TLP **14** (3): 339–361, 2014. © Cambridge University Press 2013
doi:10.1017/S147106841300001X First published online 9 August 2013

*Complexity of super-coherence problems in ASP**

MARIO ALVIANO and WOLFGANG FABER
University of Calabria, Rende (CS) 87036, Italy
(*e-mail:* {alviano,faber}@mat.unical.it)

STEFAN WOLTRAN
Vienna University of Technology, Vienna 1040, Austria
(*e-mail:* woltran@dbai.tuwien.ac.at)

submitted 14 December 2011; revised 8 August 2012; accepted 19 December 2012

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 Dynamic Magic Set, 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 a recent work, a large fragment of ASP programs, referred to as *super-coherent programs*, has been identified, for which Dynamic Magic Set is correct. The fragment contains all programs which possess at least one answer set, no matter which set of facts is added to them. Two open question remained: How complex is it to determine whether a given program is super-coherent? Does the restriction to super-coherent programs limit the problems that can be solved? Especially the first 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. Concerning the second question, we show that all relevant ASP reasoning tasks can be transformed into tasks over super-coherent programs, although this transformation is more of theoretical than practical interest.

KEYWORDS: Answer Set Programming (ASP), complexity analysis, coherence, uniform equivalence, foundations

1 Introduction

Answer Set Programming (ASP) is a powerful formalism for knowledge representation and common sense reasoning (Baral 2003). Allowing disjunction in rule heads

* Preliminary versions of this paper have been presented at the International Conference on Logic Programming (ICLP) workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP) and at the Convegno Italiano di Logica Computazionale (CILC).

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

and nonmonotonic negation in bodies, ASP can express every query belonging to the complexity class Σ_2^P (NP^{NP}). Encouraged by the availability of efficient inference engines, such as DLV (Leone *et al.* 2006), GnT (Janhunen *et al.* 2006), Cmodels (Lierler 2005), or ClaspD (Drescher *et al.* 2008), ASP has found several practical applications in various domains, including data integration (Leone *et al.* 2005), semantic-based information extraction (Manna *et al.* 2011; Manna *et al.* 2012), e-tourism (Ricca *et al.* 2010), workforce management (Ricca *et al.* 2012), 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 (Bancilhon *et al.* 1986; Ullman 1989; Beeri and Ramakrishnan 1991) have recently achieved a lot of attention. Following some earlier work (Greco 2003; Cumbo *et al.* 2004), an adapted method called *Dynamic Magic Set* (DMS) has been proposed for ASP in Alviano *et al.* (2012). However, this technique has a caveat because it is not correct (in the sense of being query-equivalent) for all ASP programs. In recent works (Alviano and Faber 2010, 2011), a large fragment of ASP programs, referred to as *super-coherent programs* (ASP^{sc}), has been identified, for which DMS can be proved to be correct. Formally, a program is super-coherent, if it is coherent (i.e., possesses at least one answer set), no matter which input (given as a set of facts) is added to the 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 ASP^{sc}? It turns out that the precise complexity is rather difficult to establish. Some bounds have been given in Alviano and Faber (2011), in particular showing decidability, but especially hardness results seemed quite hard to obtain. In particular, the following question remained unanswered: Is it possible to implement an efficient algorithm for testing super-coherence of a program, to decide for example whether DMS has to be applied or not? In this paper we provide a negative answer to this question, proving that deciding whether a propositional program is super-coherent is complete for the third level of the polynomial hierarchy in the general case, and for the second level for normal programs. As the complexity of query answering is located on lower levels of the polynomial hierarchy, our results show that implementing a sound and complete super-coherence check in a query optimization setting does not provide an approach for improving such systems.

While our main motivation for studying ASP^{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 Lifschitz and Turner (1994), if a top part of a split program is an ASP^{sc} program, then any answer set of the bottom part will give rise to at least one answer

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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. Papadimitriou and Yannakakis 1997). 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 ASP^{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 (Eiter *et al.* 2005, 2007; Oetsch *et al.* 2007). 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 super-coherent programs form a strict subset of all ASP programs, another important question arises: Does the restriction to super-coherent programs limit the problems that can be solved by them? In this paper, we show that this is not the case by embedding all relevant reasoning tasks over ASP (testing answer set existence, query answering, answer set computation) into reasoning tasks over ASP^{sc}. We also show that all reasoning tasks over normal (non-disjunctive) ASP can be embedded into tasks over normal ASP^{sc}. These results essentially demonstrate that ASP^{sc} is sufficient to encode any problem that can be solved by full ASP, and is therefore in a sense "complete." However, we would like to note that these embeddings were designed for answering this theoretical question, and might lead to significant overhead when evaluated with ASP solvers. We therefore do not advocate to use them in practical settings, and finding efficient embeddings is a challenging topic for future research.

To summarize, the main contributions of the paper are as follows:

- We show that recognizing super-coherence for disjunctive and normal programs is complete for classes Π_3^P and Π_2^P, respectively, thus more complex than the common ASP reasoning tasks.
- We provide a transformation of reasoning tasks over general programs into tasks over super-coherent programs, showing that the restriction to super-coherent programs does not curtail expressive power.
- We also briefly discuss the relation between checking for super-coherence and testing equivalence between programs where we make use of our results to sharpen complexity results due to Oetsch *et al.* (2007).

In order to focus on the essentials of these problems, in this paper we deal with propositional programs, and conjecture that the results can be extended to the non-propositional case by using complexity upgrading techniques as presented in

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Eiter *et al.* (1997) and Gottlob *et al.* (1999), arriving at completeness for classes co-NEXP$^{\text{NP}}$ and co-NEXP$^{\Sigma_2^p}$, respectively.

The remainder of this paper is organized as follows. In Section 2 we first define some terminology needed later on. In Section 3 we formulate the complexity problems that we analyze, and state our main results. The proofs for these problems are presented in Section 3.1 for disjunctive programs, and in Section 3.2 for normal programs. In Section 4 we show "completeness" of ASP$^{\text{sc}}$ via simulating reasoning tasks over ASP by tasks over ASP$^{\text{sc}}$. In Section 5 we briefly discuss the relation to equivalence problems before concluding the work in Section 6.

2 Preliminaries

In this paper we consider propositional programs, so an atom p is a member of a countable set $\mathscr{U}$. 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

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

where $p_1, \ldots, p_n, q_1, \ldots, q_m$ are atoms and $n \geqslant 0$, $m \geqslant j \geqslant 0$. The disjunction $p_1 \vee \cdots \vee p_n$ is the *head* of r, while the conjunction $q_1, \ldots, q_j, \mathit{not}\ q_{j+1}, \ldots, \mathit{not}\ q_m$ 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 (cf. Dung 1992; You and Yuan 1994) and stratified (cf. Apt *et al.* 1988) 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

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

can be replaced by the following equivalent rule:

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

where co is a fresh atom (i.e., an atom that does not occur elsewhere in the program). We also require the notion of *head-cycle-free* programs (cf. Ben-Eliyahu and Dechter 1994): A program P is called head-cycle free if no different head atoms in a rule positively depend mutually on each other.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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 (Gelfond and Lifschitz 1991). 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., Eiter *et al.* 2004, 2005):

(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 both I and J satisfy P^K, then also $(I \cap J)$ satisfies P^K.

By a query in this paper we refer to an atom, negative and conjunctive queries can be simulated by adding appropriate rules to the associated program. A query q is *bravely* true for a program P if and only if $q \in A$ for some $A \in AS(P)$. A query q is *cautiously* true for a program P if and only if $q \in A$ for all $A \in AS(P)$.

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

Definition 1 (ASP^{sc} programs; Alviano and Faber 2010, 2011)
A program P is *super-coherent* if, for every set of facts F, $AS(P \cup F) \neq \emptyset$. Let ASP^{sc} denote the set of all super-coherent programs.

Note that ASP^{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 ASP^{sc} (cf. Alviano and Faber 2011).

3 Complexity of checking super-coherence

In this section 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.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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 already holds 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 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 with, we give a straightforward 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, $AS(P \cup F') \neq \emptyset$. 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 (Lifschitz and Turner 1994), as the atoms in F'' are only defined by facts not occurring in $P \cup F'$. □

Our main results are stated below. 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.

3.1 Proof of Theorem 1

Membership follows by the following straightforward 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 (Eiter and Gottlob 1995). This shows Π_3^P-membership.

For the hardness we reduce the Π_3^P-complete problem of deciding whether quantified Boolean formulae (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 disjunctive normal form (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

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Z. More precisely, we shall construct for each such QBF Φ a program P_Φ such that Φ is true if and only if 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 if and only if 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 $\overline{S} = \{\overline{s} \mid s \in S\}$ ($S \in \{X, Y, Z\}$) 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[1] 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 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 models, 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.

[1] 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.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

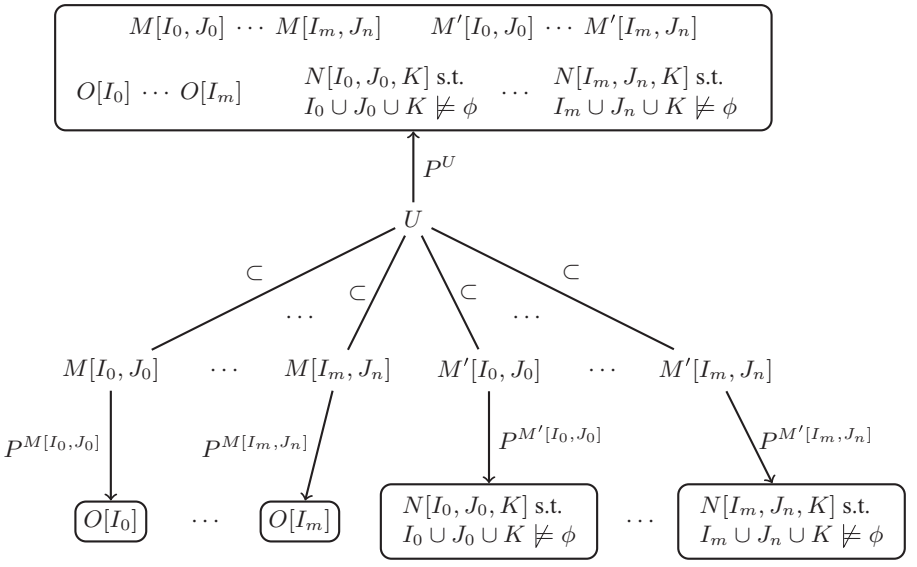


Fig. 1. Models and reduct "countermodels" of Φ-reductions.

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

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

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

In each of the cases we have obtained $\mathscr{M} \notin AS(P \cup F_\mathscr{I})$, hence $AS(P \cup F_\mathscr{I}) = \emptyset$ 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 $\{s, \bar{s}\} \subseteq F$ 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

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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 (being a proper subset of $M[I,J]$) 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 $P \cup F$. Note that the case $F \subseteq M'[I,J]$ with $w \notin F$ is already taken care of since for such F we have $F \subseteq M[I,J]$ as well.

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 super-coherent.

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

$$P_\Phi = \{x \vee \overline{x} \leftarrow;\ u \leftarrow x, \overline{x};\ w \leftarrow x, \overline{x};\ x \leftarrow u, w;\ \overline{x} \leftarrow u, w \mid x \in X\} \cup \tag{1}$$

$$\{y \vee \overline{y} \leftarrow v;\ u \leftarrow y, \overline{y};\ w \leftarrow y, \overline{y};\ y \leftarrow u, w;$$
$$\overline{y} \leftarrow u, w;\ v \leftarrow y;\ v \leftarrow \overline{y} \mid y \in Y\} \cup \tag{2}$$

$$\{z \vee \overline{z} \leftarrow v;\ u \leftarrow z, not\ w;\ u \leftarrow \overline{z}, not\ w;\ v \leftarrow z;\ v \leftarrow \overline{z};$$
$$z \leftarrow w;\ \overline{z} \leftarrow w;\ z \leftarrow u;\ \overline{z} \leftarrow u;\ w \vee u \leftarrow z, \overline{z} \mid z \in Z\} \cup \tag{3}$$

$$\{w \vee u \leftarrow l_{i,1}, \ldots, l_{i,m_i} \mid 1 \leqslant i \leqslant n\} \tag{4}$$

$$\{v \leftarrow w;\ v \leftarrow u;\ v \leftarrow not\ u\}. \tag{5}$$

The above program contains atoms associated with literals in Φ and three auxiliary atoms u, v, w. Intuitively, truth values for variables in X are guessed by rules in (1), and truth values for variables in $Y \cup Z$ are guessed by rules in (2) and (3), provided that v is true. Moreover, rules in (1) and (2) derive all atoms in $X \cup \overline{X} \cup Y \cup \overline{Y} \cup \{u, w\}$ whenever both u and w are true, or in case an inconsistent assignment for some propositional variable in $X \cup Y$ is forced by the addition of a set of facts to the program. On the other hand, all atoms associated with variables Z are inferred by rules in (3) if one of u and w is true. Atoms u and w can be inferred, for instance,

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

by some rule in (4) whenever truth values of atoms of P_Φ represent a satisfying assignment for ϕ. Furthermore, rules in (5) are such that every model of P_Φ contains v, and any answer set of P_Φ must also contain w. We finally observe that w can be derived by some rule in (4) in the program reduct, which is the case if truth values of atoms in $X \cup \overline{X} \cup Y \cup \overline{Y}$ represent a consistent assignment for propositional variables in $X \cup Y$ satisfying $\forall Z \phi$.

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), $I \cup \overline{(X \setminus I)} \cup J \cup \overline{(Y \setminus J)} \subseteq M$ 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 $N \nsubseteq M[I,J]$, resp. $N \nsubseteq M'[I,J]$, $N = U$ 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 $I \cup J \cup K \not\models \phi$. 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. Now consider 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), $O[I] \subseteq W$

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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.

3.2 Proof of Theorem 2

Membership follows by the straightforward nondeterministic algorithm for the complementary problem presented in the previous section. We have just to note that a *co*-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 conjunctive normal form (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 with 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*:

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

(1) P is given over atoms $U = X \cup Y \cup \overline{X} \cup \overline{Y} \cup \{v, w\}$, where all atoms in sets $\overline{S} = \{\overline{s} \mid s \in S\}$ ($S \in \{X, Y, Z\}$) 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 $\{\overline{l}_{i,1}, \ldots, \overline{l}_{i,m_i}\} \subseteq M$, 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 $X \cup \overline{X} \cup \{v, w\} \subseteq M$.

Similarly as in the previous section, the models of the reducts $P^{O[J^*]}$ as given in Item (4) are not specified for particular purposes, but are required to allow for a realization via normal programs taking the set of models specified in Items (2) and (3) as well as properties (P1) and (P2) from Section 2 into account.

Lemma 3
For any QBF $\Phi = \forall X \exists Y \phi$ with ϕ in CNF, a Φ-norm-reduction is super-coherent if and only if 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 $P \cup F_{\mathcal{I}}$. Due to property (P1), $M[\mathcal{I}] \setminus \{v\}$ is also 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.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

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 a 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 in this case $O[J]$ is an answer set of $P \cup F$. 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 $w \in M$ also holds in the second case because of property (4c) of Φ-norm-reductions. Thus, as a consequence of properties (4a) and (4b) of Φ-norm-reductions, $J \cup \overline{(Y \setminus J)} \subseteq M$ holds. Since $I \cup J \not\models \phi$, and because of property (4d) of Φ-norm-reductions, $v \in M$ holds. Finally, since $\{v,w\} \subseteq M$, and because of property (4e) of Φ-norm-reductions, $X \cup \overline{X} \subseteq M$ holds and, thus, $M = O[J]$.

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 (4e) of Φ-norm-reductions, $X \cup \overline{X} \cup \{v,w\} \subseteq M$ holds. Thus, by property (4a) 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 (4b) 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 super-coherent.

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. □

We have now to show that for any QBF of the desired form, a Φ-norm-reduction can be obtained in polynomial time (w.r.t. the size of Φ).

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

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

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

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

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

$$x \in X, y \in Y\} \cup \tag{8}$$

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

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

The above program contains atoms associated with literals in Φ and two auxiliary atoms v, w. Intuitively, v is derived by some rule in (9) whenever some clause of ϕ is violated. Otherwise, if v is not derived, truth of w is inferred by default because of rule (10). Moreover, truth values for variables in X are guessed by rules in (6), and truth values for variables in Y are guessed by rules in (7), provided that w is true. Finally, rules in (8) derive all atoms in $X \cup \overline{X} \cup \{v, w\}$ whenever both v and w are true, or in case an inconsistent assignment for some propositional variable is forced by the addition of a set of facts to the program. It turns out that any answer set for such a program is such that truth values of atoms in $X \cup \overline{X} \cup Y \cup \overline{Y}$ represent a consistent assignment for propositional variables in $X \cup Y$ satisfying ϕ.

Again, note that 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

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

of the form $\overline{y} \leftarrow w$ (obtained by some rule in (7)). Hence, M must satisfy properties (4a) and (4b) of Φ-norm-reductions. Property (4c) is a consequence of (7), property (4d) follows from (9), and finally property (4e) 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.

4 Simulating general answer set programs using super-coherent programs

Since ASP^{sc} programs are a proper subset of ASP programs, a natural question to ask is whether the restriction to ASP^{sc} programs limits the range of problems that can be solved. In this section we show that this is not the case, i.e., all problems solvable in ASP can be encoded in ASP^{sc}, and thus benefit from the optimization potential provided by DMS. Although these results are interesting from a theoretical point of view, we do not suggest that they have to be employed in practice.

Often ASP is associated with the decision problem of whether a program P has any answer set (the coherence problem), that is, testing whether $AS(P) \neq \emptyset$. Of course, the coherence problem becomes trivial for ASP^{sc} programs. Another type of decision problem associated with ASP is query answering, and indeed in this section we show that by using query answering it is possible to simulate both coherence problems and query answering of ASP by query answering over ASP^{sc} programs. Using the same construction, we also show that answer set computation problems for ASP programs can be simulated by appropriate ASP^{sc} programs. While these constructions always yield disjunctive programs, we also show how to adapt them in order to yield normal ASP^{sc} programs when starting from normal programs.

We start by assigning to each disjunctive ASP program a corresponding super-coherent program. In order to simplify the presentation, and without loss of generality, in this section we will only consider programs without constraints.

Definition 6
Let P be a program the atoms of which belong to a countable set $\mathcal{U}$. We construct a new program P_{strat} by using atoms of the following set:

$$\mathcal{U}_{strat} = \mathcal{U} \cup \{\alpha^T \mid \alpha \in \mathcal{U}\} \cup \{\alpha^F \mid \alpha \in \mathcal{U}\} \cup \{fail\},$$

where α^T, α^F and $fail$ are fresh symbols not belonging to $\mathcal{U}$. Program P_{strat} contains the following rules:

- For each rule r of P, a rule r_{strat} such that
  - $H(r_{strat}) = H(r)$ and
  - $B(r_{strat}) = B^+(r) \cup \{\alpha^F \mid \alpha \in B^-(r)\}$.
- For each atom α in $\mathcal{U}$, three rules of the form

$$\alpha^T \vee \alpha^F \leftarrow \tag{11}$$

$$\alpha^T \leftarrow \alpha \tag{12}$$

$$fail \leftarrow \alpha^T,\ not\ \alpha. \tag{13}$$

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Intuitively, every rule r of P is replaced by a new rule r_{strat}, in which new atoms of the form α^F replace negative literals of r. Thus, our translation must guarantee that an atom α^F is true if and only if the associated atom α is false. In fact, this is achieved by means of rules of the forms (11)–(13):

- Form (11) guarantees that either α^T or α^F belongs to every answer set of P_{strat};
- form (12) assures that α^T belongs to every model of P_{strat} containing atom α; and
- form (13) derives *fail* if α^T is true but α is false, that is, if α^T is only supported by a rule of the form (11).

It is not difficult to prove that the program P_{strat} is super-coherent.

Lemma 5
Let P be a disjunctive program. Program P_{strat} is stratified and thus super-coherent.

Proof
All negative literals in P_{strat} are those in rules of the form (13), the head of which is *fail*, an atom not occurring elsewhere in P. □

Proving correspondence between answer sets of P and P_{strat} is slightly more difficult. To this aim, we first introduce some properties of the interpretations of P_{strat}.

Lemma 6
Let I be an interpretation for P_{strat} such that:

(1) for every $\alpha \in \mathscr{U}$, precisely one of α^T and α^F belongs to I;
(2) for every $\alpha \in \mathscr{U}$, $\alpha \in I$ if and only if $\alpha^T \in I$.

Then, for every rule r of P, the following relation is established:

$$B^+(r_{strat}) \subseteq I \iff B^+(r) \subseteq (I \cap \mathscr{U}) \text{ and } B^-(r) \cap (I \cap \mathscr{U}) = \emptyset. \tag{14}$$

Proof
By combining properties of I (items (1) and (2)), we have that $\alpha^F \in I$ if and only if $\alpha \notin I$. The claim thus follows by construction of P_{strat}. In fact, $B^+(r) = B^+(r_{strat}) \cap \mathscr{U}$ and $B^-(r) = \{\alpha \mid \alpha^F \in B^+(r_{strat}) \setminus \mathscr{U}\}$. □

We are now ready to formalize and prove the correspondence between answer sets of P and P_{strat}.

Theorem 3
Let P be a program and P_{strat} be the program obtained as described in Definition 6. The following relation holds:

$$AS(P) = \{M \cap \mathscr{U} \mid M \in AS(P_{strat}) \wedge \mathit{fail} \notin M\}.$$

Proof
($\supseteq$) Let M be an answer set of P_{strat} such that $\mathit{fail} \notin M$. We shall show that $M \cap \mathscr{U}$ is an answer set of P.

We start by observing that M has the properties required by Lemma 6:

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

- The first property is guaranteed by rules of the form (11) and because atoms of the form α^F occur as head atoms only in these rules.
- The second property is ensured by rules of the forms (12) and (13), combined with the assumption $fail \notin M$.

Therefore, relation (14) holds for M, which combined with the assumption that M is a model of P_{strat}, implies that $M \cap \mathscr{U}$ is a model of P. We next show that $M \cap \mathscr{U}$ is also a minimal model of the reduct $P^{(M \cap \mathscr{U})}$.

Let $(M \cap \mathscr{U}) \setminus \Delta$ be a model of $P^{(M \cap \mathscr{U})}$ for some set $\Delta \subseteq \mathscr{U}$. We next prove that $M \setminus \Delta$ is a model of P^M_{strat}, which implies that $\Delta = \emptyset$ since M is an answer set of P_{strat}. All of the rules of P^M_{strat} obtained from (11), (12), and (13) are satisfied by $M \setminus \Delta$: Rules of P^M_{strat} obtained from (11) and (12) remain equal, and since $\Delta \subseteq \mathscr{U}$ and their heads are not in $\mathscr{U}$, satisfaction by M implies satisfaction by $M \setminus \Delta$. Rules of P^M_{strat} obtained from (13) are such that $\alpha \notin M$, in addition since $fail \notin M$ also $fail \notin M \setminus \Delta$, and since the rule is satisfied by M, $\alpha^T \notin M$ and also $\alpha^T \notin M \setminus \Delta$. Every remaining rule $r_{strat} \in P_{strat}$ is such that $r \in P$. If $B^+(r_{strat}) \subseteq M \setminus \Delta \subseteq M$, we can apply (14) and conclude $B^-(r) \cap (M \cap \mathscr{U}) = \emptyset$, that is, a rule obtained from r by removing negative literals belongs to $P^{(M \cap \mathscr{U})}$. Moreover, we can conclude that $B^+(r) = B^+(r_{strat}) \cap \mathscr{U} \subseteq (M \cap \mathscr{U}) \setminus \Delta$, and thus $\emptyset \neq H(r) \cap (M \cap \mathscr{U}) \setminus \Delta = H(r_{strat}) \cap M \setminus \Delta$.

($\subseteq$) Let M be an answer set of P. We shall show that

$$M' = M \cup \{\alpha^T \mid \alpha \in M\} \cup \{\alpha^F \mid \alpha \in \mathscr{U} \setminus M\}$$

is an answer set of P_{strat}.

We first observe that relation (14) holds for M', which, combined with the assumption that M is a model of P, implies that M' is a model of P_{strat}. We can then show that M' is also a minimal model for the reduct $P^{M'}_{strat}$. In fact, we can show that every $N \subseteq M'$ modeling $P^{M'}_{strat}$ is such that $N = M'$ in two steps:

(1) $N \setminus \mathscr{U} = M' \setminus \mathscr{U}$. This follows by rules of the form (11) and by construction of M'. In fact, rules of the form (11) belongs to $P^{M'}_{strat}$, of which N is a model by assumption. For each $\alpha \in \mathscr{U}$, these rules enforce the presence of at least one of α^T and α^F in N. By construction, M' contains exactly one of α^T or α^F for each $\alpha \in \mathscr{U}$, and we thus conclude $N \setminus \mathscr{U} = M' \setminus \mathscr{U}$.

(2) $N \cap \mathscr{U} = M' \cap \mathscr{U}$. Note that $M' \cap \mathscr{U} = M$. Moreover, from the assumption $N \subseteq M'$, we have $N \cap \mathscr{U} \subseteq M' \cap \mathscr{U} = M$. Thus, it is enough to show that $N \cap \mathscr{U}$ is a model of P^M, because in this case $N \cap \mathscr{U} = M$ would be a consequence of the assumption $M \in AS(P)$.
In order to show that $N \cap \mathscr{U}$ is a model of P^M, let us consider a rule r' of P^M with $B^+(r') \subseteq N \cap \mathscr{U}$. Such a rule has been obtained from a rule r of P such that $B^-(r) \cap M = \emptyset$. Let us consider $r_{strat} \in P_{strat}$, recall that r_{strat} has been obtained from r by replacing negatively occurring atoms α by α^F. Clearly, $r_{strat} \in P^{M'}_{strat}$ because $B^-(r_{strat}) = \emptyset$ by construction. Moreover, since $B^+(r') \subseteq N \cap \mathscr{U}$, we get $B^+(r) \subseteq N \cap \mathscr{U}$ (recall that r' is the reduct of r with respect to M, thus $B^+(r) = B^+(r')$) and consequently $B^+(r_{strat}) \cap \mathscr{U} \subseteq N \cap \mathscr{U}$ (since $B^+(r_{strat}) \cap \mathscr{U} = B^+(r)$). Furthermore, $B^+(r_{strat}) \setminus \mathscr{U} = \{\alpha^F \mid \alpha \in B^-(r)\}$, and since $B^-(r) \cap M = \emptyset$, we get $B^+(r_{strat}) \setminus \mathscr{U} \subseteq M' \setminus \mathscr{U}$ (since $\alpha^F \in M'$ if and

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

only if $\alpha \notin M$); given $N \setminus \mathscr{U} = M' \setminus \mathscr{U}$, we also have $B^+(r_{strat}) \setminus \mathscr{U} \subseteq N \setminus \mathscr{U}$. In total, we obtain $B^+(r_{strat}) \subseteq N$. Since by assumption N is a model of $P^{M'}_{strat}$, $H(r_{strat}) \cap N \neq \emptyset$ and we can conclude that $H(r') \cap (N \cap \mathscr{U}) \neq \emptyset$, that is, rule r' is satisfied by $N \cap \mathscr{U}$.

Summarizing, no $N \subsetneq M'$ is a model of $P^{M'}_{strat}$, and hence M' is a minimal model of $P^{M'}_{strat}$ and thus an answer set of P_{strat}. □

Let us now consider normal programs. Our aim is to define a translation for associating every normal program with a super-coherent normal program. Definition 6 alone is not suitable for this purpose, as rules of the form (11) are disjunctive. However, it is not difficult to prove that the application of Definition 6 to normal programs yields head-cycle-free programs.

Lemma 7
If P is a normal program, P_{strat} is head-cycle free.

Proof
Since P is normal, all disjunctive rules in P_{strat} are of the form (11). Atoms α^T and α^F are not involved in any cycle of dependencies because α^F do not appear in any other rule heads. □

Hence, for a normal program P, we construct a head-cycle-free program P_{strat}. It is well known in the literature that a head-cycle-free program P can be associated to a *uniformly equivalent* normal program $P^{\rightarrow}$, meaning that $P \cup F$ and $P^{\rightarrow} \cup F$ are equivalent, for each set of facts F. Below, we recall this result.

Definition 7 (Definition 5.11 of Eiter *et al.* 2007)
Let P be a disjunctive program. We construct a new program $P^{\rightarrow}$ as follows:

- All the rules $r \in P$ with $H(r) = \emptyset$ belong to $P^{\rightarrow}$.
- For each rule $r \in P$ with $H(r) \neq \emptyset$, and for each atom $\alpha \in H(r)$, program $P^{\rightarrow}$ contains a rule $r^{\rightarrow}$ such that $H(r^{\rightarrow}) = \{\alpha\}$, $B^+(r^{\rightarrow}) = B^+(r)$ and $B^-(r^{\rightarrow}) = B^-(r) \cup (H(r) \setminus \{\alpha\})$.

Theorem 4 (Adapted from Theorem 5.12 of Eiter *et al.* 2007)
For any head-cycle-free program P, and any set of atoms F, it holds that $AS(P \cup F) = AS(P^{\rightarrow} \cup F)$.

Thus, by combining Definitions 6 and 7, we can associate every normal program with a super-coherent normal program.

Theorem 5
For a program P, let $P^{\rightarrow}_{strat}$ be the program obtained by applying the transformation in Definition 7 to the program P_{strat}. If P is a normal program, $P^{\rightarrow}_{strat}$ is super-coherent and such that:

$$AS(P) = \{M \cap \mathscr{U} \mid M \in AS(P^{\rightarrow}_{strat}) \wedge fail \notin M\}.$$

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Proof
By Lemma 7 and Theorem 4, $AS(P_{strat} \cup F) = AS(P^{\rightarrow}_{strat} \cup F)$, for any set F of facts. Moreover, with Lemma 5, we obtain that $P^{\rightarrow}_{strat}$ is super-coherent. Finally, by using Theorem 3, $AS(P) = \{M \cap \mathscr{U} \mid M \in AS(P^{\rightarrow}_{strat}) \wedge fail \notin M\}$ holds. □

Note that the program $P^{\rightarrow}_{strat}$ can be obtained from P by applying the rewriting introduced by Definition 6, in which rules of the form (11) are replaced by rules of the following form:

$$\alpha^T \leftarrow not\ \alpha^F \qquad \alpha^F \leftarrow not\ \alpha^T. \tag{15}$$

We also note that it is possible to use rules (15) instead of rules (11) for the general, disjunctive case such that Theorem 3 would still hold. However, the proof is somewhat more involved.

We can now relate the coherence and query answering problems for ASP to corresponding query answering problems for ASP^{sc}, and thus conclude that all of the problems solvable in this way using ASP are also solvable using ASP^{sc}.

Theorem 6
Given a program P over $\mathscr{U}$,

(1) $AS(P) = \emptyset$ if and only if *fail* is cautiously true for P_{strat};
(2) a query q is bravely true for P if and only if q' is bravely true for the ASP^{sc} program $P_{strat} \cup \{q' \leftarrow q, not\ fail\}$;
(3) a query q is cautiously true for P if and only if q' is cautiously true for the ASP^{sc} program $P_{strat} \cup \{q' \leftarrow q; q' \leftarrow fail\}$,

where q' is a fresh atom which does not occur in P and P_{strat}.

Proof
We first observe that programs $P_{strat} \cup \{q' \leftarrow q, not\ fail\}$ and $P_{strat} \cup \{q' \leftarrow q; q' \leftarrow fail\}$ are ASP^{sc}. In fact, rules $q' \leftarrow q, not\ fail$ and $\{q' \leftarrow q; q' \leftarrow fail\}$ do not introduce cycles in these programs as q' does not occur in P and P_{strat}. We now prove the other statements of the theorem.

(1) If $AS(P) = \emptyset$, by Theorem 3 either $AS(P_{strat}) = \emptyset$ (this will not occur because P_{strat} is super-coherent) or $fail \in M$ for all $M \in AS(P_{strat})$. In either case, *fail* is cautiously true for P_{strat}. If *fail* is cautiously true for P_{strat}, then $fail \in M$ for all $M \in AS(P_{strat})$, hence by Theorem 3, $AS(P) = \emptyset$.

(2) Let P^+ denote $P_{strat} \cup \{q' \leftarrow q, not\ fail\}$. If q is bravely true for P, there is $M \in AS(P)$ such that $q \in M$, and by Theorem 3 there is an $M' \in AS(P_{strat})$ such that $fail \notin M'$ and $M = M' \cap \mathscr{U}$, and hence $q \in M'$. Therefore, $M' \cup \{q'\} \in AS(P^+)$ and thus q' is bravely true for P^+. If q' is bravely true for P^+, then there exists one $N \in AS(P^+)$ such that $q' \in N$. Then $N^- = N \setminus \{q'\}$ is in $AS(P_{strat})$ and *fail* is not in N and N^-, while q is in both N and N^-. Therefore, by Theorem 3, $N' = N \cap \mathscr{U}$ is in $AS(P)$ and $q \in N'$, hence q is bravely true for P.

(3) Let P^+ denote $P_{strat} \cup \{q' \leftarrow q; q' \leftarrow fail\}$. If q is cautiously true for P, then $q \in M$ for all $M \in AS(P)$. By Theorem 3, each $M' \in AS(P_{strat})$ either is of the form $M = M' \cap \mathscr{U}$ for some $M \in AS(P)$ and thus $q \in M'$ or $fail \in M'$. In either case we get $M' \cup \{q'\} \in AS(P^+)$ and hence that q' is cautiously true for P^+. If q' is

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

cautiously true for P^+, then $q' \in N$ for each $N \in AS(P^+)$. Each of these N contains either (a) *fail* or (b) q, and $AS(P_{strat}) = \{N \setminus \{q'\} \mid N \in AS(P^+)\}$. By Theorem 3, each $N^- \in AS(P)$ is of the form $N^- = N \cap \mathscr{U}$ for those $N \in AS(P^+)$ which do not contain *fail*, hence are of type (b) and contain q. Therefore, q is in all $N^- \in AS(P)$ and is therefore cautiously true for P. □

5 Some implications

Oetsch *et al.* (2007) studied the following problem under the name "uniform equivalence with projection":

- Given two programs P and Q, and two sets A, B of atoms, $P \equiv_B^A Q$ if and only if 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 if and only if $P \equiv_\emptyset^U Q$, 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. Formally, we have

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

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

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.

Oetsch *et al.* (2007) also investigated the complexity of the problem of deciding uniform equivalence with projection, 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 of Oetsch *et al.* (2007). Using Proposition 3 and the main results in this paper, we can state the following result.

Theorem 7
The problem of deciding $P \equiv_\emptyset^U Q$ for given disjunctive (resp. normal) programs P and Q is Π_3^P-complete (resp. Π_2^P-complete) even in case U is the set of all atoms occurring in P or Q. Hardness already holds if one of the programs is the empty program.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

6 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 the ASP variant of this technique only applies to the class of programs called super-coherent (Alviano and Faber 2011). Super-coherent programs are those programs 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 *per se* (for instance, since there is a strong relation to some problems in equivalence checking), and also showed that all of the interesting ASP tasks can be solved using super-coherent programs only. For these reasons we have studied 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. Our results also imply that any reasoning tasks over ASP can be transformed into tasks over ASP^{sc}. In particular, this means that all query answering tasks over ASP can be transformed into query answering over ASP^{sc}, on which the magic set technique can therefore be applied. However, we believe that the magic set technique will often not produce efficient rewritings for programs obtained by the automatic transformation of Section 4. A careful analysis of this aspect is however left for future work, as it is also not central to the topics of this paper.

Acknowledgements

This work is partly supported by Regione Calabria and EU under POR Calabria FESR 2007–2013 and within the PIA project of DLVSYSTEM s.r.l.; MIUR under the PRIN project LoDeN; and the Vienna University of Technology under the program "Innovative Ideas." We thank the anonymous reviewers (also those from ASPOCP and CILC) for their valuable comments.

References

Alviano, M. and Faber, W. 2010. Dynamic magic sets for super-consistent answer set programs. In *Proceedings of the 3rd Workshop on Answer Set Programming and Other Computing Paradigms* (*ASPOCP 2010*), M. Balduccini and S. Woltran, Eds.

Alviano, M. and Faber, W. 2011. Dynamic magic sets and super-coherent answer set programs. *AI Communications 24,* 2, 125–145.

Alviano, M., Faber, W., Greco, G. and Leone, N. 2012. Magic sets for disjunctive datalog programs. *Artificial Intelligence* (Elsevier) *187–188,* 156–192.

Apt, K. R., Blair, H. A. and Walker, A. 1988. Towards a Theory of Declarative Knowledge. In *Foundations of Deductive Databases and Logic Programming*, J. Minker, Ed. Morgan Kaufmann, Washington, DC, 89–148.

Bancilhon, F., Maier, D., Sagiv, Y. and Ullman, J. D. 1986. Magic sets and other strange ways to implement logic programs. In *Proceedings of the Fifth ACM SIGACT-SIGMOD Symposium on Principles of Database Systems*, Cambridge, MA, 1–15.

Baral, C. 2003. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, UK.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Beeri, C. and Ramakrishnan, R. 1991. On the power of magic. *Journal of Logic Programming 10,* 255–259.

Ben-Eliyahu, R. and Dechter, R. 1994. Propositional semantics for disjunctive logic programs. *Annals of Mathematics and Artificial Intelligence 12*, 53–87.

Cumbo, C., Faber, W., Greco, G. and Leone, N. 2004. 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. Springer-Verlag, Berlin, Germany, 371–385.

Drescher, C., Gebser, M., Grote, T., Kaufmann, B., König, A., Ostrowski, M. and Schaub, T. 2008. Conflict-driven disjunctive answer set solving. In *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning* (*KR 2008*), G. Brewka and J. Lang, Eds. AAAI Press, Sydney, Australia, 422–432.

Dung, P. M. 1992. On the relations between stable and well-founded semantics of logic programs. *Theoretical Computer Science 105,* 1, 7–25.

Eiter, T., Fink, M., Tompits, H. and Woltran, S. 2004. On eliminating disjunctions in stable logic programming. In *Proceedings of the 9th International Conference on Principles of Knowledge Representation and Reasoning* (*KR 2004*). AAAI Press, Palo Alto, CA, 447–458.

Eiter, T., Fink, M. and Woltran, S. 2007. Semantical characterizations and complexity of equivalences in stable logic programming. *ACM Transactions on Computational Logic 8*, 3, 1–53.

Eiter, T. and Gottlob, G. 1995. On the computational cost of disjunctive logic programming: propositional case. *Annals of Mathematics and Artificial Intelligence 15,* 3–4, 289–323.

Eiter, T., Gottlob, G. and Mannila, H. 1997. Disjunctive datalog. *ACM Transactions on Database Systems 22,* 3 (September), 364–418.

Eiter, T., Tompits, H. and Woltran, S. 2005. On solution correspondences in answer set programming. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, L. P. Kaelbling and A. Saffiotti, Eds. Professional Book Center, Denver CO, 97–102.

Gelfond, M. and Lifschitz, V. 1991. Classical negation in logic programs and disjunctive databases. *New Generation Computing 9,* 365–385.

Gottlob, G., Leone, N. and Veith, H. 1999. Succinctness as a source of expression complexity. *Annals of Pure and Applied Logic 97,* 1–3, 231–260.

Greco, S. 2003. Binding propagation techniques for the optimization of bound disjunctive queries. *IEEE Transactions on Knowledge and Data Engineering 15,* 2 (March/April), 368–385.

Janhunen, T., Niemelä, I., Seipel, D., Simons, P. and You, J.-H. 2006. Unfolding partiality and disjunctions in stable model semantics. *ACM Transactions on Computational Logic 7,* 1 (January), 1–37.

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. and Terracina, G. 2005. 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*). ACM Press, Baltimore, MD, 915–917.

Leone, N., Pfeifer, G., Faber, W., Eiter, T., Gottlob, G., Perri, S. and Scarcello, F. 2006. The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic 7,* 3 (July), 499–562.

Lierler, Y. 2005. Disjunctive answer set programming via satisfiability. In *Proceedings of Logic Programming and Nonmonotonic Reasoning – 8th International Conference (LPNMR'05)*, Diamante, Italy, September 2005, C. Baral, G. Greco, N. Leone and G. Terracina, Eds. LNCS, vol. 3662. Springer-Verlag, New York, NY, 447–451.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proceedings of the 11th International Conference on Logic Programming* (*ICLP'94*), Santa Margherita Ligure, Italy, P. Van Hentenryck, Ed. MIT Press, Cambridge, MA, 23–37.

Manna, M., Ruffolo, M., Oro, E., Alviano, M. and Leone, N. 2012. The HiLeX system for semantic information extraction. *Transactions on Large-Scale Data and Knowledge-Centered Systems*, 91–125.

Manna, M., Scarcello, F. and Leone, N. 2011. On the complexity of regular-grammars with integer attributes. *Journal of Computer and System Sciences 77,* 2, 393–421.

Oetsch, J., Tompits, H. and Woltran, S. 2007. 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*). AAAI Press, Palo Alto, CA, 458–464.

Papadimitriou, C. H. and Yannakakis, M. 1997. Tie-breaking semantics and structural totality. *Journal of Computer and System Sciences 54,* 1, 48–60.

Ricca, F., Alviano, M., Dimasi, A., Grasso, G., Ielpa, S. M., Iiritano, S., Manna, M., and Leone, N. 2010. A logic-based system for e-tourism. *Fundamenta Informaticae 105,* 1–2, 35–55.

Ricca, F., Grasso, G., Alviano, M., Manna, M., Lio, V., Iiritano, S. and Leone, N. 2012. Team-building with answer set programming in the Gioia-Tauro seaport. *Theory and Practice of Logic Programming* (Cambridge University Press) *12,* 3, 361–381.

Ullman, J. D. 1989. *Principles of Database and Knowledge Base Systems*. Computer Science Press, Los Alamitos, CA.

You, J.-H. and Yuan, L. Y. 1994. A three-valued semantics for deductive databases and logic programs. *Journal of Computer and System Sciences 49,* 2, 334–361.

https://doi.org/10.1017/S147106841300001X Published online by Cambridge University Press