Complexity of Super-Coherence Problems in ASP *

Mario Alviano[1], Wolfgang Faber[1], and Stefan Woltran[2]

[1] University of Calabria, Italy
{alviano,faber}@mat.unical.it
[2] Vienna University of Technology, Austria
woltran@dbai.tuwien.ac.at

Abstract. Adapting techniques from database theory in order to optimize Answer Set Programming (ASP) systems, and in particular the grounding components of ASP systems, is an important topic in ASP. In recent years, the Magic Set method has received some interest in this setting, and a variant of it, called DMS, has been proposed for ASP. However, this technique has a caveat, because it is not correct (in the sense of being query-equivalent) for all ASP programs. In recent work, a large fragment of ASP programs, referred to as *super-coherent programs*, has been identified, for which DMS is correct. An open question remained: How complex is it to determine whether a given program is super-coherent? This question turned out to be quite difficult to answer precisely. In this paper, we formally prove that deciding whether a propositional program is super-coherent is Π_3^P-complete in the disjunctive case, while it is Π_2^P-complete for normal programs. The hardness proofs are the difficult part in this endeavor: We proceed by characterizing the reductions by the models and reduct models which the ASP programs should have, and then provide instantiations that meet the given specifications.

1 Introduction

Answer Set Programming (ASP) is a powerful formalism for knowledge representation and common sense reasoning [5]. Allowing disjunction in rule heads and nonmonotonic negation in bodies, ASP can express every query belonging to the complexity class Σ_2^{P} ($\mathrm{NP}^{\mathrm{NP}}$). Encouraged by the availability of efficient inference engines, such as DLV [17], GnT [15], Cmodels [18], or ClaspD [8], ASP has found several practical applications in various domains, including data integration [16], semantic-based information extraction [20, 21], e-tourism [24], workforce management [25], and many more. As a matter of fact, these ASP systems are continuously enhanced to support novel optimization strategies, enabling them to be effective over increasingly larger application domains.

Frequently, optimization techniques are inspired by methods that had been proposed in other fields, for example database theory, satisfiability solving, or constraint satisfaction. Among techniques adapted to ASP from database theory, Magic Sets [26, 4, 6]

* This work will also be presented at the ASPOCP 2011 workshop. Partly supported by Regione Calabria and EU under POR Calabria FESR 2007-2013 and within the PIA project of DLVSYSTEM s.r.l., and by MIUR under the PRIN project LoDeN. We also thank the anonymous reviewers for their valuable comments.

have recently achieved a lot of attention. Following some earlier work [14, 7], recently an adapted method called *DMS* has been proposed for ASP in [3]. However, this technique has a caveat, because it is not correct (in the sense of being query-equivalent) for all ASP programs. In recent work [2, 1], a large fragment of ASP programs, referred to as *super-coherent programs* ($\mathrm{ASP}^{\mathrm{sc}}$), has been identified, for which DMS can be proved to be correct.

While our main motivation for studying $\mathrm{ASP}^{\mathrm{sc}}$ stemmed from the applicability of DMS, this class actually has many more important motivations. Indeed, it can be viewed as the class of *non-constraining programs*: Adding extensional information to these programs will always result in answer sets. One important implication of this property is for modular evaluation. For instance, when using the splitting set theorem of [19], if a top part of a split program is an $\mathrm{ASP}^{\mathrm{sc}}$ program, then any answer set of the bottom part will give rise to at least one answer set of the full program—so for determining answer set existence, there would be no need to evaluate the top part.

On a more abstract level, one of the main criticisms of ASP (being voiced especially in database theory) is that there are programs which do not admit any answer set (traditionally this has been considered a more serious problem than the related nondeterminism in the form of multiple answer sets, cf. [23]). From this perspective, programs which guarantee coherence (existence of an answer set) have been of interest for quite some time. In particular, if one considers a fixed program and a variable "database," one arrives naturally at the class $\mathrm{ASP}^{\mathrm{sc}}$ when requiring existence of an answer set. This also indicates that deciding super-coherence of programs is related to some problems from the area of equivalence checking in ASP [13, 10, 22]. For instance, when deciding whether, for a given arbitrary program P, there is a uniformly equivalent definite positive (or definite Horn) program, super-coherence of P is a necessary condition—this is straightforward to see because definite Horn programs have exactly one answer set, so a non-super-coherent program cannot be uniformly equivalent to any definite Horn program.

Since the property of being super-coherent is a semantic one, a natural question arises: How difficult is it to decide whether a given program belongs to $\mathrm{ASP}^{\mathrm{sc}}$? It turns out that the precise complexity is rather difficult to establish. Some bounds have been given in [2], in particular showing decidability, but especially hardness results seemed quite hard to obtain.

In order to focus on the essentials of this problem, in this paper we deal with propositional programs and show the precise complexity (in terms of completeness) for deciding whether a given propositional ASP program belongs to $\mathrm{ASP}^{\mathrm{sc}}$. In Section 2 we first define some terminology needed later on. In Section 3 we formulate the problem that we analyze and state the results. The remainder of the paper contains the proofs — in Section 4 for disjunctive programs and in Section 5 for normal programs — and in Section 6 we briefly discuss the relation to equivalence problems before concluding the work in Section 7.

2 Preliminaries

In this paper we consider propositional programs, so an atom p is a member of a countable set $\mathcal{U}$. A *literal* is either an atom p (a positive literal), or an atom preceded by the *negation as failure* symbol not (a negative literal). A *rule* r is of the form

$$p_1 \vee \cdots \vee p_n \leftarrow q_1, \ldots, q_j, not\ q_{j+1}, \ldots, not\ q_m$$

where $p_1, \ldots, p_n, q_1, \ldots, q_m$ are atoms and $n \geq 0, m \geq j \geq 0$. The disjunction $p_1 \vee \cdots \vee p_n$ is the *head* of r, while the conjunction $q_1, \ldots, q_j, not\ q_{j+1}, \ldots, not\ q_m$ is the *body* of r. Moreover, $H(r)$ denotes the set of head atoms, while $B(r)$ denotes the set of body literals. We also use $B^+(r)$ and $B^-(r)$ for denoting the set of atoms appearing in positive and negative body literals, respectively, and $At(r)$ for the set $H(r) \cup B^+(r) \cup B^-(r)$. A rule r is normal (or disjunction-free) if $|H(r)| = 1$ or $|H(r)| = 0$ (in this case r is also referred to as a *constraint*), positive (or negation-free) if $B^-(r) = \emptyset$, a *fact* if both $B(r) = \emptyset$ and $|H(r)| = 1$.

A *program* P is a finite set of rules; if all rules in it are positive (resp. normal), then P is a positive (resp. normal) program. Odd-cycle-free and stratified programs constitute two other interesting classes of programs. An atom p appearing in the head of a rule r *depends* on each atom q that belongs to $B(r)$; if q belongs to $B^+(r)$, p depends positively on q, otherwise negatively. A program without constraints is *odd-cycle-free* if there is no cycle of dependencies involving an odd number of negative dependencies, while it is *stratified* if each cycle of dependencies involves only positive dependencies. Programs containing constraints have been excluded by the definition of odd-cycle-free and stratified programs. In fact, constraints intrinsically introduce odd-cycles in programs as a constraint of the form

$$\leftarrow q_1, \ldots, q_j, not\ q_{j+1}, \ldots, not\ q_m$$

can be replaced by the following equivalent rule:

$$co \leftarrow q_1, \ldots, q_j, not\ q_{j+1}, \ldots, not\ q_m, not\ co,$$

where co is a fresh atom (i.e., an atom that does not occur elsewhere in the program).

Given a program P, let $At(P)$ denote the set of atoms that occur in it, that is, let $At(P) = \bigcup_{r \in P} At(r)$. An *interpretation* I for a program P is a subset of $At(P)$. An atom p is true w.r.t. an interpretation I if $p \in I$; otherwise, it is false. A negative literal $not\ p$ is true w.r.t. I if and only if p is false w.r.t. I. The body of a rule r is true w.r.t. I if and only if all the body literals of r are true w.r.t. I, that is, if and only if $B^+(r) \subseteq I$ and $B^-(r) \cap I = \emptyset$. An interpretation I *satisfies* a rule $r \in P$ if at least one atom in $H(r)$ is true w.r.t. I whenever the body of r is true w.r.t. I. An interpretation I is a *model* of a program P if I satisfies all the rules in P.

Given an interpretation I for a program P, the reduct of P w.r.t. I, denoted by P^I, is obtained by deleting from P all the rules r with $B^-(r) \cap I \neq \emptyset$, and then by removing all the negative literals from the remaining rules. The semantics of a program P is given by the set $AS(P)$ of the answer sets of P, where an interpretation M is an answer set for P if and only if M is a subset-minimal model of P^M.

In the subsequent sections, we will use the following properties that the models and models of reducts of programs satisfy (see, e.g. [9, 13]):

(P1) for any disjunctive program P and interpretations $I \subseteq J \subseteq K$, if I satisfies P^J, then I also satisfies P^K;
(P2) for any normal program P and interpretations $I, J \subseteq K$, if I and J both satisfy P^K, then also $(I \cap J)$ satisfies P^K.

We now introduce super-coherent ASP programs ($\mathrm{ASP}^{\mathrm{sc}}$ programs), the main class of programs studied in this paper.

Definition 1 ($\mathrm{ASP}^{\mathrm{sc}}$ **programs [1, 2]).** *A program P is* super-coherent *if, for every set of facts F, $AS(P \cup F) \neq \emptyset$. Let* $\mathrm{ASP}^{\mathrm{sc}}$ *denote the set of all super-coherent programs.*

Note that $\mathrm{ASP}^{\mathrm{sc}}$ programs include all odd-cycle-free programs (and therefore also all stratified programs). Indeed, every odd-cycle-free program admits at least one answer set and remains odd-cycle-free even if an arbitrary set of facts is added to its rules. On the other hand, there are programs having odd-cycles that are in $\mathrm{ASP}^{\mathrm{sc}}$, cf. [2].

An important question regards whether $\mathrm{ASP}^{\mathrm{sc}}$ programs are as expressive as ASP programs. Of course, checking coherence (existence of answer sets) is a trivial task for $\mathrm{ASP}^{\mathrm{sc}}$ programs. But when considering query answering, it turns out that expressivity is not lowered. Indeed, all expressivity results of [12] hold for disjunctive programs with stratified negation (examining the proofs, actually for disjunctive programs with input negation, that is, having at most two strata), which guarantee super-coherence and are a proper subset of $\mathrm{ASP}^{\mathrm{sc}}$. It therefore follows that all properties in Σ_2^P or Π_2^P are expressible by $\mathrm{ASP}^{\mathrm{sc}}$ programs using a query under brave or cautious reasoning, respectively. The picture is less clear for nondisjunctive $\mathrm{ASP}^{\mathrm{sc}}$ programs.

However, we should point out that many (probably most) existing ASP programs follow a "Guess&Check" or "Generate&Test" methodology, which usually relies on integrity constraints, the presence of which usually contradicts super-coherence. As an alternative, violated integrity constraints can derive a special atom, on which the query atom should depend negatively. If the Guess/Generate part involves non-stratified negation, it depends on how this construct is used in the encoding. If it just encodes a choice, this can often be easily converted to a disjunction, while for encodings that entangle guess and check using unstratified negation, an automated conversion to an $\mathrm{ASP}^{\mathrm{sc}}$ program seems less straightforward. In general, however, we feel that for obtaining computationally efficient $\mathrm{ASP}^{\mathrm{sc}}$ encodings, different encoding methodologies should be developed.

3 Problem Statement and Main Theorems

In this paper, we study the complexity of the following natural problem.

– Given a program P, is P super-coherent, i.e. does $AS(P \cup F) \neq \emptyset$ hold for any set F of facts.

We will study the complexity for this problem for the case of disjunctive logic programs and non-disjunctive (normal) logic programs. We first have a look at a similar problem, which turns out to be rather trivial to decide.

Proposition 1. *The problem of deciding whether, for a given disjunctive program P, there is a set F of facts such that $AS(P \cup F) \neq \emptyset$ is* NP*-complete;* NP*-hardness holds already for normal programs.*

Proof. We start by observing that there is F such that $AS(P \cup F) \neq \emptyset$ if and only if P has at least one classical model. Indeed, if M is a model of P, then $P \cup M$ has M as its answer set. On the other hand, if P has no model, then no addition of facts F will yield an answer set for $P \cup F$. It is well known that deciding whether a program has at least one (classical) model is NP-complete for both disjunctive and normal programs. □

In contrast, the complexity for deciding super-coherence is surprisingly high, which we shall show next. To start, we give a straight-forward observation.

Proposition 2. *A program P is super-coherent if and only if for each set $F \subseteq At(P)$, $AS(P \cup F) \neq \emptyset$.*

Proof. The only-if direction is by definition. For the if-direction, let F be any set of facts. F can be partitioned into $F' = F \cap At(P)$ and $F'' = F \setminus F'$. By assumption, $P \cup F'$ is coherent. Let M be an answer set of $P \cup F'$. We shall show that $M \cup F''$ is an answer set of $P \cup F = P \cup F' \cup F''$. This is in fact a consequence of the splitting set theorem [19], as the atoms in F'' are only defined by facts not occurring in $P \cup F'$. □

Our main results are as follows. The proofs are contained in the subsequent sections.

Theorem 1. *The problem of deciding super-coherence for disjunctive programs is Π_3^P-complete.*

Theorem 2. *The problem of deciding super-coherence for normal programs is Π_2^P-complete.*

4 Proof of Theorem 1

Membership follows by the following straight-forward nondeterministic algorithm for the complementary problem, i.e. given a program P, does there exist a set F of facts such that $AS(P \cup F) = \emptyset$: we guess a set $F \subseteq At(P)$ and check $AS(P \cup F) = \emptyset$ via an oracle-call. Restricting the guess to $At(P)$ can be done by Proposition 2. Checking $AS(P \cup F) = \emptyset$ is known to be in Π_2^P [11]. This shows Π_3^P-membership.

For the hardness we reduce the Π_3^P-complete problem of deciding whether QBFs of the form $\forall X \exists Y \forall Z \phi$ are true to the problem of super-coherence. Without loss of generality, we can consider ϕ to be in DNF and, indeed, $X \neq \emptyset$, $Y \neq \emptyset$, and $Z \neq \emptyset$. We also assume that each disjunct of Φ contains at least one variable from X, one from Y and one from Z. More precisely, we shall construct for each such QBF Φ a program P_Φ such that Φ is true iff P_Φ is super-coherent. Before showing how to actually construct P_Φ from Φ in polynomial time, we give the required properties for P_Φ. We then show that for programs P_Φ satisfying these properties, the desired relation (Φ is true iff P_Φ is super-coherent) holds, and finally we provide the construction of P_Φ.

Definition 2. *Let $\Phi = \forall X \exists Y \forall Z \phi$ be a QBF with ϕ in DNF. We call any program P satisfying the following properties a* Φ-reduction*:*

1. *P is given over atoms $U = X \cup Y \cup Z \cup \overline{X} \cup \overline{Y} \cup \overline{Z} \cup \{u, v, w\}$, where all atoms in sets $\overline{S} = \{\overline{s} \mid s \in S\}$ and $\{u, v, w\}$ are fresh and mutually disjoint;*
2. *P has the following models:*
   - *U*
   - *for each $I \subseteq X$, $J \subseteq Y$,*

$$M[I, J] = I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)} \cup Z \cup \overline{Z} \cup \{u, v\}$$

   and

$$M'[I, J] = I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)} \cup Z \cup \overline{Z} \cup \{v, w\};$$

3. *for each $I \subseteq X$, $J \subseteq Y$, the models[3] of the reduct $P^{M[I,J]}$ are $M[I, J]$ and*

$$O[I] = I \cup \overline{(X \setminus I)};$$

4. *for each $I \subseteq X$, $J \subseteq Y$, the models of the reduct $P^{M'[I,J]}$ are $M'[I, J]$ and*
   - *for each $K \subseteq Z$ such that $I \cup J \cup K \not\models \phi$,*

$$N[I, J, K] = I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)} \cup K \cup \overline{(Z \setminus K)} \cup \{v\};$$

5. *the models of the reduct P^U are given only by the models already mentioned above, i.e. U itself, $M[I, J]$, $M'[I, J]$, and $O[I]$, for each $I \subseteq X$, $J \subseteq Y$, and $N[I, J, K]$ for each $I \subseteq X$, $J \subseteq Y$, $K \subseteq Z$, such that $I \cup J \cup K \not\models \phi$.*

The structure of models of Φ-reductions and the "countermodels" (see below what we mean by this term) of the relevant reducts is sketched in Figure 1. The center of the diagram contains the models of the Φ-reduction and their subset relationship. For each of the model the respective box lists the "countermodels," by which we mean those reduct models which can serve as counterexamples for the original model being an answer set, that is, those reduct models which are proper subsets of the original model.

We just note at this point that the models of the reduct P^U given in Item 5 are not specified for particular purposes, but are required to allow for a realization via disjunctive programs. In fact, these models are just an effect of property (P1) mentioned in Section 2. However, before showing a program satisfying the properties of a *Φ-reduction*, we first show the rationale behind the concept of Φ-reductions.

Lemma 1. *For any QBF $\Phi = \forall X \exists Y \forall Z \phi$ with ϕ in DNF, a Φ-reduction is super-coherent iff Φ is true.*

Proof. Suppose that Φ is false. Hence, there exists an $\mathcal{I} \subseteq X$ such that, for all $J \subseteq Y$, there is a $K_Y \subseteq Z$ with $\mathcal{I} \cup J \cup K_Y \not\models \phi$. Now let P be any Φ-reduction and $F_{\mathcal{I}} = \mathcal{I} \cup \overline{(X \setminus \mathcal{I})}$. We show that $AS(P \cup F_{\mathcal{I}}) = \emptyset$, thus P is not super-coherent. Let $\mathcal{M}$ be a model of $P \cup F_{\mathcal{I}}$. Since P is a Φ-reduction, the only candidates for $\mathcal{M}$ are U, $M[\mathcal{I}, J]$, and $M'[\mathcal{I}, J]$, where $J \subseteq Y$. Indeed, for each $I \neq \mathcal{I}$, $M[I, J]$ and $M'[I, J]$ cannot be models of $P \cup F_{\mathcal{I}}$ because $F_{\mathcal{I}} \not\subseteq M[I, J]$, resp. $F_{\mathcal{I}} \not\subseteq M'[I, J]$. We now analyze these three types of potential candidates:

[3] Here and below, for a reduct P^M we only list models of the form $N \subseteq M$, since those are the relevant ones for our purposes. Recall that $N = M$ is always a model of P^M in case M is a model of P.

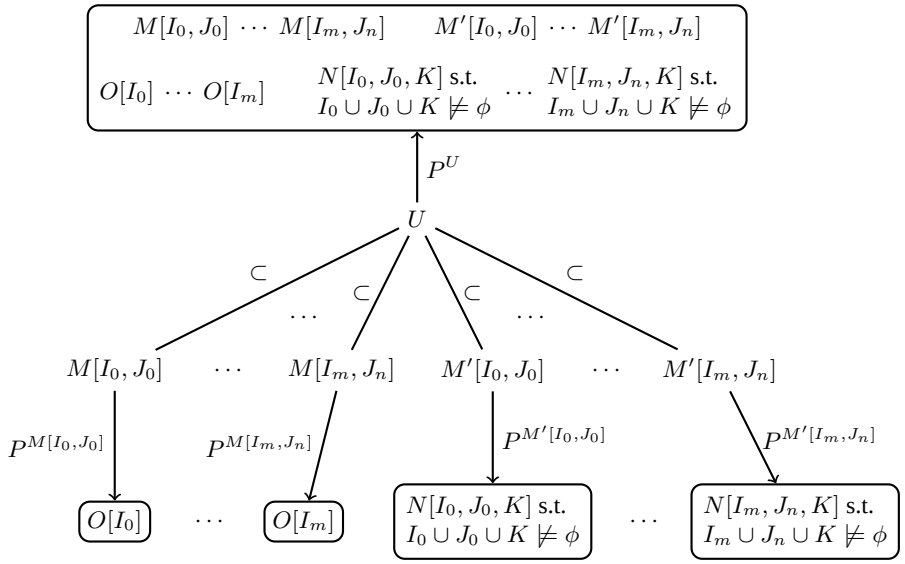

Fig. 1. Models and reduct "countermodels" of Φ-reductions

- $\mathcal{M} = U$: Then, for instance, $M[\mathcal{I}, J] \subset U$ is a model of $(P \cup F_{\mathcal{I}})^{\mathcal{M}} = P^{\mathcal{M}} \cup F_{\mathcal{I}}$ for any $J \subseteq Y$. Thus, $\mathcal{M} \notin AS(P \cup F_{\mathcal{I}})$.
- $\mathcal{M} = M[\mathcal{I}, J]$ for some $J \subseteq Y$. Then, by the properties of Φ-reductions, $O[\mathcal{I}] \subset \mathcal{M}$ is a model of $(P \cup F_{\mathcal{I}})^{\mathcal{M}} = P^{\mathcal{M}} \cup F_{\mathcal{I}}$. Thus, $\mathcal{M} \notin AS(P \cup F_{\mathcal{I}})$.
- $\mathcal{M} = M'[\mathcal{I}, J]$ for some $J \subseteq Y$. By the initial assumption, there exists a $K_Y \subseteq Z$ with $\mathcal{I} \cup J \cup K_Y \not\models \phi$. Then, by the properties of Φ-reductions, $N[\mathcal{I}, J, K] \subset \mathcal{M}$ is a model of $P^{\mathcal{M}}$. Thus, $\mathcal{M} \notin AS(P \cup F_{\mathcal{I}})$.

In each of the cases we have obtained $\mathcal{M} \notin AS(P \cup F_{\mathcal{I}})$, hence $AS(P \cup F_{\mathcal{I}}) = \emptyset$ and P is not super-coherent.

Suppose that Φ is true. It is sufficient to show that for each $F \subseteq U$, $AS(P \cup F) \neq \emptyset$. We have the following cases:

If $\{s, \overline{s}\} \subseteq F$ for some $s \in X \cup Y$ or $\{u, w\} \subseteq F$. Then $U \in AS(P \cup F)$ since U is a model of $P \cup F$ and each potential model $M \subset U$ of the reduct P^U (see the properties of Φ-reductions) does not satisfy $F \subseteq M$; thus each such M is not a model of $P^U \cup F = (P \cup F)^U$.

Otherwise, we have $F \subseteq M[I, J]$ or $F \subseteq M'[I, J]$ for some $I \subseteq X$, $J \subseteq Y$. In case $F \subseteq M[I, J]$ and $F \not\subseteq O[I]$, we observe that $M[I, J] \in AS(P \cup F)$ since $O[I]$ is the only model of the reduct $P^{M[I,J]}$. Thus for each such F there cannot be a model $M \subset M[I, J]$ of $P^{M[I,J]} \cup F = (P \cup F)^{M[I,J]}$. As well, in case $F \subseteq M'[I, J]$, where $w \in F$, $M'[I, J]$ can be shown to be an answer set of $P \cup F$. Indeed, in this case no $M \subset M'[I, J]$ is a model of $P^{M'[I,J]}$ because Φ is true.

It remains to consider the case $F \subseteq O[I]$ for each $I \subseteq X$. We show that $M'[I, J]$ is an answer set of $P \cup F$, for some $J \subseteq Y$. Since Φ is true, we know that, for each

$I \subseteq X$, there exists a $J_I \subseteq Y$ such that, for all $K \subseteq Z$, $I \cup J_I \cup K \models \phi$. As can be verified by the properties of Φ-reductions, then there is no model $M \subset M'[I, J_I]$ of $P^{M'[I,J_I]}$. Consequently, there is also no such model of $(P \cup F)^{M'[I,J_I]}$, and thus $M'[I, J_I] \in AS(P \cup F)$.

So in each of these cases $AS(P \cup F) \neq \emptyset$ and since these cases cover all possible $F \subseteq U$, we obtain that P is supercoherent.

In total we have shown that Φ being false implies that any Φ-reduction P is not super-coherent, while Φ being true implies that any Φ-reduction is super-coherent, which proves the lemma.

□

It remains to show that for any QBF of the desired form, a Φ-reduction can be obtained in polynomial time (w.r.t. the size of Φ). For the construction below, let us denote a negated atom a in the propositional part of the QBF Φ as $\overline{a}$.

Definition 3. *For any QBF $\Phi = \forall X \exists Y \forall Z \phi$ with $\phi = \bigvee_{i=1}^{n} l_{i,1} \wedge \cdots \wedge l_{i,m_i}$ a DNF (i.e., a disjunction of conjunctions over literals), we define*

$$
\begin{aligned}
P_\Phi = \{&x \vee \overline{x} \leftarrow;\ u \leftarrow x, \overline{x};\ w \leftarrow x, \overline{x};\ x \leftarrow u, w;\ \overline{x} \leftarrow u, w \mid x \in X\} \cup && (1)\\
&\{y \vee \overline{y} \leftarrow v;\ u \leftarrow y, \overline{y};\ w \leftarrow y, \overline{y};\ y \leftarrow u, w;\\
&\ \overline{y} \leftarrow u, w;\ v \leftarrow y;\ v \leftarrow \overline{y} \mid y \in Y\} \cup && (2)\\
&\{z \vee \overline{z} \leftarrow v;\ u \leftarrow z, not\ w;\ u \leftarrow \overline{z}, not\ w;\ v \leftarrow z;\ v \leftarrow \overline{z};\\
&\ z \leftarrow w;\ \overline{z} \leftarrow w;\ z \leftarrow u;\ \overline{z} \leftarrow u;\ w \vee u \leftarrow z, \overline{z} \mid z \in Z\} \cup && (3)\\
&\{w \vee u \leftarrow l_{i,1}, \ldots, l_{i,m_i} \mid 1 \leq i \leq n\} && (4)\\
&\{v \leftarrow w;\ v \leftarrow u;\ v \leftarrow not\ u\}. && (5)
\end{aligned}
$$

Obviously, the program from above definition can be constructed in polynomial time in the size of the reduced QBF. To conclude the proof of Theorem 1 it is thus sufficient to show the following relation.

Lemma 2. *For any QBF $\Phi = \forall X \exists Y \forall Z \phi$, the program P_Φ is a Φ-reduction.*

Proof. Obviously, $At(P_\Phi)$ contains the atoms as required in 1) of Definition 2. We continue to show 2). To see that U is a model of P_Φ is obvious. We next show that the remaining models M are all of the form $M[I, J]$ or $M'[I, J]$. First we have $v \in M$ because of the rules $v \leftarrow u$ and $v \leftarrow not\ u$ in (5). In case $w \in M$, $Z \cup \overline{Z} \subseteq M$ by the rules in (3). In case $w \notin M$, we have $K \cup \overline{(Z \setminus K)} \subseteq M$ for some $K \subseteq Z$, since $v \in M$ and by (3). But then, since $w \notin M$, $u \in M$ holds (rules $u \leftarrow z, not\ w$ resp. $u \leftarrow \overline{z}, not\ w$). Hence, also here $Z \cup \overline{Z} \subseteq M$. In both cases, we observe that by (1) and (2), $I \cup \overline{(X \setminus U)} \cup J \cup \overline{(Y \setminus J)} \subseteq M$, for some $I \subseteq X$ and $J \subseteq Y$. This yields the desired models, $M[I, J]$, $M'[I, J]$. It can be checked that no other model exists by showing that for $N \not\subseteq M[I, J]$, resp. $N \not\subseteq M'[I, J]$, $N = U$ follows.

We next show that, for each $I \subseteq X$ and $J \subseteq Y$, $P^{M[I,J]}$ and $P^{M'[I,J]}$ possess the required models. Let us start by showing that $O[I]$ is a model of $P^{M[I,J]}$. In fact, it can be observed that all of the rules of the form $x \vee \overline{x} \leftarrow$ in (1) are satisfied because either

x or $\overline{x}$ belong to $O[I]$, while all of the other rules in $P^{M[I,J]}$ are satisfied because of a false body literal. We also note that each strict subset of $O[I]$ does not satisfy some rule of the form $x \vee \overline{x} \leftarrow$, and thus it is not a model of $P^{M[I,J]}$. Similarly, any interpretation W such that $O[I] \subset W \subset M[I,J]$ does not satisfy some rule in $P^{M[I,J]}$ (note that rules of the form $u \leftarrow z$ and $u \leftarrow \overline{z}$ occur in $P^{M[I,J]}$ because $w \notin M[I,J]$; such rules are obtained by rules in (3)).

Let us now consider $P^{M'[I,J]}$ and let $W \subseteq M'[I,J]$ be one of its models. We shall show that either $W = M'[I,J]$, or $W = N[I,J,K]$ for some $K \subseteq Z$ such that $I \cup J \cup Z \not\models \phi$. Note that v is a fact in $P^{M'[I,J]}$, hence v must belong to W. By (1) and (2), since $v \in W$ and $W \subseteq M'[I,J]$, we can conclude that all of the atoms in $I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)}$ belong to W. Consider now the atom w. If w belongs to W, by the rules in (3) we conclude that all of the atoms in $Z \cup \overline{Z}$ belong to W, and thus $W = M'[I,J]$. Otherwise, if $w \notin W$, by the rules of the form $z \vee \overline{z} \leftarrow v$ in (3), there must be a set $K \subseteq Z$ such that $K \cup \overline{(Z \setminus K)}$ is contained in W. Note that no other atoms in $Z \cup \overline{Z}$ can belong to W because of the last rule in (3). Hence, $W = N[I,J,K]$. Moreover, $w \notin W$ and $u \notin W$ imply that $I \cup J \cup K \not\models \phi$ holds because of (4).

Finally, one can show that P^U does not yield additional models as those which are already present by other models. Let $W \subseteq U$ be a model of P^U. By (1), $O[I] \subseteq W$ must hold for some $I \subseteq X$. Consider now the atom v. If $v \notin W$, we conclude that the model W is actually $O[I]$. We can thus consider the other case, i.e. $v \in W$. By (2), $J \cup \overline{(Y \setminus J)} \subseteq W$ must hold for some $J \subseteq Y$. Consider now the atom u. If $u \in W$, we have $Z \cup \overline{Z} \subseteq W$ because of (3). If no other atom belongs to W, then $W = M[I,J]$ holds. Otherwise, if any other atom belongs to W, it can be checked that W must be equal to U. We can then consider the case in which $u \notin W$, and the atom w. Again, we have two possibilities. If w belongs to W, by (3) we conclude that all of the atoms in $Z \cup \overline{Z}$ belong to W, and thus either $W = M'[I,J]$ or $W = U$. Otherwise, if $w \notin W$, by the rules of the form $z \vee \overline{z} \leftarrow v$ in (3), there must be a set $K \subseteq Z$ such that $K \cup \overline{(Z \setminus K)}$ is contained in W. Note that no other atoms in $Z \cup \overline{Z}$ can belong to W because of the last rule in (3). Hence, $W = N[I,J,K]$. Moreover, because of (4), $w \notin W$ and $u \notin W$ imply that $I \cup J \cup K \not\models \phi$ holds. □

Note that the program from Definition 3 does not contain constraints. As a consequence, the Π_3^P-hardness result presented in this section also holds if we only consider disjunctive ASP programs without constraints.

5 Proof of Theorem 2

Membership follows by the straight-forward nondeterministic algorithm for the complementary problem presented in the previous section. We have just to note that a $co-\mathrm{NP}$ oracle can be used for checking the consistency of a normal program. Thus, Π_2^P-membership is established.

For the hardness we reduce the Π_2^P-complete problem of deciding whether QBFs of the form $\forall X \exists Y \phi$ are true to the problem of super-coherence. Without loss of generality, we can consider ϕ to be in CNF and, indeed, $X \neq \emptyset$, and $Y \neq \emptyset$. We also assume that

each clause of Φ contains at least one variable from X and one from Y. More precisely, we shall adapt the notion of Φ-reduction to normal programs. In particular, we have to take into account a fundamental difference between disjunctive and normal programs: while disjunctive programs allow for using disjunctive rules for guessing a subset of atoms, such a guess can be achieved only by means of unstratified negation within a normal program. For example, one atom in a set $\{x, y\}$ can be guessed by means of the following disjunctive rule: $x \vee y \leftarrow$. Within a normal program, the same result can be obtained by means of the following rules: $x \leftarrow not\ y$ and $y \leftarrow not\ x$. However, these last rules would be deleted in the reduced program associated with a model containing both x and y, which would allow for an arbitrary subset of $\{x, y\}$ to be part of a model of the reduct. More generally speaking, we have to take Property (P2), as introduced in Section 2, into account. This makes the following definition a bit more cumbersome compared to Definition 2.

Definition 4. *Let $\Phi = \forall X \exists Y \phi$ be a QBF with ϕ in CNF. We call any program P satisfying the following properties a* Φ-norm-reduction*:*

1. *P is given over atoms $U = X \cup Y \cup \overline{X} \cup \overline{Y} \cup \{v, w\}$, where all atoms in sets $\overline{S} = \{\overline{s} \mid s \in S\}$ and $\{v, w\}$ are fresh and mutually disjoint;*
2. *P has the following models:*
   - *for each $J \subseteq Y$, and for each J^* such that $J \cup \overline{(Y \setminus J)} \subseteq J^* \subseteq Y \cup \overline{Y}$*
   $$O[J^*] = X \cup \overline{X} \cup J^* \cup \{v, w\};$$
   - *for each $I \subseteq X$,*
   $$M[I] = I \cup \overline{(X \setminus I)} \cup \{v\};$$
   - *for each $I \subseteq X$, $J \subseteq Y$, such that $I \cup J \models \phi$,*
   $$N[I, J] = I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)} \cup \{w\};$$
3. *the only models of a reduct $P^{M[I]}$ are $M[I]$ and $M[I] \setminus \{v\}$; the only model of a reduct $P^{N[I,J]}$ is $N[I, J]$;*
4. *each model M of a reduct $P^{O[J^*]}$ satisfies the following properties:*
   (a) for each $y \in Y$ such that $y \in O[J^*]$ and $\overline{y} \notin O[J^*]$, if $w \in M$, then $y \in M$;
   (b) for each $y \in Y$ such that $\overline{y} \in O[J^*]$ and $y \notin O[J^*]$, if $w \in M$, then $\overline{y} \in M$;
   (c) if $(Y \cup \overline{Y}) \cap M \neq \emptyset$, then $w \in M$;
   (d) if there is a clause $l_{i,1} \vee \cdots \vee l_{i,m_i}$ of ϕ such that $\{\bar{l}_{i,1}, \ldots, \bar{l}_{i,m_i}\} \subseteq M$, then $v \in M$;
   (e) if there is an $x \in X$ such that $\{x, \overline{x}\} \subseteq M$, or there is a $y \in Y$ such that $\{y, \overline{y}\} \subseteq M$, or $\{v, w\} \subseteq M$, then it must hold that $X \cup \overline{X} \cup \{v, w\} \subseteq M$.

Lemma 3. *For any QBF $\Phi = \forall X \exists Y \phi$ with ϕ in CNF, a Φ-norm-reduction is super-coherent iff Φ is true.*

Proof. Suppose that Φ is false. Hence, there exists an $\mathcal{I} \subseteq X$ such that, for all $J \subseteq Y$, $\mathcal{I} \cup J \not\models \phi$. Now, let P be any Φ-norm-reduction and $F_{\mathcal{I}} = \mathcal{I} \cup \overline{(X \setminus \mathcal{I})}$. We show that $AS(P \cup F_{\mathcal{I}}) = \emptyset$, thus P is not super-coherent. Let $\mathcal{M}$ be a model of $P \cup F_{\mathcal{I}}$. Since

P is a Φ-norm-reduction, the only candidates for $\mathcal{M}$ are $O[J^*]$ for some $J \subseteq Y$ and J^* such that $J \cup \overline{(Y \setminus J)} \subseteq J^* \subseteq Y \cup \overline{Y}$, $M[\mathcal{I}]$, and $N[\mathcal{I}, J']$, where $J' \subseteq Y$ satisfies $\mathcal{I} \cup J' \models \phi$. However, from our assumption (for all $J \subseteq Y$, $\mathcal{I} \cup J \not\models \phi$), no such $N[\mathcal{I}, J']$ exists. Thus, it remains to consider $O[J^*]$ and $M[\mathcal{I}]$. By the properties of Φ-norm-reductions, $M[\mathcal{I}] \setminus \{v\}$ is a model of $P^{M[\mathcal{I}]}$, and hence $M[\mathcal{I}] \setminus \{v\}$ is also a model of $P^{M[\mathcal{I}]} \cup F_{\mathcal{I}} = (P \cup F_{\mathcal{I}})^{M[\mathcal{I}]}$. Thus, $M[\mathcal{I}]$ is not an answer set of $P \cup F_{\mathcal{I}}$. On the other hand, it can be checked that $M[\mathcal{I}] \setminus \{v\}$ is a model of $P^{O[J^*]} \cup F_{\mathcal{I}} = (P \cup F_{\mathcal{I}})^{O[J^*]}$, for any $O[J^*]$, and so none of these $O[J^*]$ are answer sets of $P \cup F_{\mathcal{I}}$ either. Since this means that no model of $P \cup F_{\mathcal{I}}$ is an answer set, we conclude $AS(P \cup F_{\mathcal{I}}) = \emptyset$, and hence P is not super-coherent.

Suppose that Φ is true. It is sufficient to show that, for each $F \subseteq U$, $AS(P \cup F) \neq \emptyset$. We distinguish the following cases for $F \subseteq U$:

$F \subseteq I \cup \overline{(X \setminus I)} \cup \{v\}$ for some $I \subseteq X$: If $v \in F$, then $M[I]$ is the unique model of $P^{M[I]} \cup F = (P \cup F)^{M[I]}$, and thus $M[I] \in AS(P \cup F)$. Otherwise, if $v \notin F$, since Φ is true, there exists a $J \subseteq Y$ such that $I \cup J \models \phi$. Thus, $N[I, J]$ is a model of $P \cup F$, and since no subset of $N[I, J]$ is a model of $(P \cup F)^{N[I,J]}$ (by property 3 of Φ-norm-reductions), $N[I, J] \in AS(P \cup F)$.

$I \cup \overline{(X \setminus I)} \subset F \subseteq N[I, J]$ for some $I \subseteq X$ and $J \subseteq Y$ such that $I \cup J \models \phi$: In this case $N[I, J]$ is a model of $P \cup F$ and, by property 3 of Φ-norm-reductions, $N[I, J]$ is also the unique model of $P^{N[I,J]} \cup F = (P \cup F)^{N[I,J]}$.

$I \cup \overline{(X \setminus I)} \subset F \subseteq N[I, J]$ for some $I \subseteq X$ and $J \subseteq Y$ such that $I \cup J \not\models \phi$: We shall show that $O[J]$ is an answer set of $P \cup F$ in this case. Let M be a model of $P^{O[J]} \cup F = (P \cup F)^{O[J]}$. Since $I \cup \overline{(X \setminus I)} \subset F \subseteq N[I, J]$, either $w \in F$ or $(Y \cup \overline{Y}) \cap F \neq \emptyset$. Clearly, $F \subseteq M$ and so $w \in M$ in the first case. Note that $w \in M$ holds also in the second case because of property 4(c) of Φ-norm-reductions. Thus, as a consequence of properties 4(a) and 4(b) of Φ-norm-reductions, $J \cup \overline{(Y \setminus J)} \subseteq M$ holds. Since $I \cup J \not\models \phi$ and because of property 4(d) of Φ-norm-reductions, $v \in M$ holds. Finally, since $\{v, w\} \subseteq M$ and because of property 4(e) of Φ-norm-reductions, $X \cup \overline{X} \subseteq M$ holds and, thus, $M = O[I]$.

In all other cases, either $\{v, w\} \subseteq F$, or there is an $x \in X$ such that $\{x, \overline{x}\} \subseteq F$, or there is a $y \in Y$ such that $\{y, \overline{y}\} \subseteq F$. We shall show that in such cases there is an $O[J^*]$ which is an answer set of $P \cup F$. Let $O[J^*]$ be such that $J^* = F \cap (Y \cup \overline{Y})$ and let M be a model of $P^{O[J^*]} \cup F = (P \cup F)^{O[J^*]}$ such that $M \subseteq O[J^*]$. We shall show that $O[J^*] \subseteq M$ holds, which would imply that $O[J^*] = M$ is an answer set of $P \cup F$. Clearly, $F \subseteq M$ holds. By property 4(e) of Φ-norm-reductions, $X \cup \overline{X} \cup \{v, w\} \subseteq M$ holds. Thus, by property 4(a) of Φ-norm-reductions and because $w \in M$, it holds that $y \in M$ for each $y \in Y$ such that $y \in O[J^*]$ and $\overline{y} \notin O[J^*]$. Similarly, by property 4(b) of Φ-norm-reductions and because $w \in M$, it holds that $\overline{y} \in M$ for each $y \in Y$ such that $\overline{y} \in O[J^*]$ and $y \notin O[J^*]$. Moreover, for all $y \in Y$ such that $\{y, \overline{y}\} \subseteq O[J^*]$, it holds that $\{y, \overline{y}\} \subseteq F \subseteq M$. Therefore, $O[J^*] \subseteq M$ holds and, consequently, $O[J^*] \in AS(P \cup F)$.

So in each of these cases $AS(P \cup F) \neq \emptyset$ and since these cases cover all possible $F \subseteq U$, we obtain that P is supercoherent.

Summarizing, we have shown that Φ being false implies that any Φ-norm-reduction P is not super-coherent, while Φ being true implies that any Φ-norm-reduction is super-coherent, which proves the lemma.

□

Definition 5. *For any QBF $\Phi = \forall X \exists Y \phi$ with $\phi = \bigwedge_{i=1}^{n} l_{i,1} \vee \cdots \vee l_{i,m_i}$ in CNF, we define*

$$N_\Phi = \{x \leftarrow not\ \overline{x};\ \overline{x} \leftarrow not\ x \mid x \in X\} \cup \tag{6}$$

$$\{y \leftarrow not\ \overline{y}, w;\ \overline{y} \leftarrow not\ y, w;\ w \leftarrow y;\ w \leftarrow \overline{y} \mid y \in Y\} \cup \tag{7}$$

$$\{z \leftarrow v, w;\ z \leftarrow x, \overline{x};\ z \leftarrow y, \overline{y} \mid z \in X \cup \overline{X} \cup \{v, w\}, x \in X, y \in Y\} \cup \tag{8}$$

$$\{v \leftarrow \bar{l}_{i,1}, \ldots, \bar{l}_{i,m_i} \mid 1 \leq i \leq n\} \cup \tag{9}$$

$$\{w \leftarrow not\ v\}. \tag{10}$$

Again, the program from the above definition can be constructed in polynomial time in the size of the reduced QBF. To conclude the proof, it is thus sufficient to show the following relation.

Lemma 4. *For any QBF $\Phi = \forall X \exists Y \phi$ with ϕ in CNF, the program N_Φ is a Φ-norm-reduction.*

Proof. We shall first show that N_Φ has the requested models. Let M be a model of N_Φ. Let us consider the atoms v and w. Because of the rule $w \leftarrow not\ v$ in (10), three cases are possible:

1. $\{v, w\} \subseteq M$. Thus, $X \cup \overline{X} \subseteq M$ holds because of (8). Moreover, there exists $J \subseteq Y$ such that $J \cup \overline{(Y \setminus J)} \subseteq M$ because of (7). Note that any other atom in U could belong to M. These are the models $O[J^*]$.
2. $v \in M$ and $w \notin M$. Thus, there exists $I \subseteq X$ such that $I \cup \overline{(X \setminus I)} \subseteq M$ because of (6). Moreover, no atoms in $Y \cup \overline{Y}$ belong to M because of (7) and $w \notin M$ by assumption. Thus, $M = M[I]$ in this case.
3. $v \notin M$ and $w \in M$. Thus, there exist $I \subseteq X$ and $J \subseteq Y$ such that $I \cup \overline{(X \setminus I)} \subseteq M$ and $J \cup \overline{(Y \setminus J)} \subseteq M$ because of (6) and (7). Hence, in this case $M = N[I, J]$ and, because of (9), it holds that $I \cup J \models \phi$.

Let us consider a reduct $P^{M[I]}$ and one of its models $M \subseteq M[I]$. First of all, note that $P^{M[I]}$ contains a fact for each atom in $I \cup \overline{(X \setminus I)}$. Thus, $I \cup \overline{(X \setminus I)} \subseteq M$ holds. Note also that, since each clause of ϕ contains at least one variable from Y, all of the rules of (9) have at least one false body literal. Hence, either $M = M[I]$ or $M = M[I] \setminus \{v\}$, as required by Φ-norm-reductions.

For a reduct $P^{N[I,J]}$ such that $I \cup J \models \phi$ it is enough to note that $P^{N[I,J]}$ contains a fact for each atom of $N[I, J]$.

Let us consider a reduct $P^{O[J^*]}$ and one of its models $M \subseteq O[J^*]$. The first observation is that for each $y \in Y$ such that $y \in O[J^*]$ and $\overline{y} \notin O[J^*]$, the reduct $P^{O[J^*]}$ contains a rule of the form $y \leftarrow w$ (obtained by some rule in (7)). Similarly, for each

$y \in Y$ such that $\overline{y} \in O[J^*]$ and $y \notin O[J^*]$, the reduct $P^{O[J^*]}$ contains a rule of the form $\overline{y} \leftarrow w$ (obtained by some rule in (7)). Hence, M must satisfy properties 4(a) and 4(b) of Φ-norm-reductions. Property 4(c) is a consequence of (7), property 4(d) follows from (9) and, finally, property 4(e) must hold because of (8). □

Note that the program from Definition 5 does not contain constraints. As a consequence, the Π_2^P-hardness result presented in this section also holds if we only consider normal ASP programs without constraints.

6 Some Implications

In [22] the following problem has been studied under the name "uniform equivalence with projection:"

> Given two programs P and Q, and two sets A, B of atoms, $P \equiv_B^A Q$ iff for each set $F \subseteq A$ of facts, $\{I \cap B \mid I \in AS(P \cup F)\} = \{I \cap B \mid I \in AS(Q \cup F)\}$.

Let us call A the context alphabet and B the projection alphabet. As is easily verified the following relation holds.

Proposition 3. *A program P over atoms U is super-coherent iff $P \equiv_\emptyset^U Q$, where Q is an arbitrary definite Horn program.*

Note that $P \equiv_\emptyset^U Q$ means $\{I \cap \emptyset \mid I \in AS(P \cup F)\} = \{I \cap \emptyset \mid I \in AS(Q \cup F)\}$ for all sets $F \subseteq U$. Now observe that for any $F \subseteq U$, both of these sets are either empty or contain the empty set, depending on whether the programs (extended by F) have answer sets.

$$\{I \cap \emptyset \mid I \in AS(P \cup F)\} = \begin{cases} \emptyset & \text{iff } AS(P \cup F) = \emptyset \\ \{\emptyset\} & \text{iff } AS(P \cup F) \neq \emptyset \end{cases}$$

$$\{I \cap \emptyset \mid I \in AS(Q \cup F)\} = \begin{cases} \emptyset & \text{iff } AS(Q \cup F) = \emptyset \\ \{\emptyset\} & \text{iff } AS(Q \cup F) \neq \emptyset \end{cases}$$

If Q is a definite Horn program, then $AS(Q \cup F) \neq \emptyset$ for all $F \subseteq U$, and therefore the statement of Proposition 3 becomes equivalent to checking whether $AS(P \cup F) \neq \emptyset$ for all $F \subseteq U$, and thus whether P is super-coherent.

In [22], the complexity of the problem of deciding uniform equivalence with projection has also been investigated, reporting Π_3^P-completeness for disjunctive programs and Π_2^P-completeness for normal programs. However, these hardness results use bound context alphabets $A \subset U$ (where U are all atoms from the compared programs). Our results thus strengthen the observations in [22]. Using Proposition 3 and the main results in this paper, we obtain Π_3^P-hardness (resp. Π_2^P-hardness in the case of normal programs) for uniform equivalence with projection even for the particular parameterization where the context alphabet is unrestricted, the projection set is empty, and one of the compared programs are of a very simple structure (in fact, even the empty program is sufficient for Q in Proposition 3).

7 Conclusion

Many recent advances in ASP rely on the adaptions of technologies from other areas. One important example is the Magic Set method, which stems from the area of databases and is used in state-of-the-art ASP grounders. Recent work showed that a particular variant of this technique only applies to a certain class of programs called super-coherent [2]. Super-coherent programs are those which possess at least one answer set, no matter which set of facts is added to them. We believe that this class of programs is interesting of its own (for instance, since there is a certain relation to some problems in equivalence checking) and thus studied here the exact complexity of recognizing the property of super-coherence for disjunctive and normal programs. Our results show that the problems are surprisingly hard, viz. complete for Π_3^P and resp. Π_2^P. One direction of future work is to search for methods to turn arbitrary programs into super-coherent ones with minimal changes. Our proofs might provide valuable foundations for such methods. That said, for using super-coherent programs efficiently for applications, we believe that an approach that uses a methodology different from the currently prevailing "Guess&Check" or "Generate&Test" should be developed.

References

1. Alviano, M., Faber, W.: Dynamic magic sets for super-consistent answer set programs. In: Balduccini, M., Woltran, S. (eds.) Proceedings of the 3rd Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2010) (Jul 2010)
2. Alviano, M., Faber, W.: Dynamic magic sets and super-coherent answer set programs. AI Communications 24(2), 125–145 (2011)
3. Alviano, M., Faber, W., Greco, G., Leone, N.: Magic sets for disjunctive datalog programs. Tech. Rep. 09/2009, Dipartimento di Matematica, Università della Calabria, Italy (2009), `http://www.wfaber.com/research/papers/TRMAT092009.pdf`
4. Bancilhon, F., Maier, D., Sagiv, Y., Ullman, J.D.: Magic Sets and Other Strange Ways to Implement Logic Programs. In: Proceedings of the Fifth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems. pp. 1–15. Cambridge, Massachusetts (1986)
5. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003)
6. Beeri, C., Ramakrishnan, R.: On the power of magic. Journal of Logic Programming 10(1–4), 255–259 (1991)
7. Cumbo, C., Faber, W., Greco, G., Leone, N.: Enhancing the magic-set method for disjunctive datalog programs. In: Proceedings of the the 20th International Conference on Logic Programming – ICLP'04. LNCS, vol. 3132, pp. 371–385 (2004)
8. Drescher, C., Gebser, M., Grote, T., Kaufmann, B., König, A., Ostrowski, M., Schaub, T.: Conflict-Driven Disjunctive Answer Set Solving. In: Brewka, G., Lang, J. (eds.) Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning (KR 2008). pp. 422–432. AAAI Press, Sydney, Australia (2008)
9. Eiter, T., Fink, M., Tompits, H., Woltran, S.: On eliminating disjunctions in stable logic programming. In: Proceedings of the 9th International Conference on Principles of Knowledge Representation and Reasoning (KR 2004). pp. 447–458. AAAI Press (2004)
10. Eiter, T., Fink, M., Woltran, S.: Semantical Characterizations and Complexity of Equivalences in Stable Logic Programming. ACM Transactions on Computational Logic 8(3), 1–53 (2007)

11. Eiter, T., Gottlob, G.: On the computational cost of disjunctive logic programming: Propositional case. Annals of Mathematics and Artificial Intelligence 15(3-4), 289–323 (1995)
12. Eiter, T., Gottlob, G., Mannila, H.: Disjunctive Datalog. ACM Transactions on Database Systems 22(3), 364–418 (Sep 1997)
13. Eiter, T., Tompits, H., Woltran, S.: On Solution Correspondences in Answer Set Programming. In: Kaelbling, L.P., Saffiotti, A. (eds.) Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05). pp. 97–102. Professional Book Center (2005)
14. Greco, S.: Binding Propagation Techniques for the Optimization of Bound Disjunctive Queries. IEEE Transactions on Knowledge and Data Engineering 15(2), 368–385 (March/April 2003)
15. Janhunen, T., Niemelä, I., Seipel, D., Simons, P., You, J.H.: Unfolding Partiality and Disjunctions in Stable Model Semantics. ACM Transactions on Computational Logic 7(1), 1–37 (Jan 2006)
16. Leone, N., Gottlob, G., Rosati, R., Eiter, T., Faber, W., Fink, M., Greco, G., Ianni, G., Kałka, E., Lembo, D., Lenzerini, M., Lio, V., Nowicki, B., Ruzzi, M., Staniszkis, W., Terracina, G.: The INFOMIX System for Advanced Integration of Incomplete and Inconsistent Data. In: Proceedings of the 24th ACM SIGMOD International Conference on Management of Data (SIGMOD 2005). pp. 915–917. ACM Press, Baltimore, Maryland, USA (Jun 2005)
17. Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S., Scarcello, F.: The DLV System for Knowledge Representation and Reasoning. ACM Transactions on Computational Logic 7(3), 499–562 (Jul 2006)
18. Lierler, Y.: Disjunctive Answer Set Programming via Satisfiability. In: Baral, C., Greco, G., Leone, N., Terracina, G. (eds.) Logic Programming and Nonmonotonic Reasoning — 8th International Conference, LPNMR'05, Diamante, Italy, September 2005, Proceedings. LNCS, vol. 3662, pp. 447–451. Springer Verlag (Sep 2005)
19. Lifschitz, V., Turner, H.: Splitting a Logic Program. In: Van Hentenryck, P. (ed.) Proceedings of the 11th International Conference on Logic Programming (ICLP'94). pp. 23–37. MIT Press, Santa Margherita Ligure, Italy (Jun 1994)
20. Manna, M., Ruffolo, M., Oro, E., Alviano, M., Leone, N.: The HiLeX System for Semantic Information Extraction. Transactions on Large-Scale Data and Knowledge-Centered Systems (2011), to appear
21. Manna, M., Scarcello, F., Leone, N.: On the complexity of regular-grammars with integer attributes. Journal of Computer and System Sciences (JCSS) 77(2), 393–421 (2011)
22. Oetsch, J., Tompits, H., Woltran, S.: Facts do not cease to exist because they are ignored: Relativised uniform equivalence with answer-set projection. In: Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI'07). pp. 458–464. AAAI Press (2007)
23. Papadimitriou, C.H., Yannakakis, M.: Tie-breaking semantics and structural totality. Journal of Computer and System Sciences 54(1), 48–60 (1997)
24. Ricca, F., Alviano, M., Dimasi, A., Grasso, G., Ielpa, S.M., Iiritano, S., Manna, M., Leone, N.: A Logic-Based System for e-Tourism. Fundamenta Informaticae 105(1–2), 35–55 (2010)
25. Ricca, F., Grasso, G., Alviano, M., Manna, M., Lio, V., Iiritano, S., Leone, N.: Team-building with answer set programming in the Gioia-Tauro seaport. Theory and Practice of Logic Programming (2011), to appear, doi:10.1017/S147106841100007X
26. Ullman, J.D.: Principles of Database and Knowledge Base Systems. Computer Science Press (1989)