consistency remains unchanged if we allow nondisjunctive programs besides *not*-free programs in the input. (Membership for nondisjunctive programs in NP follows as P^I is a collection of Horn clauses.) For DLPs which allow disjunction in the head and simultaneous occurrence of "*not*", we obtain the following result.

Theorem 3
DSM-**Consistency** *is Σ_2^P-hard. This holds even if P contains no integrity clauses and "not" has a single occurrence in P.*

Proof. We show this by the following reduction of deciding the validity of a quantified Boolean formula

$$\Phi = \exists x_1 \cdots \exists x_n \forall y_1 \cdots \forall y_m E, \quad n, m \geq 1.$$

We may assume that $E = D_1 \vee \cdots \vee D_r$ and each $D_i = L_{i,1} \wedge L_{i,2} \wedge L_{i,3}$ is a conjunction of literals $L_{i,j}$ over atoms $x_1, \ldots, x_n, y_1, \ldots, y_m$; deciding if such a Φ is valid is still Σ_2^P-hard [63]. Let $v_1, \ldots, v_n$ and $z_1, \ldots, z_m, w$ be new propositional atoms and define the following

[3] Although this proposition is formulated only for programs without integrity clauses, the proof applies to the general case.

DLP P:

$$
\begin{array}{llll}
x_i \vee v_i \leftarrow & & & \text{for each } i = 1, \dots, n \\
y_j \vee z_j \leftarrow & y_j \leftarrow w & z_j \leftarrow w & \\
w \leftarrow y_j, z_j & & & \text{for each } j = 1, \dots, m \\
w \leftarrow \sigma(L_{k,1}), \sigma(L_{k,2}), \sigma(L_{k,3}) & & & \text{for each } k = 1, \dots, r \\
w \leftarrow \mathit{not}\, w & & &
\end{array}
$$

where σ maps literals from atoms $x_1, \dots, x_n$, $y_1, \dots, y_m$ to atoms as follows:

$$
\sigma(L) = \begin{cases} v_i & \text{if } L = \neg x_i \text{ for some } i = 1, \dots, n \\ z_j & \text{if } L = \neg y_j \text{ for some } j = 1, \dots, m \\ L & \text{otherwise} \end{cases}
$$

Intuitively, v_i corresponds to $\neg x_i$ and z_j corresponds to $\neg y_j$. [4]

If M is a stable model of P, then $w \in M$ must hold (this is assured by the clause $w \leftarrow \mathit{not}\, w$). Thus P^M consists of all clauses of P except $w \leftarrow \mathit{not}\, w$. Consequently, for every $1 \leq j \leq m$, y_j and z_j must be in M, which follows from the clauses $y_j \leftarrow w$, $z_j \leftarrow w$. Further, for every $i = 1, \dots, n$ it must hold that $x_i \in M$ or $v_i \in M$, which follows from the clause $x_i \vee v_i \leftarrow$; from the condition $M \in \mathrm{MM}(P^M)$, however, it follows that not both $x_i \in M$ and $v_i \in M$ hold, for otherwise $M' = M - \{v_i\}$ is a model of P^M such that $M' < M$. Consequently, exactly one of $x_i \in M$ and $v_i \in M$ must be true.

We show that P has a stable model iff the formula Φ is valid.

"$\Rightarrow$": Let M be a stable model of P. Let the truth assignment φ to the atoms $x_1, \dots, x_n$ be defined by

$$
\varphi(x_i) = \begin{cases} \mathit{true} & \text{if } x_i \in M \\ \mathit{false} & \text{if } v_i \in M \end{cases}, \quad \text{for } i = 1, \dots, n.
$$

Notice that φ is well-defined. Since M is a minimal model of P^M, it follows that each interpretation I that coincides with M on $x_1, v_1, \dots, x_n, v_n$ and contains exactly one of y_j, z_j for each $j = 1, \dots, m$, and does not contain w, is not a model of P^M. For otherwise I would be a model of P^M such that $I < M$, which contradicts that M is a stable model of P. Since I is not model of P^M, there must exist some $i = 1, \dots, r$ such that $I \models \sigma(L_{i,j})$, for $1 \leq j \leq 3$. It follows that for every extension of φ to the atoms $y_1, \dots, y_m$, D_i is true for some $i = 1, \dots, k$. Thus E evaluates to true. Consequently, $\exists x_1 \cdots \exists x_n \forall y_1 \cdots \forall y_m E$, i.e. Φ, is valid.

"$\Leftarrow$": assume that Φ is valid. That is, there exists a truth assignment φ to the atoms $x_1, \dots, x_n$ such that every extension of φ to $y_1, \dots, y_m$ satisfies E. Let I be the following interpretation:

$$
\begin{aligned}
I \;=\; & \{x_i : \varphi(x_i) = \mathit{true}, i = 1, \dots, n\} \cup \{v_i : \varphi(x_i) = \mathit{false}, i = 1, \dots, n\} \cup \\
& \{y_1, z_1, \dots, y_m, z_m, w\}.
\end{aligned}
$$

[4] Actually the clauses $w \leftarrow y_j, z_j$ are redundant and could be omitted. We include them for intelligibility and ease of argumentation.

Notice that P^I are the clauses of P except $w \leftarrow not\, w$ and that I is a model of P^I. We show that I is a minimal model of P^I, which implies that I is a stable model of P. Assume to the contrary that $I \notin \mathrm{MM}(P^I)$, i.e. there exists $M \in \mathrm{M}(P^I)$ such that $M < I$. We observe that M must coincide with I on $x_1, v_1, \ldots, x_n, v_n$, that $w \notin M$, and that exactly one of y_j, z_j is in M for every $j = 1, \ldots, m$. By the extension property of φ, however, it follows that for the extension of φ to $y_1, \ldots, y_m$ defined by $\varphi(y_i) = \mathit{true}$ if $M \models y_i$ and $\varphi(y_i) = \mathit{false}$ if $M \models z_i$, $1 \leq i \leq m$, there must exist some $i = 1, \ldots, k$ such that $M \models \sigma(L_{i,j})$, for $1 \leq j \leq 3$. But this implies $w \in M$, contradiction. Thus I is a minimal model of P^I, and consequently I is a stable model of P.

Since P is constructible from Φ in polynomial time, the theorem follows. ∎

We obtain as a corollary the following result.

Corollary 4
DSM-**Entailment** *is* Π_2^P*-hard.*

Proof. Let P' be the DLP obtained by adding to DLP P the clause $p \leftarrow$, where the atom p does not occur in P. Every stable model of P' satisfies p but does not satisfy $\neg p$. Hence, $\mathrm{DSM}(P') \models \neg p$ if and only if $\mathrm{DSM}(P') = \emptyset$, which is equivalent to $\mathrm{DSM}(P) = \emptyset$. Theorem 3 implies that deciding $\mathrm{DSM}(P) = \emptyset$ is Π_2^P-hard, from which the result follows. ∎

It is important that this hardness result can be noticeably strengthened. In fact, the result holds even if P is a positive DLP.

Theorem 5
DSM-**Entailment** *is* Π_2^P*-hard, even if P is positive and F is a literal.*

Proof. Consider the DLP P from the proof of Theorem 3 and let P' be the program P except the clause $w \leftarrow not\, w$. Recall that every stable model M of P must contain w. Since $P^M = P'$, it holds that M is a minimal model of P'. On the other hand, every $M \in \mathrm{MM}(P')$ such that $w \in M$, is a stable model of P. Consequently, P has a stable model iff P' has a minimal model M such that $M \models w$. Since P' is positive, $\mathrm{DSM}(P') = \mathrm{MM}(P')$; thus it holds that $\mathrm{DSM}(P') \not\models \neg w$ iff P is DSM-consistent. Since deciding the latter is by Theorem 3 Σ_2^P-hard, the result follows. ∎

Notice that disjunction in the heads of clauses is essential for our proof of this result. If P is nondisjunctive, then DSM-**Entailment** is in coNP (notice that for any I, P^I is a collection of Horn clauses in this case), and by the well-known results for definite programs coNP-complete.

Corollary 6
Let P be a positive DLP and let p be an atom. Deciding whether $\mathrm{MM}(P) \models \neg p$ *is* Π_2^P*-hard.*

Proof. As P is a positive DLP, $\mathrm{DSM}(P) = \mathrm{MM}(P)$. ∎

This result sharpens the result of Lemma 3.1 in [21] that deciding whether a literal $\neg u$ follows from the minimal models of a collection of propositional clauses is Π_2^P-hard. The proof of the cited result involved integrity clauses.

Using these results, we establish the following complexity characterization of disjunctive stable model semantics.

Theorem 7
DSM-**Consistency** *is* Σ_2^P*-complete.*

Proof. By Theorem 3 it remains to show membership in Σ_2^P.

A guess for $M \in \mathrm{DSM}(P)$ can be verified in polynomial time with an NP oracle: Indeed, it is easy to see that P^M is computable in polynomial time. Testing $M \in \mathrm{MM}(P^M)$ is in coNP (cf. [10]) and hence decidable with one query to an NP oracle. Consequently, DSM-**Consistency** is in Σ_2^P. ■

Theorem 8
DSM-**Entailment** *is* Π_2^P*-complete.*

Proof. By Theorem 5 it remains to show membership of the problem in Π_2^P.

A guess for $M \in \mathrm{DSM}(P)$ such that $M \not\models F$ can be verified in polynomial time with an NP oracle (cf. proof of Theorem 7). This means that the complement of DSM-**Entailment** is in Σ_2^P, which implies that DSM-**Entailment** is in Π_2^P. ■

Now let us turn attention to partial stable model semantics. We need to introduce the concept of partial (i.e. 3-valued) Herbrand interpretation for a definition, cf. [49].

A *partial (Herbrand) interpretation* I is a consistent set of literals. $Pos(I)$ (resp. $Neg(I)$) denotes the set of positive (resp. negative) literals in I. The atoms of positive literals in I are considered to be *true* in I, while the atoms of negative literals are considered to be *false* in I. All other atoms are considered to be *undefined* in I. The set of atoms is extended by three distinguished atoms **t**,**f**,**u**, which are respectively true, false, and undefined in every interpretation. Occurrence of these atoms in the head of any program clause is not permitted.[5]

Each interpretation I can be seen as a function $A \to \{0, 0.5, 1\}$, where A is the set of atoms including **t**,**f**, and **u**, such that $I(p) = 0$ (resp. $I(p) = 0.5$, $I(p) = 1$) iff p is false (resp. undefined, true) in I.

The truth value $V_I(F)$ of a propositional formula in I is defined by recursion as follows. If F is an atom, then $V_I(F) = I(F)$; if $F = \neg G$, then $V_I(F) = 1 - V_I(G)$; if $F = F_1 \wedge F_2$, then $V_I(F) = \min(V_I(F_1), V_I(F_2))$; if $F = F_1 \vee F_2$, then $V_I(F) = \max(V_I(F_1), V_I(F_2))$; and if $F = F_1 \leftarrow F_2$, then $V_I(F) = 1$ if $V_I(F_1) \geq V_I(F_2)$ and $V_I(F) = 0$ otherwise. Here the maximum (resp. minimum) of the empty set is defined as 0 (resp. 1).

A partial interpretation I is a *partial model* of formula F iff $V_I(F) = 1$, i.e. I satisfies F. The collection of all partial models of a DLP P is denoted by $M_p(P)$. The partial order $\leq$ is defined on $M_p(P)$ by $M_1 \leq M_2$ iff $Pos(M_1) \subseteq Pos(M_2)$ and $Neg(M_1) \supseteq Neg(M_2)$; $<$ and minimal partial models are defined in the obvious way. $MM_p(P)$ denotes the set of minimal partial models of P.

A partial interpretation I is called *total* if no atom (except **u**) is undefined in I. The

[5] As Przymusinski remarks, this assumption is not essential and could be dropped.

projection of such an I to the nondistinguished atoms corresponds to a standard Herbrand interpretation. For convenience, we will identify I with this standard interpretation and vice versa.

Let P be a DLP and I be a partial interpretation. Przymusinski defines the *quotient* $\frac{P}{I}$ *of* P *modulo* I as follows. Each occurrence of a literal $not\,p$ in P is substituted by its truth value in I where "*not*" is interpreted classically, i.e. if $V_I(not\,p) = 0$ (resp. 0.5, 1), then $not\,p$ is replaced with $\mathbf{f}$ (resp. $\mathbf{u}$, $\mathbf{t}$).

This transformation properly generalizes the Gelfond-Lifschitz transformation; if I is total and $\mathbf{u}$ does not occur in P, then $\frac{P}{I}$ is equivalent to the Gelfond-Lifschitz transformation P^I.

A partial interpretation I of a DLP P is called a *(disjunctive) partial stable model of* P iff $I \in \mathrm{MM}_p(\frac{P}{I})$. Then, define

$$\mathrm{PDSM}(P) = \{I : I \in \mathrm{MM}_p(\tfrac{P}{I})\}.$$

Of course, partial stable models of P are models of P.

Proposition 9 (cf. [49][6])
Every partial stable model of a DLP P is a minimal partial model of P.

Example 5
Let

$$P = \quad a \vee b \vee c \leftarrow \qquad b \leftarrow not\,a \qquad c \leftarrow not\,b \qquad a \leftarrow not\,c$$

and let $I = \{a\}$. It is easily checked that I is a partial model of P. Consider

$$\frac{P}{I} = \quad a \vee b \vee c \leftarrow \qquad b \leftarrow \mathbf{f} \qquad c \leftarrow \mathbf{u} \qquad a \leftarrow \mathbf{u}.$$

I is not a minimal partial model of $\frac{P}{I}$, however, since $M = \{a, \neg b\}$ is a model of $\frac{P}{I}$ and $M < I$. Thus I is not a partial stable model of P. In fact, P has no such model.

Partial stable models of a DLP P can be forced to be total models. Denote by $T(P)$ the program obtained from P by adding for each atom p the clause $\leftarrow p, not\,p$.

Proposition 10
Let P be a DLP in which $\mathbf{u}$ *does not occur.*[7] *Each partial model of $T(P)$ is total, and, moreover, the partial stable models of $T(P)$ are the stable models of P.*

Proof. It is easy to see that each partial model of $T(P)$ is total. Hence the result follows from Propositions 1 and 9. ■

Thus, if integrity clauses are permitted, DSM-**Consistency** and DSM-**Entailment** are polynomially reducible to PDSM-**Consistency** and PDSM-**Entailment**, respectively.

[6] Again, this proposition is formulated only for programs without integrity clauses, but the proof applies to the general case.

[7] Since we defined (total) stable models in a 2-valued context, this assumption is necessary.

Not much surprising, consistency checking and entailment have for partial stable model semantics the same upper bounds as for stable model semantics. The question remains what happens if integrity clauses are excluded. As we will show, this does not lead to a decrease in complexity.

Theorem 11
PDSM-**Consistency** *is* Σ_2^P*-complete.* Σ_2^P*-hardness holds even if* P *does not contain integrity clauses.*

Proof. Membership in Σ_2^P is shown as follows. For any partial interpretation I and any formula F, $V_I(F)$ is efficiently computable and hence $I \models F$ can be decided efficiently. A guess for $M \in \mathrm{PDSM}(P)$ can be verified in polynomial time with an NP oracle for the following: $\frac{P}{M}$ can be computed efficiently. Furthermore, testing whether $M \in \mathrm{MM}_p(\frac{P}{M})$ can be done by using an NP oracle to decide whether there exists no $M' \in M_p(\frac{P}{M})$ such that $M' < M$.

Hardness is shown by a polynomial transformation of deciding whether the DLP P in the proof of Theorem 3 has a (total) stable model. Let P' be the DLP which consists of the following clauses.

(1) $w \leftarrow \mathit{not}\, w$

(2) $w \leftarrow r$ for every atom r distinct from w.

(3) $C' = a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_m, \mathit{not}\, w$ for each clause

$$C = a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_m \quad \text{from } P.$$

We show first that in any partial stable model M of P', w must be undefined and no (nondistinguished) atom can be true. Clearly, w cannot be false in any model of P'. Assume that w is true in M. Then $\frac{P'}{M}$ is equivalent to the program that consists of all clauses $w \leftarrow r$ from P'. M is not a minimal partial model of this program, however. Hence, M is not a partial stable model of P', contradiction. Consequently, w is undefined in M. The clauses $w \leftarrow r$ enforce that no atom r from P can be true in M; hence we are done.

Because $\mathit{not}\, w$ occurs in the body of each rule C', the partial stable models of P' intuitively correspond 1-1 to the stable models of P, where "undefined" (resp. "false") in partial models corresponds to "true" (resp. "false") in total models.

We show that if M is a stable model of P then the partial interpretation $M' = Neg(M)$ (recall that we identify M with the corresponding partial interpretation) is a partial stable model of P'. As M is a stable model of P, $w \in M$. Hence, w is undefined in M'. Notice that $\frac{P'}{M'}$ contains each clause C from P except $w \leftarrow \mathit{not}\, w$, augmented by $\mathbf{u}$ in the body. Since M satisfies C, it follows that M' satisfies C'. Thus, M' is a partial model of $\frac{P'}{M'}$. We show that $M' \in \mathrm{MM}_p(\frac{P'}{M'})$. Assume that there exists a partial model I of $\frac{P'}{M'}$ such that $I < M'$. w must be undefined in I, and hence no atom is true in I. Let J be the total interpretation that satisfies $Neg(J) = Neg(I)$. It follows that J is a model of P^M such that $J < M$. Hence, $M \notin \mathrm{MM}(P^M)$, thus M is not a stable model of P, contradiction. Consequently, $M' \in \mathrm{MM}_p(\frac{P'}{M'})$. It follows that M' is a partial stable model of P'.

In the same way it can be shown that if M is a partial stable model of P', then the total interpretation M' such that $Neg(M') = Neg(M)$ is a stable model of P.

Consequently, P' is PDSM-consistent iff P is DSM-consistent. Since P' is constructible in polynomial time, the result follows. ∎

We conclude this section by considering the entailment problem. The following lemma, which is among the results in [48], relates minimal and stable models of positive DLPs.

Lemma 12
Let P be a positive DLP in which **u** *does not occur. Then all partial stable models are total, and* $\mathrm{PDSM}(P) = \mathrm{DSM}(P) = \mathrm{MM}(P)$.

We use this lemma for the following characterization of inference from partial stable models.

Theorem 13
PDSM-**Entailment** *is* Π_2^P*-complete, and* Π_2^P*-hard even if P is positive and F is a literal.*

Proof. The hardness part follows immediately from Lemma 12 and Corollary 6.

The membership part is as follows. A guess for $M \in \mathrm{PDSM}(P)$ such that $M \not\models F$ can be verified in polynomial time with an NP oracle (cf. the proof of Theorem 11). Consequently, deciding $\mathrm{PDSM}(P) \not\models F$ is in Σ_2^P, from which the result follows. ∎

4 Closed world reasoning and extensions

One of the first and most well-known rules for inferring negative information was Reiter's *Closed World Assumption* (CWA) [52] for positive definite programs. Informally, CWA adds to P each literal $\neg x$ such that $\mathrm{M}(P) \not\models x$. This is not suitable for disjunctive logic programs, since the result of applying CWA may be inconsistent.[8] Several generalizations and extensions of CWA to disjunctive logic programs have been proposed in the literature, e.g. [43, 26, 67, 28], as well as improvements to some of those extensions [55, 51, 56, 13]. Complexity results for these semantics in the case of finite propositional theories have been derived in [60, 11, 21, 13]. In this section, we succinctly introduce various semantics and apply results of the previous section to slightly sharpen known results, and, for the sake of completeness, we collect some easy new results.

Minker adapted the CWA for disjunctive logic programs by introducing the *Generalized CWA* (GCWA) in [43], which adds all literals $\neg x$ to P such that atom x is false in all minimal models of P. The respective models of P can be characterized as follows.

$$\mathrm{GCWA}(P) = \{M \in \mathrm{M}(P) : \forall x \in A(P).\, \mathrm{MM}(P) \models \neg x \Rightarrow M \models \neg x\}$$

The *Careful Closed World Assumption* (CCWA) of Gelfond and Przymusinska [26] generalizes the GCWA as follows. For any partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ of the atoms $A(P)$, each literal

[8] It is interesting to note that deciding whether CWA-**Consistency** is coNP-hard and in $\Delta_2^P[O(\log n)]$, but not in coD^P ($\mathrm{coD}^P \supseteq \mathrm{NP} \cup \mathrm{coNP}$) unless the polynomial hierarchy collapses [21].

$\neg x$, $x \in P$, is added to P such that $\mathrm{MM}(P;\hat{P};\hat{Z}) \models \neg x$. Thus,

$$\mathrm{CCWA}_{\hat{P};\hat{Z}}(P) = \{M \in \mathrm{M}(P) : \forall x \in A(P).\, \mathrm{MM}(P;\hat{P};\hat{Z}) \models \neg x \Rightarrow M \models \neg x\}$$

Notice that if $\hat{P} = A(P)$, then CCWA is identical to GCWA.

The *Extended GCWA* (EGCWA) was introduced by Yahya and Henschen [67] for inferring negative clauses. P is augmented by each clause

$$\leftarrow a_1, \cdots, a_n$$

that is true in every minimal model of P. It holds that

$$\mathrm{EGCWA}(P) = \mathrm{MM}(P).$$

The EGCWA is generalized by the *Extended CWA* (ECWA) of Gelfond et al. [28] in the following way. For any partition $\langle \hat{P};\hat{Q};\hat{Z}\rangle$ of $A(P)$,

$$\mathrm{ECWA}_{\hat{P};\hat{Z}}(P) = \mathrm{MM}(P;\hat{P};\hat{Z}).$$

ECWA reduces to EGCWA in case $Q = Z = \emptyset$. ECWA is equivalent to *circumscription* (CIRC) of McCarthy [41, 42] as defined by Lifschitz in [32]. For $\langle \hat{P};\hat{Q};\hat{Z}\rangle$, define the following formula

$$CIRC(P;\hat{P};\hat{Z}) = P[\hat{P};\hat{Z}] \wedge \neg\exists \hat{P}'\hat{Z}'(P[\hat{P}';\hat{Z}'] \wedge (\hat{P}' < \hat{P}))$$

(cf.[32] for details). Then $\mathrm{CIRC}_{\hat{P};\hat{Z}}(P)$ is the collection of models of $CIRC(P;\hat{P};\hat{Z})$. In case of finite propositional DLPs, it holds that $\mathrm{CIRC}_{\hat{P};\hat{Z}}(P) = \mathrm{MM}(P;\hat{P};\hat{Z})$ [28].

It follows from the definitions that each of these semantics $\mathcal{S}$ is independent of the syntactical representation of a program. Thus it may be assumed that P is *not*-free. (In fact, e.g. GCWA was originally defined for such programs.) Furthermore, it is well-known that for each such $\mathcal{S}$, P is $\mathcal{S}$-consistent iff P is consistent. Consequently,

Proposition 14
If a DLP P contains no integrity clauses, then P is $\mathcal{S}$-consistent for every semantics $\mathcal{S}$ among GCWA, CCWA, EGCWA, ECWA, *and* CIRC.

In this case, $\mathcal{S}$-**Consistency** is trivial. It becomes most likely more complex if integrity clauses are permitted.

Proposition 15
$\mathcal{S}$-**Consistency** *is* NP*-complete for every semantics $\mathcal{S}$ among* GCWA, CCWA, EGCWA, ECWA, *and* CIRC.

The following result has been established previously.

Proposition 16 ([21])
$\mathcal{S}$-**Entailment** *is in* Π_2^P *for $\mathcal{S}$ among* EGCWA, ECWA,[9] *and* CIRC *and in* $\Delta_3^P[O(\log n)]$ *for $\mathcal{S}$ among* GCWA *and* CCWA.

[9] In [11] a proof of membership of ECWA-**Entailment** in Π_2^P is already sketched.

Furthermore, Π_2^P-hardness was shown in [21] as a lower bound of $\mathcal{S}$-**Entailment** for each of these semantics. This improved on previous coNP-hardness and NP-hardness results [60, 11]. From Corollary 6, we obtain the following slight sharpening of this result.

Theorem 17
$\mathcal{S}$-**Entailment** *where P is a positive DLP is Π_2^P-hard for every $\mathcal{S}$ among* GCWA*,* CCWA*,* EGCWA*,* ECWA*, and* CIRC*.*

The GCWA has the undesirable feature that it interprets, if possible, the disjunction $p \vee q \leftarrow$ exclusively, i.e. adopts the models in which exactly one of p and q is true and rejects the inclusive interpretation, i.e. the model in which both p and q are true. Ross and Topor introduced in [55] a semantics called *Disjunctive Database Rule* (DDR), which interprets disjunction inclusively. Independently, Rajasekar et al. developed the Weak GCWA (WGCWA) [51] to cope with the same problem. It turned out that DDR semantics and the WGCWA are equivalent [51].

We need an additional concept for a definition of DDR. Given a DLP P and an interpretation I of P, define

$$T_P(I) \;=\; \{a_1,\ldots,a_n : \; a_1 \vee \cdots \vee a_n \leftarrow b_1,\cdots,b_k \in P \text{ and } b_j \in I, \text{ for all } j=1,\ldots,k\},$$

and

$$T_P\!\uparrow\! 0 = \emptyset, \quad T_P\!\uparrow\! n+1 = T_P(T_P\!\uparrow\! n), \quad \text{and} \quad T_P\!\uparrow\!\omega = \bigcup_{n=0}^{\infty} T_P\!\uparrow\! n.$$

The *Disjunctive Database Rule* (DDR) can be characterized as follows. Assume that P is a *not*-free DLP. Then,

$$\mathrm{DDR}(P) = \{M \in \mathrm{M}(P) : \forall x \in A(P) - T_P\!\uparrow\!\omega.\, M \models \neg x\}.$$

Informally, DDR adds to P all literals $\neg x$ where atom x does not occur in $T\!\uparrow\!\omega$. Notice that, unlike the above semantics, DDR is syntax-dependent.

Example 6
Consider the logically equivalent programs P_1 and P_2:

$$P_1 = \quad a \leftarrow$$
$$P_2 = \quad a \vee b \leftarrow \qquad a \leftarrow \; .$$

Then, $T_{P_1}\!\uparrow\!\omega = \{a\}$ and $T_{P_2}\!\uparrow\!\omega = \{a,b\}$. Hence, $\mathrm{DDR}(P_1) \models \neg b$*, while* $\mathrm{DDR}(P_2) \not\models \neg b$*.*

The complexity of consistency checking is easily determined. Ross and Topor show that P is DDR-consistent if P is consistent. Since the converse is obvious, DDR-**Consistency** and WGCWA-**Consistency** are trivial if P does not contain integrity clauses, as P is consistent in this case. In the general case, we obtain by the NP-hardness of MSAT the following simple result.

Proposition 18
DDR-**Consistency** *and* WGCWA-**Consistency** *are* NP*-complete.*

Next we consider entailment. It is not hard to see that $T_P \uparrow \omega$ can be computed efficiently; hence, the complexity of DDR-**Entailment** and WGCWA-**Entailment** is also easy.

Proposition 19
DDR-**Entailment** *and* WGCWA-**Entailment** *are* coNP*-complete.*

Notice that Chan proved that if F is a literal, then DDR-**Entailment** is coNP-complete in general, but polynomial if no integrity clauses are present [13].
Chan improved the DDR semantics by taking care of negative clauses in P, which are not respected by DDR.

Example 7
Consider the program

$$P = \quad a \vee b \leftarrow \quad \leftarrow a, b \quad c \leftarrow a, b.$$

Then DDR$(P) \not\models \neg c$*, which is not very intuitive.*

Under Chan's *Possible Worlds Semantics* (PWS), however, PWS$(P) \models \neg c$ as suggested by intuition.
Independently, Sakama developed his *Possible Models Semantics* (PMS) [56] for disjunctive programs. Chan showed that PMS and PWS are in fact equivalent.
For space reasons, we provide here only a characterization of the PMS. A *split program* of a DLP P is any program obtained if every clause

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_m, \quad n \geq 2,$$

in P is replaced for some nonempty $N \subseteq \{1, \ldots, n\}$ by the Horn clauses

$$a_p \leftarrow b_1, \cdots, b_m, \quad \text{for all } p \in N.$$

Let $\mathcal{SP}(P)$ be the family of all split programs of P. Let P be a *not*-free DLP. Then,

$$\mathrm{PMS}(P) = \{M \in \mathrm{MM}(P') : P' \in \mathcal{SP}(P)\}.$$

Notice that the PMS coincides with the DDR if P is positive; hence, PMS and PWS are syntax dependent. Chan has shown [13] that if P is consistent, then P is PWS-consistent; the converse follows from the definition. From the NP-hardness of MSAT, we thus obtain the following.

Proposition 20
PWS-**Consistency** *and* PMS-**Consistency** *are* NP*-complete.*

Concerning entailment, Chan has shown [13] that if F is a literal, then PWS-**Entailment** is coNP-complete in general, but is polynomial if integrity clauses are excluded. We obtain the following.

Theorem 21
PWS-**Entailment** *and* PMS-**Entailment** *are* coNP*-complete.*

Proof. It remains to show the membership part, which can be done as follows. A guess for $M \in \mathrm{PMS}(P)$ can be verified in polynomial time as follows.[10] Guess $P' \in \mathcal{SP}(P)$ to verify M. Since P' is a collection of Horn clauses, P' has in case of consistency a unique minimal model M', which is efficiently computable. If $M' = M$, then $M \in \mathrm{PMS}(P)$ holds. Thus, deciding $\mathrm{PMS}(P) \not\models F$ is in NP, which implies membership of the problems in coNP. ■

5 Stratified disjunctive programs

The concept of stratification, which had been discussed by Chandra and Harel [14], was introduced for logic programs independently by Apt, Blair, and Walker [2] and by van Gelder [64]; Przymusinski generalized it to DLPs (without integrity clauses) [48]. A DLP P without integrity clauses is *stratified* iff it is possible to partition the atoms into strata $\langle S_1, \ldots, S_r \rangle$, such that for every clause

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k, \mathit{not}\, c_1, \cdots, \mathit{not}\, c_m$$

in P there exists a constant c, $1 \leq c \leq r$, such that

$$\begin{aligned} &\mathrm{Stratum}(a_i) = c, \text{ for all } 1 \leq i \leq n, \\ &\mathrm{Stratum}(b_j) \leq c, \text{ for all } 1 \leq j \leq k, \quad \text{and} \\ &\mathrm{Stratum}(c_l) < c, \text{ for all } 1 \leq l \leq m, \end{aligned}$$

where $\mathrm{Stratum}(x) = i$ iff $x \in S_i$. Any such $\langle S_1, \ldots, S_r \rangle$ is a *stratification* of P.[11] It is well-known that a stratification of a DLP P can be efficiently found. In particular, every positive DLP is stratified by choosing $S = \langle A(P) \rangle$.

Gelfond et al. [28] defined the *Iterated CWA* (ICWA) as iterated application of ECWA to a stratified DLP. Let $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ be a partition of $A(P)$ and $S = \langle S_1, \ldots, S_r \rangle$ a stratification of P, and let P_i be the clauses from P that contain only atoms from S_j, $j \leq i$ in their heads . Furthermore, let $\hat{P}_i = \hat{P} \cap S_i$, $\hat{Z}_i = \hat{Z} \cap (S_1 \cup \cdots \cup S_i)$.

The ICWA of P can be characterized as follows (cf. [28]).

$$\begin{aligned} &\mathrm{ICWA}_{\hat{P}_1;\hat{Z}_1}(P_1) = \mathrm{ECWA}_{\hat{P}_1;\hat{Z}_1}(P_1), \\ &\mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_{n+1};\hat{Z}_{n+1}}(P_{n+1}) = \\ &\qquad \mathrm{ECWA}_{\hat{P}_{n+1};\hat{Z}_{n+1}}(P_{n+1} \cup \mathcal{F}(\mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_n;\hat{Z}_n}(P_n))), \quad n > 0, \\ &\mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_r;\hat{Z}}(P) = \mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_r;\hat{Z}_r}(P_r), \end{aligned}$$

where $\mathcal{F}(\mathcal{M})$ is some DLP P' such that $\mathrm{M}(P') = \mathcal{M}$.

[10] Chan follows a different proof.

[11] It is not clear how to extend the concept of stratification to programs with integrity clauses in a reasonable way. Hence, we do not consider such an extension.

Stratifiability of a program assures consistency; this is maintained for ICWA under arbitrary partitions of $A(P)$.

Proposition 22 ([28])
Let P be a DLP P (without integrity clauses) stratified by S. Then, for any partition $\pi = \langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ of $A(P)$, P is ICWA*-consistent with respect to S and π.*

Thus ICWA-**Consistency** is trivial.[12] Now let us consider entailment of a formula. Recall that a positive DLP P is stratified by taking $S = \langle A(P) \rangle$. Thus for any partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$, we have $\mathrm{ICWA}_{\hat{P};\hat{Z}}(P) = \mathrm{ECWA}_{\hat{P};\hat{Z}}(P)$. From Theorem 17 we hence obtain the following.

Theorem 23
ICWA-**Entailment**, *given a stratification S and a partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ of $A(P)$, is Π_2^P-hard, even if P is positive and F is a literal.*

By results in [28, Section 6], the models under ICWA can be described by the intersection of certain applications of ECWA.

Lemma 24 ([28])
For any DLP P stratified by S and any partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$,

$$\mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_r; \hat{Z}}(P) = \bigcap_{i=1}^{r} \mathrm{ECWA}_{\hat{P}_i; \hat{P}_{i+1} \cup \cdots \cup \hat{P}_r \cup \hat{Z}}(P).$$

The next lemma, which is implicitly used in [21], immediately follows from the results on circumscription in [10] and the equivalence of CIRC and ECWA in the propositional case.

Lemma 25
Let P be a DLP, let $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ be a partition of $A(P)$, and let $M \in \mathrm{M}(P)$. Deciding whether $M \in \mathrm{ECWA}_{\hat{P};\hat{Z}}(P)$ is in coNP.

We thus obtain the following result.

Theorem 26
ICWA-**Entailment**, *given a stratification $S = \langle S_1, \ldots, S_r \rangle$ and a partition $\langle \hat{P}; \hat{Q}; \hat{Z} \rangle$ of $A(P)$, is Π_2^P-complete.*

Proof. By Theorem 23 it remains to show the membership part.
A guess for a model $M \in \mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_r; \hat{Z}}(P)$ such that $M \not\models F$ can be verified in polynomial time with an NP oracle: Indeed, from Lemma 24 $M \in \mathrm{ICWA}_{\hat{P}_1 > \cdots > \hat{P}_r; \hat{Z}}(P)$ iff $M \in \mathrm{ECWA}_{\hat{P}_i; \hat{P}_{i+1} \cup \cdots \cup \hat{P}_r \cup \hat{Z}}(P)$, for all $1 \leq i \leq r$. Each of these membership tests is by Lemma 25 possible with a query to an NP oracle. Hence, membership of ICWA-

[12] However, notice that integrity clauses are not permitted, and that $\mathcal{S}$-**Consistency** is trivial for many other semantics in this case, too.

Entailment in Π_2^P follows. ■

6 Perfect model semantics

Perfect model semantics was proposed by Przymusinski in [48] to capture the meaning of normal logic programs which are not stratifiable. It involves a preference relation among models based on the structure of the clauses which is, like ICWA, in the spirit of minimal model semantics.

For a DLP P without integrity clauses, the priority relation $<$ on atoms [48] is defined using an auxiliary relation $\preceq$ as follows. For each clause

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k, not\, c_1, \cdots, not\, c_m$$

from P, it holds that

(i) $a_i < c_j$, for all $1 \leq i \leq n$, $1 \leq j \leq m$,

(ii) $a_i \preceq b_j$, for all $1 \leq i \leq n$, $1 \leq j \leq k$, and

(iii) $a_i \preceq a_j$ for all $1 \leq i, j \leq n$.

Now $<$ and $\preceq$ are the smallest extensions to (i)–(iii) which satisfy $a < b \Rightarrow a \preceq b$ and which are closed under transitivity, i.e. $a \preceq b, b \preceq c \Rightarrow a \preceq c$ and $a \preceq b$, $b < c$ (resp. $d < a$) $\Rightarrow a < c$ (resp. $d < b$). Intuitively, $x < y$ means that y has higher priority than x.

We naturally extend this definition to general DLPs, in which integrity clauses are simply ignored for defining $\preceq$ and $<$. Notice that this generalization has no effect on the complexity of the considered problems in the general case.

Definition 27
The preference order $\ll$ *on* $\mathrm{M}(P)$ *is defined by* $M_1 \ll M_2$ *iff* $M_1 \neq M_2$ *and for each* $x \in M_1 - M_2$ *there exists* $y \in M_2 - M_1$ *such that* $x < y$.

Notice that $M_1 < M_2 \Rightarrow M_1 \ll M_2$. It is not hard to find an algorithm that, given a pair of atoms x, y, decides $x < y$ efficiently. Consequently,

Proposition 28
Given P *and* $M_1, M_2 \in \mathrm{M}(P)$, $M_1 \ll M_2$ *is efficiently decidable.*

A model M of P such that no $M' \in \mathrm{M}(P)$ is preferred over M, i.e. $M' \ll M$, is called *perfect*. The perfect model semantics is defined as follows.

$$\mathrm{PERF}(P) = \{M \in \mathrm{M}(P) : \forall M' \in \mathrm{M}(P).\, M' \not\ll M\}.$$

Przymusinski has shown the following properties.

Proposition 29 ([48])
PERF$(P) \subseteq$ MM(P), *and* $M \in$ MM(P) *is perfect iff there exists no* $M' \in$ MM(P) *such that* $M' \ll M$.

Notice that the program

$$P = \quad p \leftarrow not\, p$$

which is not stratifiable, has the perfect model $\{p\}$. However, not every consistent (and hence MM-consistent) program P has a perfect model.

Example 8
The program

$$P = \quad q \leftarrow \neg p \quad p \leftarrow \neg q$$

has no perfect model: MM$(P) = \{\{p\},\{q\}\}$ *and* $\{p\} \ll \{q\}$, $\{q\} \ll \{p\}$.

To our best knowledge, no complexity results for PERF-**Consistency and** PERF-**Entailment** have been derived so far, for the general case as well as for the restrictions to definite and nondisjunctive programs.

The following result provides a lower bound for the problem of consistency checking in the general case.

Theorem 30
PERF-**Consistency** *is* Σ_2^P*-hard, even if* P *does not contain integrity clauses.*

Proof. We show this by a reduction of deciding the validity of a quantified Boolean formula

$$\Phi = \exists x_1 \cdots \exists x_n \forall y_1 \cdots \forall y_m E, \quad n, m \geq 1,$$

where $E = D_1 \vee \cdots \vee D_r$ and each $D_i = L_{i,1} \wedge L_{i,2} \wedge L_{i,3}$ is a conjunction of literals $L_{i,j}$ over atoms $x_1, \ldots, x_n$, $y_1, \ldots, y_m$. The reduction is similar to the one in the proof of Theorem 3. Let $v_1, \ldots, v_n$ and $z_1, \ldots, z_m, w, w'$ be new propositional atoms and define the following DLP P:

$$\begin{array}{llll}
x_i \vee v_i \leftarrow & & & \text{for each } i = 1, \ldots, n \\
y_j \vee z_j \leftarrow & y_j \leftarrow w, w' & z_j \leftarrow w, w' & \\
w \leftarrow y_j, z_j & w' \leftarrow y_j, z_j & & \text{for each } j = 1, \ldots, m \\
\multicolumn{3}{l}{w \vee w' \leftarrow \sigma(L_{k,1}), \sigma(L_{k,2}), \sigma(L_{k,3})} & \text{for each } k = 1, \ldots, r \\
w \leftarrow not\, w' & w' \leftarrow not\, w & &
\end{array}$$

where σ is as in the proof of Theorem 3, i.e. it maps literals from atoms $x_1, \ldots, x_n$, $y_1, \ldots, y_m$ to atoms as follows:

$$\sigma(L) = \begin{cases} v_i & \text{if } L = \neg x_i \text{ for some } i = 1, \ldots, n \\ z_j & \text{if } L = \neg y_j \text{ for some } j = 1, \ldots, m \\ L & \text{otherwise} \end{cases}$$

Again, v_i intuitively corresponds to $\neg x_i$ and z_j to $\neg y_j$.

We make the following observations (1)–(3). (1) For any model M of P, if $M \models y_j \wedge z_j$ for some $1 \leq j \leq m$ or $M \models w \wedge w'$, then $M \models w \wedge w'$ and $M \models y_j \wedge z_j$ for all $1 \leq j \leq m$. (2) If $M \in \mathrm{MM}(P)$, then M satisfies exactly one of x_i and v_i, for every $1 \leq i \leq n$. (3) For every $1 \leq i \leq n$, there exists no atom p such that $x_i < p$ or $v_i < p$, and for each atom q from $\{w, w', y_1, z_1, \ldots, y_m, z_m\}$, $q < w$ and $q < w'$.

We show that P has a perfect model iff the formula Φ is valid.

"$\Rightarrow$": Let M be a perfect model of P. Then, since $M \in \mathrm{MM}(P)$, from (2) M satisfies exactly one of x_i and v_i, for every $1 \leq i \leq n$. It holds that $M \models w \wedge w'$. For if not, then assume, since $M \models w \vee w'$, that $M \models w$. The symmetry between w and w' in P implies that $M' = M - \{w\} \cup \{w'\}$ is another minimal model of P. This model satisfies $M' \ll M$, hence M is not a perfect model of P, which is a contradiction. The case $M \models w'$ is analogous. Thus $M \models w \wedge w'$, and hence $y_j, z_j \in M$, for each $1 \leq j \leq m$. Let the truth value assignment φ to $x_1, \ldots, x_n$ be defined by

$$\varphi(x_i) = \begin{cases} \mathit{true} & \text{if } x_i \in M \\ \mathit{false} & \text{if } v_i \in M \end{cases}, \quad \text{for } i = 1, \ldots, n.$$

Since M is a perfect (and hence a minimal) model of P, for every extension of φ to $y_1, \ldots, y_m$, there must exist a k from $1, \ldots, r$ such that φ satisfies $L_{k,j}$, for each $1 \leq j \leq 3$. For otherwise, there would exist a model M' of P such that $M' < M$ and hence $M' \ll M$. (M' is straightforwardly constructed from φ and assigns w and w' false.) This M' invokes a contradiction that M is a perfect model of P. Consequently, Φ is valid.

"$\Leftarrow$": Suppose Φ is valid. Let φ be a truth value assignment to $x_1, \ldots, x_n$ such that every extension of φ to $y_1, \ldots, y_m$ satisfies E. Let I be the following interpretation:

$$\begin{aligned} I \;=\; & \{x_i : \varphi(x_i) = \mathit{true}, 1 \leq i \leq n\} \cup \{v_i : \varphi(x_i) = \mathit{false}, 1 \leq i \leq n\} \cup \\ & \{y_1, z_1, \ldots, y_m, z_m, w, w'\} \end{aligned}$$

Then, I is a model of P, and in fact a minimal model of P. Let M be any minimal model of P distinct from I. (2) implies that there must exist an i from $1, \ldots, n$ such that $x_i \in M - I$ or $v_i \in M - I$. From (3) it follows that $M \not\ll I$. Thus from Proposition 29 it follows that I is a perfect model of P.

Since P can be constructed in polynomial time, the result follows. ■

We obtain the following complexity characterization of PERF-**Consistency**.

Theorem 31
PERF-**Consistency** *is* Σ_2^P*-complete.*

Proof. By Theorem 30 it remains to show membership in this class.

A guess M for a perfect model of P can be verified with an NP oracle in polynomial time. Indeed, deciding whether there exists no $M' \in \mathrm{M}(P)$ such that $M' \ll M$ is in coNP as deciding $\ll$ is polynomial. ■

An interesting issue is the complexity of PERF-**Consistency** for nondisjunctive DLPs. Recall that it is NP-complete to decide whether a nondisjunctive program (with or without integrity clauses) has a stable model. If integrity clauses were permitted, then, under the above definition of $<$, PERF-**Consistency** would be still Σ_2^P-hard. (The program P in the proof of Theorem 30 can be easily rewritten such that the transformation works in this case.) However, the complexity of this problem is unclear if integrity clauses are not permitted. The problem remains intractable (coNP-hard), but it is questionable whether it is still Σ_2^P-hard. We provide a result that gives some evidence that for such programs the problem is not in NP $\cup$ coNP.

We show this by a reduction from UMINSAT, which is the problem to decide if a collection $\mathcal{C} = \{C_1, \ldots, C_n\}$ of propositional clauses has a unique minimal satisfying truth assignment, i.e. whether a logically equivalent DLP P has a unique minimal model. The following has been shown in [21].

Proposition 32
UMINSAT *is* coNP*-hard and, unless the polynomial hierarchy collapses,* UMINSAT *does not belong to* coD^P*.*

coD^P is a complexity class that contains –most likely properly– NP $\cup$ coNP [15]. We note the following lemma.

Lemma 33
UMINSAT *is polynomially transformable to deciding whether a definite DLP has a unique minimal model.*

Proof. Let a, b, and c be new atoms not occurring in $\mathcal{C}$. Let P be a DLP logically equivalent to $\mathcal{C} \cup \{\neg c\}$. Clearly, P has a unique minimal model iff $\mathcal{C}$ has a unique minimal satisfying truth assignment. Let the DLP P' contain the following clauses:

(1) $a \leftarrow \mathit{not}\, b, c$

(2) $x \leftarrow c$ for each atom x occurring in $\mathcal{C}$, and

(3) $c \leftarrow \neg C_i$ for each $C_i \in \mathcal{C}$,

where $\neg C_i$ is the conjunction of the opposites of all literals in C_i. Notice that

$$\begin{aligned} \mathrm{M}(P') \quad = \quad & \{I,\, I \cup \{a\},\, I \cup \{b\},\, I \cup \{a, b\} : I \in \mathrm{M}(P)\} \cup \\ & \{A(P) \cup \{a\},\, A(P) \cup \{b\},\, A(P) \cup \{a, b\}\}. \end{aligned}$$

Thus it follows that P' has a unique minimal model iff P has a unique model, and the lemma follows. ■

Theorem 34
PERF*-consistency for definite DLPs is* coNP*-hard and, unless the polynomial hierarchy collapses, not in* NP $\cup$ coNP*.*

Proof. Let P be a DLP as in Lemma 33. Let p be a new atom not occurring in P, and let P' be the DLP obtained from P by adding for each pair of atoms q, r that occur in P the clause

$$q \leftarrow \mathit{not}\, r, p.$$

This clause assures that $q < r$ holds in P'. Notice that no minimal model of P' contains p. Thus, for any distinct minimal models M and M' of P', $M \ll M'$ and $M' \ll M$ holds. Hence it follows from Proposition 29 that P' has a perfect model iff P' has a unique minimal model. It holds that P' has a unique minimal model iff P has a unique minimal model. Consequently, P' has a perfect model iff P has a unique minimal model. Thus the result follows. ■

Now let us consider the complexity of entailment. For the lower bound of PERF-**Entailment**, we make use of the following property.

Proposition 35 ([48])
If P is a positive DLP, then $\mathrm{PERF}(P) = \mathrm{MM}(P)$.

Theorem 36
PERF-**Entailment** *is* Π_2^P*-complete.* Π_2^P*-hardness holds even if P is positive and F is a literal.*

Proof. Membership in Π_2^P holds as a guess for $M \in \mathrm{PERF}(P)$ such that $M \not\models F$ can be verified in polynomial time with an NP oracle (cf. proof of Theorem 30). Hardness for Π_2^P under the asserted restriction follows from Corollary 6 and Proposition 35. ■

As for PERF-**Consistency**, it is open whether the complexity of PERF-**Entailment** is for definite DLPs the same as in the general case. (If P is nondisjunctive, the problem remains Π_2^P-hard.) The problem is trivially coNP-hard if F is an arbitrary formula; whether this applies to the case where F is a literal remains unclear. From Theorem 34, we obtain the following interesting lower bound.

Theorem 37
PERF-**Entailment** *where P is definite and F is a literal is* NP*-hard and, unless the polynomial hierarchy collapses, not in* $\mathrm{NP} \cup \mathrm{coNP}$.

Proof. Let P be a definite DLP P and p an atom not occurring in P. Then, $\mathrm{PERF}(P \cup \{p \leftarrow\}) \models \neg p$ iff P is not PERF-consistent. Since deciding the latter is coNP-hard and not in $\mathrm{NP} \cup \mathrm{coNP}$ by Theorem 34, the result follows. ■

7 Classical negation

Gelfond and Lifschitz [25] pointed out that traditional logic programming does not allow to deal directly with incomplete information, which is a shortcoming for convenient knowledge representation. In order to overcome this limitation, they introduced extended definite logic programs, which permit classical negation besides negation as default. They defined the answer set semantics for such programs, which is in the spirit of stable model semantics. Moreover, they considered also the generalization of extended definite logic programs by allowing for disjunction in the rule heads, and suitably generalized the answer set semantics for the resulting extended disjunctive logic programs (EDLPs). Przymusinski proposed in [49] an extension of DLPs which is basically equivalent to EDLPs with answer set semantics.

Gelfond and Lifschitz emphasized that the answer set semantics of extended definite programs can be equivalently described by a reduction of such programs into a fixpoint nonmonotonic formalism, and described such a reduction to Reiter's default logic [53]. The view of extended definite programs as default theories led Ben-Eliyahu and Dechter to apply techniques developed for answering queries on default theories to EDLPs [4]. They showed that answer set semantics for the large class of headcycle-free EDLPs can be efficiently expressed in propositional logic in polynomial time. Generalizing the results of Marek and Truszczyński [38, 39] and Bidoit and Froidevaux [6], they obtained that for this class the problems of deciding whether an answer set exists and whether a set of literals occurs in any (resp. every) answer set is efficiently reducible to deciding satisfiability (resp. provability) of a propositional formula. A similar efficient reduction for the class of all propositional EDLPs was left as an open issue in [4], which has been resolved (unless the polynomial hierarchy collapses at its first level) in [20].

An *extended disjunctive logic program* (EDLP) is a collection of rules of the form

$$L_1|\cdots|L_k \leftarrow L_{k+1},\ldots,L_m, not\, L_{m+1},\ldots, not\, L_n,$$

where $n \geq m \geq k \geq 0$ and each L_i is a literal under classical negation "$\neg$", and "*not*" is a negation-by-default operator. The symbol "$|$" is used to distinguish the nonstandard "effective" disjunction in the head of a rule from standard logical disjunction. A rule with empty head ($k = 0$) is called *integrity rule*. An EDLP P in which "*not*" does not occur is called *not-free*. We consider here finite propositional EDLPs in which all atoms are propositions, and omit the phrase "finite propositional" in the sequel.

Answer sets of EDLPs are defined as follows. Let a *context* [4] be any subset of Lit, the set of literals under $\neg$ from the atoms of P. Let P be a *not*-free EDLP. Call a context S *closed under* P [40] iff for each rule $L_1|\cdots|L_k \leftarrow L_{k+1},\ldots,L_m$ in P, if $L_{k+1},\ldots,L_m \in S$, then for some $i = 1,\ldots,k$, $L_i \in S$. An *answer set* of P is any in terms of $\subseteq$ minimal context S such that (1) S is closed under P and (2) if S is inconsistent, then $S = Lit$.

An answer set of a general EDLP P is defined as follows. Let the *reduct of* P *with respect to context* S (denoted by $Red(P,S)$) be the EDLP obtained from P by deleting

(i) each rule that has $not\, L$ in its body for some $L \in S$, and

(ii) all subformulae of the form $not\, L$ of the bodies of the remaining rules.

Any context S which is an answer set of $Red(P,S)$ is an *answer set* of P. By ANSW(P) we denote the collection of all consistent answer sets of an EDLP P. An EDLP P is ANSW-*consistent* iff ANSW$(P) \neq \emptyset$, and P *entails* a propositional formula F ($P \models F$) iff $F \in Cn(S)$ for each $S \in$ ANSW(S).

Example 9
Consider the following EDLP borrowed from [49], which states that everyone is pronounced not guilty unless proven otherwise:

$$\begin{array}{rcl} innocent \,|\, guilty & \leftarrow & charged \\ \neg guilty & \leftarrow & not\, proven \\ charged & \leftarrow & \end{array}$$

P has the single answer set $\{\neg guilty, innocent, charged\}$. Neither proven nor $\neg$proven appears in the answer set, which informally means that nothing is known about proven, and both not proven, not ($\neg$proven) are assumed by default.

The problems ANSW-**Consistency** and ANSW-**Entailment** are defined analogous to the problems $\mathcal{S}$-**Consistency** and $\mathcal{S}$-**Entailment** for standard DLPs. Notice that the complexity of these problems for general EDLPs, the latter stated for instances where F is a conjunction of literals, was left open in [4].

We provide the solution using the correspondence between stable models of disjunctive logic programs and answer sets of EDLPs. Let for every DLP clause

$$C = a_1 \vee \cdots \vee a_n \leftarrow b_1, \cdots, b_k, not\, b_{k+1}, \cdots, not\, b_m$$

be $e(C)$ the rule

$$a_1 | \cdots | a_n \leftarrow b_1, \ldots, b_k, not\, b_{k+1}, \ldots, not\, b_m,$$

and denote by $e(P)$ the collection of all $e(C)$ for $C \in P$.

Lemma 38
Let P be a DLP and M be an interpretation. Then, M is a stable model of P iff M is a consistent answer set of $e(P)$. [13]

Proof. Notice that for any interpretation I, $e(P^I) = Red(e(P), I)$, and that each answer set of $e(P)$ contains only positive literals, i.e. corresponds to some interpretation. Since negation does not occur in P^I resp. $Red(e(P), I)$, it follows for every interpretation $J \subseteq I$, that J is closed under $Red(e(P), I)$ if J is a model of P^I and vice versa. Consequently, J is a minimal model of P^I iff J is a minimal context which is closed under $Red(e(P), I)$. Hence it follows that I is a stable model of P iff I is a consistent answer set of $e(P)$. ■

Theorem 39
ANSW-**Consistency** *is Σ_2^P-complete. Σ_2^P-hardness holds even if "$\neg$" does not occur in P and P contains no integrity clauses.*

Proof. Membership in Σ_2^P is shown as follows. A guess S for a consistent answer set S of P can be verified in polynomial time with an NP oracle: $Red(P, S)$ is efficiently computable, and deciding whether $S' \subset S$ exists such that S' is closed under $Red(P, S)$ is possible with a call to the NP oracle. Hardness for Σ_2^P under the asserted restriction follows from Lemma 38 and Theorem 3. ■

[13] M is seen as a subset of the Herbrand base, i.e. as a set of positive literals.

Theorem 40
ANSW-**Entailment** *is* Π_2^P*-complete.* Π_2^P*-hardness holds even if "*$\neg$*" does not occur in* P*,* P *contains no integrity rule, and* F *is a literal.*

Proof. Membership in Π_2^P follows since a guess S for a consistent answer set S of P such that $S \not\models F$ can be verified in polynomial time with an NP oracle (cf. proof of Theorem 39). Hardness for Π_2^P under the asserted restriction can be easily derived from Theorem 39. Let P' be the program resulting from a "$\neg$"-free program P that contains no integrity clauses by adding the clause $p \vee q \leftarrow$, where p and q are new atoms. Then, clearly $\mathrm{ANSW}(P') \models p$ iff $\mathrm{ANSW}(P) = \emptyset$; hence, the result follows. ■

We conclude this section with a remark on implications of our results to a recent extension of logic programming. Marek, Rajasekar, and Truszczyński propose in [37] an extension of EDLPs in which the atomic formulae L_i in a rule are from a class $\mathcal{F}$ of propositional formulae (which includes all atoms) instead from *Lit*, and define the concept of answer set (in their terms stable answer set) accordingly. They report that deciding whether a program P has a stable answer set is in Σ_2^P, and that deciding whether every stable answer set of P entails a formula $F \in \mathcal{F}$ is in Π_2^P. They conjecture that the problems are Σ_2^P-complete resp. Π_2^P-complete if $\mathcal{F}$ consists of all atoms only. Their framework, however, reduces in this case to EDLPs and answer sets as above. Thus from Theorem 39 this conjecture is easily proved.

8 Application to Nonmonotonic Logics

It is well-known that logic programming is closely related to various forms of nonmonotonic reasoning, and that logic programs can be equivalently expressed in nonmonotonic formalisms by use of transformations, cf. [46]. We exploit this relation to obtain new complexity results for nonmonotonic logics.

In particular, we consider in this section applications of the above complexity results for disjunctive logic programming to disjunctive default logic [27] and autoepistemic logic [45, 38]. We assume that the reader is familiar with Reiter's default logic [53] and Moore's autoepistemic logic [45].

8.1 Disjunctive default logic

Disjunctive default theory has been proposed in [27] as a generalization to default logic to overcome difficulties of default logic in handling disjunctive information. A *disjunctive default* is a rule of the form

$$\frac{\alpha : \beta_1, \ldots, \beta_m}{\gamma_1 | \cdots | \gamma_n},$$

where α, $\beta_1, \ldots, \beta_m$, $\gamma_1, \ldots, \gamma_n$, $(n, m \geq 0)$ are quantifier-free first-order formulae. α is the *prerequisite*, the β_i are the *justifications*, and the γ_j are the *consequents*. The familiar default rule of Reiter's formalism is obtained for $n = 1$. A disjunctive default rule allows to effectively conclude one of the γ_i's (rather than their disjunction) if α is derivable and $\neg\beta_1, \ldots, \neg\beta_m$ are not derivable. The symbol "|" denotes as in EDLPs effective disjunction.

A *disjunctive default theory* is a set of disjunctive defaults. The semantics of a disjunctive default theory is defined in terms of its *extensions*, which are deductively closed sets of formulae defined analogous to extensions of Reiter's default logic (cf. [27] for details). We consider in the following finite disjunctive default theories over a propositional language.

The main reasoning tasks for disjunctive default logic are the following. Given a disjunctive default theory D, decide (i) whether D has an extension; (ii) whether a given formula φ belongs to some extension of D; and (iii) whether a given formula φ belongs to every extension of D.

By very recent results in [37], (i) and (ii) are in Σ_2^P and (iii) is in Π_2^P. Notice that the same upper bounds have been established for the corresponding problems in Reiter's default logic [29, 62].

Since disjunctive default logic properly generalizes Reiter's default logic, it follows from the results in [29, 62] that (i) and (ii) are Σ_2^P-hard and that (iii) is Π_2^P-hard if arbitrary propositional formulae are permitted to appear in the defaults.

Our results on EDLPs allow to sharpen these lower bounds considerably, namely to disjunctive default theories where α and all β_i and γ_j are conjunctions of literals. Notice that for classical default logic, under this restriction (i) and (ii) are in NP and (iii) is in coNP (this can be easily shown from the constructive fixed-point characterization of extensions). Moreover, by the results in [31], the problems are also hard and hence complete for the respective classes.

As shown in [27], a propositional EDLP P can be transformed into an equivalent disjunctive default theory $emb_D(P)$ by replacing every rule

$$L_1|\cdots|L_k \leftarrow L_{k+1},\ldots,L_m, not\, L_{m+1},\ldots, not\, L_n,$$

with the disjunctive default

$$\frac{L_{k+1}\wedge\cdots\wedge L_m : \overline{L}_{m+1},\ldots,\overline{L}_n}{L_1|\cdots|L_k}, \tag{1}$$

where $\overline{L}$ is the opposite literal of L.

Proposition 41 ([27, Theorem 7.2])
Let P be a propositional EDLP. Then S is an answer set of P iff S is the set of literals from an extension of $emb_D(P)$.

Applying this embedding and our results on EDLPs in the previous section, we arrive at the following sharpening of the lower bounds for the main reasoning tasks in disjunctive default logic.

Theorem 42
Let D be a finite propositional disjunctive default theory such that each prerequisite α and all justifications β_i and consequents γ_j occurring in D are conjunctions of literals. Then, deciding (i) whether D has an extension is Σ_2^P-hard; (ii) whether a given literal L belongs to some extension of D is Σ_2^P-hard; and (iii) whether a given literal L occurs in every extension of D is Π_2^P-hard.

Proof. Let P be an EDLP in which "$\neg$" does not occur and which does not contain any integrity rule. Each answer set of P must be consistent. Hence from Proposition 41 each extension of $emb_D(P)$ is consistent in this case. Thus (i) follows from Theorem 39. To show (ii), add the clause $p \leftarrow$ to P, where p is a new atom not occurring in P, and let L be p; hence (ii) follows from Theorem 39. (iii) follows from Theorem 40. ■

As seen from the proof, the result of Theorem 42 remains valid if all justifications and consequents occurring in D must be literals and extensions must be consistent, i.e. "extension" is replaced by "consistent extension". Notice that Theorem 42 extends to completeness results.

8.2 Autoepistemic logic

Recall that E is an autoepistemic expansion [45] of an autoepistemic theory T, i.e. a set of formulae from a modal language with modal operator L, iff

$$E = Cn(T \cup \{L\varphi : \varphi \in E\} \cup \{\neg L\varphi : \varphi \notin E\})$$

[49] offers a transformation by which any DLP P without integrity clauses can be embedded into an equivalent autoepistemic theory $emb_A(P)$. For every clause

$$a_1 \vee \cdots \vee a_n \leftarrow b_1, \ldots, b_k, \mathit{not}\, b_{k+1}, \ldots, \mathit{not}\, b_m$$

of P, the formula

$$b_1 \wedge \cdots \wedge b_k \wedge \neg L b_{k+1} \wedge \cdots \wedge \neg L b_m \Rightarrow a_1 \vee \cdots \vee a_n, \tag{2}$$

is in $emb_A(P)$, and for every propositional atom p the formula

$$p \Rightarrow Lp$$

is in $emb_A(P)$, which contains nothing else. It is easy to see that every stable set (cf. [38] for a definition) which contains $emb_A(P)$, and hence every stable expansion of $emb_A(P)$, contains p or $\neg p$ for every atom p.

Przymusinski has shown the following property of $emb_A(P)$.

Proposition 43 ([49])
Let P be a DLP without integrity clauses. There is a 1-1 correspondence between the stable models of P and the stable expansions of $emb_A(P)$, such that atom p is true in a stable model iff p belongs to the corresponding stable expansion.

The main reasoning tasks in autoepistemic logic are the following. Given an autoepistemic theory T, decide (i) whether T has an expansion; (ii) whether a given formula φ belongs to some expansion of T; and (iii) whether φ belongs to every expansion of T. Niemelä [47] showed that in case of a finite propositional T, (i) and (ii) are in Σ_2^P and (iii) is in Π_2^P. Gottlob complemented these upper bounds with respective hardness results [29]; his proof, however, required that complex formulae can occur in T.

Our results on stable model semantics lead by virtue of the embedding $emb_A(\cdot)$ to the following noticeable sharpening of these hardness results. Call an autoepistemic theory a *disjunctive autoepistemic literal theory* (DALT) if it contains only disjunctions $D_1 \vee \cdots \vee D_n$, where each D_i is a modal literal LB or $\neg LB$, where B is a literal, or D_i is a literal.

Theorem 44
Given a finite propositional DALT T, deciding (i) whether T has a stable expansion is Σ_2^P-hard; (ii) whether a given atom p belongs to some stable expansion of T is Σ_2^P-hard; and (iii) whether a given atom p belongs to every stable expansion of T is Π_2^P-hard.

Proof. Let P be a DLP without integrity clauses. Notice that each stable expansion of $emb_A(P)$ is consistent. We apply Theorem 3 and Proposition 43. (i) follows immediately. To show (ii) and (iii), let p be a new atom which does not occur in P; p belongs to some stable expansion of $emb_A(P \cup \{p \leftarrow\})$ iff P has a stable model, which proves (ii). Further, p belongs to every stable expansion of $emb_A(P \cup \{p \leftarrow p\})$ iff P has no stable model, which proves (iii). ■

As seen from the proof, Theorem 44 holds also if only consistent stable expansions are permitted, i.e. "expansion" is replaced by "consistent expansion". By the results in [47], the hardness results extend to completeness results.

To conclude this section, we mention that the results of this paper have further applications to variants of standard autoepistemic logic. For example, the above embedding $emb_A(\cdot)$ also can be applied to moderately grounded expansions and parsimonious stable expansions [19]. Notice that based on the results of [19], Schaerf was able to derive similar complexity results for these variants of autoepistemic logic [57]. Furthermore, complexity results for restricted fragments of reflexive autoepistemic logic [61] and 3-valued autoepistemic logic [50] can be obtained by embeddings of EDLPs into reflexive autoepistemic logic [33, 40] and of DLPs into 3-valued autoepistemic logic [49].

9 Conclusion

In this paper we have analyzed the computational cost of important problems for disjunctive logic programming in the case of finite propositional programs. In particular, the complexity of deciding whether a disjunctive logic program (DLP) has a model under semantics $\mathcal{S}$ ($\mathcal{S}$-**Consistency**) and the complexity of deciding whether a propositional formula is a consequence of all models of a DLP under semantics $\mathcal{S}$ ($\mathcal{S}$-**Entailment**) have been studied for various well-known semantics $\mathcal{S}$ of DLPs. Besides sharpenings of know results, new results for the disjunctive stable model semantics (for models (DSM) as well as partial models (PDSM)) and the Perfect Model Semantics (PERF) have been derived. Furthermore, the complexity of the corresponding problems for extended disjunctive logic programs (EDLPs) under the answer set semantics has been settled. Our results provide the solutions to problems left open in [4] and to a conjecture in [37]. In particular, the question in [4] whether stable semantics can be transformed for all EDLPs in polynomial time into satisfiability or provability of a propositional formula has a negative answer unless the polynomial hierarchy collapses at its first level.

It appeared that $\mathcal{S}$-**Consistency** is Σ_2^P-complete for DSM, PDSM, ANSW, and PERF, and NP-complete for all remaining semantics except the Iterated Closed World Assumption (ICWA), which was restricted to programs without integrity clauses, i.e. clauses with empty heads. Under this restriction $\mathcal{S}$-**Consistency** remains Σ_2^P-hard for DSM, PDSM, ANSW, and PERF, but becomes polynomial for all other semantics. An intuitive explanation of Σ_2^P-hardness of DSM resp. PDSM are two interacting sources of complexity: The (potentially exponential) number of candidates M for a stable model and, due to a minimality condition on M, the difficulty of checking whether a candidate is a stable model. Analogous explanations applies to ANSW and PERF.

$\mathcal{S}$-**Entailment** turned out to be Π_2^P-hard for all considered semantics except the Disjunctive Database Rule (DDR) (resp. the equivalent Weak Generalized Closed World Assumption (WGCWA)) and the Possible Models Semantics (PMS) (resp. the equivalent Possible Worlds Semantics (PWS)), for which the problem is coNP-complete. For all Π_2^P-hard semantics the problem is also known to be in Π_2^P except the Generalized Closed World Assumption (GCWA) and the Careful Closed World Assumption (CCWA), for which $\Delta_3^P[O(\log n)]$ is the currently best known upper bound. It was shown that the Π_2^P-hardness results still hold under the restriction that the program contains no integrity clauses, that negation does not occur in the program, and that F is a literal. In this case, the problem was known to be polynomial for DDR, WGCWA, PMS, and PWS. An intuitive explanation of Π_2^P-hardness are two sources of complexity: The (potentially exponential) number of candidates for a model which does not satisfy F and, due to some minimality criterion, the difficulty of verifying that a classical model is a model under the particular semantics. This difficulty is caused by disjunctions in the heads of clauses or, in some case, alternatively by integrity clauses.

We believe that this paper supports a better understanding of the computational properties of finite propositional DLPs. Exact complexity characterizations of tasks in logic programming help to gain insight into obstacles to efficient logic programming; the reader is referred to [12, 59] for more on this issue.

The results of this paper and recent complexity results for nonmonotonic logics (cf. [12] for an overview) imply as a byproduct that queries to DLPs resp. EDLPs can be efficiently translated into reasoning tasks in many forms of nonmonotonic reasoning. For disjunctive default theory and autoepistemic logic we obtained new complexity results by such transformations. Vice versa, the results imply that reasoning tasks in many nonmonotonic formalisms can be efficiently reduced to disjunctive logic programming. The computational relationship underlines the close connection between disjunctive logic programming and nonmonotonic logics, and supports that logic programming is a competitive tool for knowledge representation.

Acknowledgements

The authors would like to thank Helmut Veith and an anonymous referee for their valuable comments on this paper.

References

[1] K. Apt and H. Blair. Arithmetic Classification of Perfect Models of Stratified Programs. In R. Kowalski and K. Bouwen, editors, *Proceedings of the Fifth Joint International Conference and Symposium on Logic Programming (JICSLP-88)*, pages 766–779. MIT Press, 1988.

[2] K. Apt, H. Blair, and A. Walker. Towards a Theory of Declarative Knowledge. In Minker [44],

pages 89–148.

[3] C. Baral, J. Lobo, and J. Minker. Generalized Disjunctive Well-Founded Semantics for Logic Programs: Declarative Semantics. In Z. Ras, M. Zemenkova, and M. Emrich, editors, *Proceedings Fourth International Symposium on Methods for Intelligent Systems (ISMIS-90)*, pages 465–473, October 1990.

[4] R. Ben-Eliyahu and R. Dechter. Propositional Semantics for Disjunctive Logic Programs. In *Proceedings Joint International Conference and Symposium on Logic Programming JICSLP-92*, pages 813–827, Washington DC, November 1992. Full paper to appear in *Annals of Mathematics and Artificial Intelligence.*

[5] N. Bidoit and C. Froidevaux. General Logic Databases and Programs: Default Semantics and Stratification. *Information and Computation*, 19:15–54, 1991.

[6] N. Bidoit and C. Froidevaux. Negation by Default and Unstratifiable Programs. *Theoretical Computer Science*, 78:85–112, 1991.

[7] H. Blair and C. Cholak. The Complexity of Local Stratification. Technical report, School of Computer and Information Sciences, Syracuse University, 1992.

[8] H. Blair, W. Marek, A. Nerode, and J. Remmel, editors. *Informal Proceedings of the Workshop on Structural Complexity and Recursion-Theoretic Methods in Logic Programming*, Washington DC, November 1992. Cornell University, Mathematical Sciences Institute.

[9] H. Blair, W. Marek, and J. Schlipf. The Expressiveness of Locally Stratified Programs. Technical Report 92-8, Mathematical Sciences Institute, Cornell University, 1992. *Annals of Mathematics and Artificial Intelligence* 1995, to appear.

[10] M. Cadoli. The complexity of model checking for circumscriptive formulae. *Information Processing Letters*, 44:113–118, 1992.

[11] M. Cadoli and M. Lenzerini. The Complexity of Closed World Reasoning and Circumscription. Technical Report RAP. 17.91, Universitá di Roma “La Sapienza”, Dipartimento di Informatica e Sistemistica, September 1991. Extended Abstract in *Proceedings AAAI-90*, pp. 550–555. To appear in *Journal of Computer and System Sciences.*

[12] M. Cadoli and M. Schaerf. A Survey of Complexity Results for Non-monotonic Logics. *Journal of Logic Programming*, 17:127–160, 1993.

[13] E. Chan. A Possible Worlds Semantics for Disjunctive Databases. *IEEE Transactions on Data and Knowledge Engineering*, 5(2):282–292, 1993.

[14] A. Chandra and D. Harel. Horn Clause Queries and Generalizations. *Journal of Logic Programming*, 2:1–15, 1985.

[15] R. Chang and P. Rohatgi. On Unique Satisfiability and Randomized Reductions. *Bulletin of the EATCS*, 47:151–159, 1990.

[16] J. Chomicki and V. S. Subrahmanian. Generalized Closed World Assumption is $\prod_2^0$-Complete. *Information Processing Letters*, 34:289–291, 1990.

[17] J. Dix and M. Müller. Abstract Properties and Computational Complexity of Semantics for Disjunctive Logic Programs. In Blair et al. [8], pages 15–28.

[18] J. Dix and M. Müller. Implementing Semantics of Disjunctive Logic Programs Using Fringes and Abstract Properties. In L.-M. Pereira and A. Nerode, editors, *Proceedings of the Second International Workshop on Logic Programming and Nonmonotonic Reasoning (LPNMR-93)*, pages 43–59, Lisbon, Portugal, July 1993. MIT Press.

[19] T. Eiter and G. Gottlob. Reasoning with Parsimonious and Moderately Grounded Expansions. *Fundamenta Informaticae*, 17(1,2):31–53, 1992.

[20] T. Eiter and G. Gottlob. Complexity Results for Disjunctive Logic Programming and Application to Nonmonotonic Logics. In D. Miller, editor, *Proceedings of the International Logic Programming Symposium (ILPS)*, pages 266–278, Vancouver, 1993. MIT Press.

[21] T. Eiter and G. Gottlob. Propositional Circumscription and Extended Closed World Reasoning are Π_2^P-complete. *Theoretical Computer Science*, 114(2):231–245, 1993. Addendum 118:315.

[22] J. Fernández and J. Minker. Semantics of Disjunctive Deductive Databases. In *Proceedings 4th Intl. Conference on Database Theory (ICDT-92)*, pages 21–50, Berlin, October 1992.

[23] M. Garey and D. S. Johnson. *Computers and Intractability – A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, 1979.

[24] M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. In *Proceedings Fifth Logic Programming Symposium*, pages 1070–1080, Cambridge Mass., 1988. MIT Press.

[25] M. Gelfond and V. Lifschitz. Classical Negation in Logic Programs and Disjunctive Databases. *New*

Generation Computing, 9:365–385, 1991.

[26] M. Gelfond and H. Przymusinska. Negation as Failure: Careful Closure Procedure. *Artificial Intelligence*, 30:273–287, 1986.

[27] M. Gelfond, H. Przymusinska, V. Lifschitz, and M. Truszczyński. Disjunctive Defaults. In *Proceedings International Conference on Principles of Knowledge Representation and Reasoning KR-91*, pages 230–237, 1991.

[28] M. Gelfond, H. Przymusinska, and T. Przymusinski. On the Relationship Between Circumscription and Negation as Failure. *Artificial Intelligence*, 38:75–94, 1989.

[29] G. Gottlob. Complexity Results for Nonmonotonic Logics. *Journal of Logic and Computation*, 2(3):397–425, June 1992.

[30] D. S. Johnson. A Catalog of Complexity Classes. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume A, chapter 2. Elsevier Science Publishers B.V. (North-Holland), 1990.

[31] H. Kautz and B. Selman. Hard Problems for Simple Default Logics. *Artificial Intelligence*, 49:243–279, 1991.

[32] V. Lifschitz. Computing Circumscription. In *Proceedings International Joint Conference on Artificial Intelligence IJCAI-85*, pages 121–127, 1985.

[33] V. Lifschitz and G. Schwarz. Extended Logic Programs as Autoepistemic Theories. In L.-M. Pereira and A. Nerode, editors, *Proceedings of the Second International Workshop on Logic Programming and Nonmonotonic Reasoning (LPNMR-93)*, pages 101–114, Lisbon, Portugal, July 1993. MIT Press.

[34] J. Lobo, J. Minker, and A. Rajasekar. *Foundations of Disjunctive Logic Programming.* MIT Press, Cambridge, MA, 1992.

[35] W. Marek, A. Nerode, and J. Remmel. A Theory of Nonmonotonic Rule Systems II. *Annals of Mathematics and Artificial Intelligence*, 5:229–264, 1992.

[36] W. Marek, A. Nerode, and J. Remmel. How Complicated is the Set of Stable Models of a Recursive Logic Program? *Annals of Pure and Applied Logic*, 56:119, 1992.

[37] W. Marek, A. Rajasekar, and M. Truszczyński. Complexity of Computing with Extended Propositional Logic Programs. In Blair et al. [8], pages 93–102.

[38] W. Marek and M. Truszczyński. Autoepistemic Logic. *Journal of the ACM*, 38(3):588–619, 1991.

[39] W. Marek and M. Truszczyński. Computing Intersection of Autoepistemic Expansions. In Nerode et al. [46], pages 37–50.

[40] W. Marek and M. Truszczyński. Reflexive Autoepistemic Logic and Logic Programming. In L.-M. Pereira and A. Nerode, editors, *Proceedings of the Second International Workshop on Logic Programming and Nonmonotonic Reasoning (LPNMR-93)*, pages 115–131, Lisbon, Portugal, July 1993. MIT Press.

[41] J. McCarthy. Circumscription – A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13:27–39, 1980.

[42] J. McCarthy. Applications of Circumscription to Formalizing Common-Sense Knowledge. *Artificial Intelligence*, 28:89–116, 1986.

[43] J. Minker. On Indefinite Data Bases and the Closed World Assumption. In *Proceedings of the* 6^{th} *Conference on Automated Deduction (CADE-82)*, pages 292–308, 1982.

[44] J. Minker, editor. *Foundations of Deductive Databases and Logic Programming.* Morgan Kaufman, Washington DC, 1988.

[45] R. Moore. Semantical Considerations on Nonmonotonic Logics. *Artificial Intelligence*, 25:75–94, 1985.

[46] A. Nerode, W. Marek, and V. Subrahmanian, editors. *Proceedings of the 1st Intl. Workshop on Logic Programming and Nonmonotonic Reasoning (LPNMR-91)*, Washington DC, July 1991. MIT Press.

[47] I. Niemelä. Towards Automatic Reasoning. In *Proc. Europ. Workshop on Logics in AI, Amsterdam, September 1990*, number 478 in Lecture Notes in Computer Science. Springer, 1991.

[48] T. Przymusinski. On the Declarative and Procedural Semantics of Stratified Deductive Databases. In Minker [44], pages 193–216.

[49] T. Przymusinski. Stable Semantics for Disjunctive Programs. *New Generation Computing*, 9:401–424, 1991.

[50] T. Przymusinski. Three-valued Nonmonotonic Formalisms and Semantics of Logic Programming. *Artificial Intelligence*, 49:309–344, 1991.

[51] A. Rajasekar, J. Lobo, and J. Minker. Weak Generalized Closed World Assumption. *Journal of*

Automated Reasoning, 5:293–307, 1989.
[52] R. Reiter. On Closed-World Databases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 55–76. Plenum Press, New York, 1978.
[53] R. Reiter. A Logic for Default Reasoning. *Artificial Intelligence*, 13:81–132, 1980.
[54] K. Ross. The Well-Founded Semantics for Disjunctive Logic Programs. In *Proceedings First Intl. Conf. on Dedcutive and Object-Oriented Databases (DOOD-89)*, pages 352–369, 1989.
[55] K. Ross and R. Topor. Inferring Negative Information From Disjunctive Databases. *Journal of Automated Reasoning*, 4(2):397–424, 1988.
[56] C. Sakama. Possible Model Semantics for Disjunctive Databases. In *Proceedings First Intl. Conf. on Dedcutive and Object-Oriented Databases (DOOD-89)*, pages 337–351, Kyoto Japan, 1989.
[57] M. Schaerf. Logic Programming and Autoepistemic Logics: New Relations and Complexity Results. In P. Torasso, editor, *Proceedings of the Italian AI Conference (IA*AI)*, number 728 in Springer LNCS/AI, October 1993.
[58] J. Schlipf. The Expressive Powers of Logic Programming Semantics. Technical Report CIS-TR-90-3, Computer Science Department, University of Cincinnati, 1990. Preliminary version in *Proceedings PODS-90*, pp. 196–204. To appear in *Journal of Computer and System Sciences*.
[59] J. Schlipf. A Survey of Complexity and Undecidability Results in Logic Programming. In Blair et al. [8], pages 93–102.
[60] J. S. Schlipf. When is Closed World Reasoning Tractable ? In Z. Ras and L. Saitta, editors, *Proceedings Third International Symposium on Methodologies for Intelligent Systems (ISMIS-88)*, pages 485–494, Turin, Italy, October 12-15 1988. Elsevier Science Publishers B.V. (North-Holland).
[61] G. Schwarz. Autoepistemic Logic of Knowledge. In Nerode et al. [46], pages 260–274.
[62] J. Stillman. The Complexity of Propositional Default Logic. In *Proceedings AAAI-92*, pages 794–799, 1992.
[63] L. Stockmeyer and A. Meyer. Word Problems Requiring Exponential Time. In *Proceedings of the Fifth ACM Symposium on the Theory of Computing (STOC-73)*, pages 1–9, 1973.
[64] A. van Gelder. Negation as Failure Using Tight Derivations for General Logic Programs. In Minker [44], pages 1149–1176.
[65] A. van Gelder, K. Ross, and J. Schlipf. The Well-Founded Semantics for General Logic Programs. *Journal of the ACM*, 38(3):620–650, 1991.
[66] K. Wagner. Bounded Query Classes. *SIAM J. Comp.*, 19(5):833–846, 1990.
[67] A. Yahya and L. Henschen. Deduction in Non-Horn Databases. *Journal of Automated Reasoning*, 1(2):141–160, 1985.