Properties of Answer Set Programming with Convex Generalized Atoms

Mario Alviano and Wolfgang Faber

Department of Mathematics and Computer Science
University of Calabria
87036 Rende (CS), Italy
{alviano,faber}@mat.unical.it

Abstract. In recent years, Answer Set Programming (ASP), logic programming under the stable model or answer set semantics, has seen several extensions by generalizing the notion of an atom in these programs: be it aggregate atoms, HEX atoms, generalized quantifiers, or abstract constraints, the idea is to have more complicated satisfaction patterns in the lattice of Herbrand interpretations than traditional, simple atoms. In this paper we refer to any of these constructs as generalized atoms. Several semantics with differing characteristics have been proposed for these extensions, rendering the big picture somewhat blurry. In this paper, we analyze the class of programs that have convex generalized atoms (originally proposed by Liu and Truszczyński in [10]) in rule bodies and show that for this class many of the proposed semantics coincide. This is an interesting result, since recently it has been shown that this class is the precise complexity boundary for the FLP semantics. We investigate whether similar results also hold for other semantics, and discuss the implications of our findings.

1 Introduction

Various extensions of the basic Answer Set Programming language have been proposed by allowing more general atoms in rule bodies, for example aggregate atoms, HEX atoms, dl-atoms, generalized quantifiers, or abstract constraints. A number of semantics have been proposed for such programs, most notably the FLP semantics [7] and a number of coinciding semantics that we will collectively refer to as PSP semantics (from Pelov, Son, and Pontelli) [13, 17]. All of these semantics coincide with traditional ASP semantics when no generalized atoms are present. Moreover, they coincide on programs that have atomic rule heads and contain only monotonic generalized atoms. In [9] it is furthermore hinted that the semantics also coincide on programs that have atomic rule heads and contain only convex generalized atoms. However, no formal proof is available for this claim, and the informal explanation given in [9] is not as general as it could be, as we will show.

In this paper, we undertake a deeper investigation on the similarities and differences between the FLP and PSP semantics. In order to do this, we consider a simplified, yet expressive propositional language: sets of rules with atomic heads and bodies that are formed of a single "structure," which are functions mapping interpretations to Boolean values[1]. Clearly, structures encompass atoms, literals, and conjunctions thereof, but can

---

[1] Note that (apart from the name) there is no connection to structures in first-order logic.

represent any propositional formula, generalized atom, or conjunctions of generalized atoms. Each structure has an associated domain, which is the set of propositional atoms on which the structure's truth valuation depends. We can then classify the structures by their semantic properties, in particular, we will focus on the class of convex structures, which have single contiguous areas of truth in the lattice of interpretations. Convex structures include atoms and literals, and they are closed under conjunction (but not under negation or disjunction).

We first formally prove the claim that the FLP and PSP semantics coincide on programs with convex structures, as originally reported in [9]. We will then move on to the main focus of this paper, trying to understand whether there is any larger class for which the semantics coincide. It is known that for programs with general structures all PSP answer sets are FLP answer sets, but not all FLP answer sets are PSP answer sets. The precise boundary for exhibiting the semantic difference is instead unknown.

We will approach this question using complexity arguments. Recently, we could show that convex structures form the precise boundary for a complexity jump in the polynomial hierarchy on cautious reasoning (but most other decision problems as well) for the FLP semantics. Cautious reasoning is Π_2^P-complete for the FLP semantics when allowing any non-convex structure and its variants (renaming atoms) in the input program, but it is $coNP$-complete for convex structures. When considering the PSP semantics, cautious reasoning is also Π_2^P-complete when allowing any kind of structures in the input. This follows from a result in [13], and we provide an alternative proof in this paper. Analyzing this proof, it becomes clear that there is a different source of complexity for PSP than for FLP.

We then show that this different source of complexity also yields a different shape of the boundary for the complexity jump in PSP. Indeed, we first show that for a simple non-convex structure, cautious reasoning is still in $coNP$ for the PSP semantics, while the problem is Π_2^P-hard in the presence of this structure for the FLP semantics. It turns out that the same argument works for many non-convex structures, in particular, for all structures with a domain size bounded by a constant. The domain size therefore serves as a parameter that simplifies the complexity of the problem for the PSP semantics (unless the polynomial hierarchy collapses to its first level). This also means that the complexity boundary for PSP has a non-uniform shape, in the sense that an infinite number of different non-convex structures must be available for obtaining Π_2^P-hardness for cautious reasoning. This is in contrast to the FLP semantics, where the presence of a single non-convex structure is sufficient.

2 Syntax and Semantics

In this section we first introduce the syntax used in the paper. This is mainly based on the notion of structures, i.e., functions mapping interpretations into Boolean truth values. Then, we introduce few semantic notions and in particular we characterize structures in terms of monotonicity. Finally, we define the two semantics analyzed in this paper, namely FLP and PSP.

2.1 Syntax

Let $\mathcal{U}$ be a fixed, countable set of propositional atoms. An interpretation I is a subset of $\mathcal{U}$. A structure S on $\mathcal{U}$ is a mapping of interpretations into Boolean truth values. Each structure S has an associated, finite domain $D_S \subset \mathcal{U}$, indicating those atoms that are relevant to the structure.

Example 1. A structure S_1 modeling a conjunction $a_1, \ldots, a_n$ $(n \geq 0)$ of propositional atoms is such that $D_{S_1} = \{a_1, \ldots, a_n\}$ and, for every interpretation I, S_1 maps I to true if and only if $D_{S_1} \subseteq I$.

A structure S_2 modeling a conjunction $a_1, \ldots, a_m, not\ a_{m+1}, \ldots, not\ a_n$ $(n \geq m \geq 0)$ of literals, where $a_1, \ldots, a_n$ are propositional atoms and not denotes *negation as failure*, is such that $D_{S_2} = \{a_1, \ldots, a_n\}$ and, for every interpretation I, S_2 maps I to true if and only if $\{a_1, \ldots, a_m\} \subseteq I$ and $\{a_{m+1}, \ldots, a_n\} \cap I = \emptyset$.

A structure S_3 modeling an aggregate $COUNT(\{a_1, \ldots, a_n\}) \neq k$ $(n \geq k \geq 0)$, where $a_1, \ldots, a_n$ are propositional atoms, is such that $D_{S_3} = \{a_1, \ldots, a_n\}$ and, for every interpretation I, S_3 maps I to true if and only if $|D_{S_3} \cap I| \neq k$.

A general rule r is of the following form:

$$H(r) \leftarrow B(r) \tag{1}$$

where $H(r)$ is a propositional atom in $\mathcal{U}$ referred as the head of r, and $B(r)$ is a structure on $\mathcal{U}$ called the body of r. A general program P is a set of general rules.

Example 2. Let S_4 map to true any interpretation I such that $I \cap \{a, b\} \neq \{b\}$, and let S_5 map to true any interpretation I such that $I \cap \{a, b\} \neq \{a\}$. Hence, program $P_1 = \{a \leftarrow S_4; b \leftarrow S_5\}$ is equivalent to the following program with aggregates:

$$a \leftarrow SUM(\{a = 1, b = -1\}) \geq 0$$
$$b \leftarrow SUM(\{a = -1, b = 1\}) \geq 0$$

Note that no particular assumption is made on the syntax of rule bodies; in the case of normal propositional logic programs these structures are conjunctions of literals. We assume that structures are closed under propositional variants, that is, if S is a structure, for any bijection $\sigma : \mathcal{U} \to \mathcal{U}$, also $S\sigma$ is a structure, and the associated domain is $D_{S\sigma} = \{\sigma(a) \mid a \in D_S\}$.

Example 3. Consider S_4 and S_5 from Example 2, and a bijection σ_1 such that $\sigma_1(a) = b$. Hence, $S_5 = S_4\sigma_1$, that is, S_5 is a variant of S_4.

Given a set of structures $\mathbf{S}$, by datalog$^{\mathbf{S}}$ we refer to the class of programs that may contain only the following rule bodies: structures corresponding to conjunctions of atoms, any structure $S \in \mathbf{S}$, or any of its variants $S\sigma$.

Example 4. For every $n \geq m \geq 0$, let $S^{m,n}$ denote the structure S_2 from Example 1. The class of normal datalog programs is datalog$^{\{S^{m,n} \mid n \geq m \geq 0\}}$.

Note that this syntax does not explicitly allow for negated structures. One can, however, choose the complementary structure for simulating negation. This would be akin to the "negation as complement" interpretation of negated aggregates that is prevalent in the literature.

2.2 Semantics

Let $I \subseteq \mathcal{U}$ be an interpretation. I is a model for a structure S, denoted $I \models S$, if S maps I to true. Otherwise, if S maps I to false, I is not a model of S, denoted $I \not\models S$. We require that atoms outside the domain of S are irrelevant for modelhood, that is, for any interpretation I and $X \subseteq \mathcal{U} \setminus D_S$ it holds that $I \models S$ if and only if $I \cup X \models S$. Moreover, for any bijection $\sigma : \mathcal{U} \to \mathcal{U}$, let $I\sigma = \{\sigma(a) \mid a \in I\}$, and we require that $I\sigma \models S\sigma$ if and only if $I \models S$. I is a model of a rule r of the form (1), denoted $I \models r$, if $H(r) \in I$ whenever $I \models B(r)$. I is a model of a program P, denoted $I \models P$, if $I \models r$ for every rule $r \in P$.

Example 5. Consider program P_1 from Example 2. It can be observed that $\emptyset \not\models P_1$ and $\{a, b\} \not\models P_1$ (both rules have true bodies but false heads), while $\{a\} \models P_1$ and $\{b\} \models P_1$.

Structures can be characterized in terms of *monotonicity* as follows.

Definition 1 (Monotone Structures). *A structure S is monotonic if for all pairs X, Y of interpretations such that $X \subset Y$, $X \models S$ implies $Y \models S$.*

Definition 2 (Antimonotone Structures). *A structure S is antimonotonic if for all pairs Y, Z of interpretations such that $Y \subset Z$, $Z \models S$ implies $Y \models S$.*

Definition 3 (Convex Structures). *A structure S is convex if for all triples X, Y, Z of interpretations such that $X \subset Y \subset Z$, $X \models S$ and $Z \models S$ implies $Y \models S$.*

Note that monotonic and antimonotonic structures are convex. Moreover, note that convex structures are closed under conjunction (but not under disjunction or negation).

Example 6. Structure $S^{m,n}$ from Example 4 is convex in general; it is monotonic if $m = n$, and antimonotonic if $m = 0$. Structure S_3 from Example 1, instead, is non-convex if $n > k > 0$; it is monotonic if $k = 0$, and antimonotonic if $n = k$.

We first describe a reduct-based semantics, usually referred to as FLP, which has been described and analyzed in [6, 7].

Definition 4 (FLP Reduct). *The FLP reduct P^I of a program P with respect to I is defined as the set $\{r \in P \mid I \models B(r)\}$.*

Definition 5 (FLP Answer Sets). *I is an FLP answer set of P if $I \models P^I$ and for each $J \subset I$ it holds that $J \not\models P^I$.*

Example 7. Consider program P_1 from Example 2 and the interpretation $\{a\}$. The reduct $P_1^{\{a\}}$ is $\{a \leftarrow S_4\}$. Since $\{a\}$ is a minimal model of the reduct, $\{a\}$ is an FLP answer set of P_1. Similarly, it can be observed that $\{b\}$ is another FLP answer set. Actually, these are the only FLP answer sets of the program.

We will next describe a different semantics, using the definition of [17], called "fix-point answer set" in that paper. Theorem 3 in [17] shows that it is actually equivalent to the two-valued fix-point of ultimate approximations of generalized atoms in [13][2], and therefore with stable models for ultimate approximations of aggregates as defined in [14]. We will refer to it as PSP to abbreviate Pelov/Son/Pontelli, the names most frequently associated with this semantics.

Definition 6 (Conditional Satisfaction). *A structure S on $\mathcal{U}$ is conditionally satisfied by a pair of interpretations (I, M), denoted $(I, M) \models S$, if $J \models S$ for each J such that $I \subseteq J \subseteq M$.*

Definition 7 (PSP Answer Sets). *An interpretation M is a PSP answer set if M is the least fixpoint of the following operator:*

$$K_M^P(I) = \{H(r) \mid r \in P \wedge (I, M) \models B(r)\}. \tag{2}$$

Example 8. Consider program P_1 from Example 2 and the interpretation $\{a\}$. The least fixpoint of $K_{\{a\}}^{P_1}$ is $\{a\}$. In fact, $\emptyset \models S_4$ and $\{a\} \models S_4$, hence $(\emptyset, \{a\}) \models S_4$, while $\{a\} \not\models S_5$ and thus $(\emptyset, \{a\}) \not\models S_5$ and $(\{a\}, \{a\}) \not\models S_5$. Therefore, $\{a\}$ is a PSP answer set. Also $\{b\}$ is a PSP answer set.

On programs considered in this paper, PSP answer sets also coincide with "answer sets" defined in [18] (by virtue of Proposition 10 in [18]) and "well-justified FLP answer sets" of [15] (by virtue of Theorem 5 in [15]). The latter is particularly interesting, as it is defined by first forming the FLP reduct. Indeed, as shown in [15], the operator K_M^P can be equivalently defined as follows:

$$K_M^P(I) = \{H(r) \mid r \in P^M \wedge (I, M) \models B(r)\}. \tag{3}$$

There are several other semantic definitions on programs that have some restrictions on the admissible structures, which also coincide with the PSP semantics on programs as defined in this paper with the respective structure restriction. Examples are [12] for monotonic structures (that are also allowed to occur in rule heads in that paper), or [16] that allows for structures corresponding to cardinality and weight constraints and largely coincide with the PSP semantics (see [13] for a discussion on structures on which the semantics coincides).

In this paper we are mainly interested in cautious reasoning, defined next.

Definition 8 (Cautious Reasoning). *A propositional atom a is a cautious consequence of a program P under FLP (resp. PSP) semantics, denoted $P \models_c^{FLP} a$ (resp. $P \models_c^{PSP} a$), if a belongs to all FLP (resp. PSP) answer set of P.*

Example 9. Consider program P_1 from Example 2. We have $P_1 \not\models_c^{FLP} a$ and $P_1 \not\models_c^{FLP} b$, and similar for PSP semantics. If we add $a \leftarrow b$ and $b \leftarrow a$ to the program, then there is only one FLP answer set, namely $\{a, b\}$, and no PSP answer sets. In this case a and b are cautious consequences of the program (under both semantics).

---

[2] There is an even closer relationship, as the operator $K_M^P(I)$ of [17] coincides with $\phi_P^{aggr,1}(I, M)$ defined in [13], as shown in the appendix of [17]

3 Exploring the Relationship between the FLP and PSP Semantics

In this section, we examine in detail how the FLP and PSP semantics relate. We shall proceed in three steps. First, we formally prove that FLP and PSP semantics coincide on programs with convex structures in Section 3.1. Next, we turn towards complexity as a tool to understand whether there can be any larger class of coinciding programs. We start in Section 3.2 with a result that shows that programs without restrictions exhibit the same complexity under both FLP and PSP semantics. However, it is known that the semantics do not coincide for programs without restrictions, and we examine the complexity proofs to highlight the different complexity sources. These findings are then applied in Section 3.3 in order to identify programs with bounded non-convex structures, on which the complexities for FLP and PSP semantics differ. Under usual complexity assumptions, this also implies that programs with convex aggregates is the largest class of programs on which FLP and PSP coincide.

3.1 Unison: Convex Structures

In this section we show that for programs with convex aggregates the FLP and PSP semantics coincide. In [9] it is stated that many semantics (and in particular, FLP and PSP) "agree on [...] programs with convex aggregates" because "they can be regarded as special programs with monotone constraints." However, the comment on regarding convex aggregates as monotone constraints relies on a transformation described in [10] that transforms convex structures into conjunctions of positive and negated monotone constraints. Since our language does not explicitly allow negation, and in particular since convex structures are not closed under negation, we next prove in a more direct manner that the FLP and PSP semantics coincide on convex structures.

One direction of the proof relies on the well-known more general fact that each PSP answer set is also an FLP answer set. This has been stated as Theorem 2 in [17] and Proposition 8.1 in [14].

Theorem 1. *Let P be program whose body structures are convex, and let M be an interpretation. M is an FLP answer set of P if and only if M is an PSP answer set of P.*

Proof. The left implication follows from Theorem 2 in [17]. For the right implication, let M be an FLP answer set of P. Let $K_0 := \emptyset$, $K_{i+1} := K_M^P(K_i)$ for $i \geq 0$, and let K be the fixpoint of this sequence. Since M is a minimal model of P^M by definition of FLP answer set, we can prove the claim by showing (i) $K \models P^M$ and (ii) $K \subseteq M$.

(i) Consider a rule $r \in P^M$ such that $K \models B(r)$. We have to show $H(r) \in K$. Since $r \in P^M$, $M \models B(r)$ holds. Thus, $(K, M) \models B(r)$ and therefore $H(r) \in K$.

(ii) We prove $K_i \subseteq M$ for each $i \geq 0$. We use induction on i. The base case is trivially true as $K_0 = \emptyset \subseteq M$. Suppose $K_i \subseteq M$ for some $i \geq 0$ in order to prove $K_{i+1} \subseteq M$. By definition of K_M^P, for each $a \in K_{i+1}$ there is $r \in P^M$ such that $H(r) = a$ and $(K_i, M) \models B(r)$. Thus, $M \models B(r)$, which implies $a \in M$. □

Therefore, programs with convex structures form a class of programs for which the FLP and PSP semantics coincide. In the following, we will show that it is likely also the largest class for which this holds.

3.2 Consonance: Complexity of Unrestricted Structures

In this section we will examine the computational impact of allowing non-convex structures. We will limit ourselves to structures for which the truth value with respect to an interpretation can be determined in polynomial time. Moreover, we will focus on cautious reasoning, but similar considerations apply also to related problems such as brave reasoning, answer set existence, or answer set checking.

It is known that cautious reasoning over programs with arbitrary structures under the FLP semantics is Π_2^P-complete in general, as shown in [7]. Pelov has shown Σ_2^P-completeness for deciding the existence of PSP answer sets in [13], from which Π_2^P-completeness for cautious reasoning under the PSP semantics can be derived. We formally state this result now and provide a different proof than Pelov's that will more directly lead to the subsequent considerations.

Theorem 2. *Cautious reasoning under PSP semantics is Π_2^P-complete.*

Proof. Membership follows by Corollary 1 of [17]. For the hardness, we provide a reduction from 2-QBF$_\forall$. Let $\Psi = \forall x_1 \cdots \forall x_m \exists y_1 \cdots \exists y_n\ E$, where E is in 3CNF. Formula Ψ is equivalent to $\neg\Psi'$, where $\Psi' = \exists x_1 \cdots \exists x_m \forall y_1 \cdots \forall y_n\ E'$, and E' is a 3DNF equivalent to $\neg E$ and obtained by applying De Morgan's laws. To prove the claim we construct a program P_Ψ such that $P_\Psi \models_c^{PSP} w$ (w a fresh atom) if and only if Ψ is valid, i.e., iff Ψ' is invalid.

Let $E' = (l_{1,1} \wedge l_{1,2} \wedge l_{1,3}) \vee \cdots \vee (l_{k,1} \wedge l_{k,2} \wedge l_{k,3})$, for some $k \geq 1$. Program P_Ψ is the following:

$$
\begin{array}{llll}
x_i^T \leftarrow not\ x_i^F & x_i^F \leftarrow not\ x_i^T & i \in \{1,\ldots,m\} & (4)\\
y_i^T \leftarrow not\ y_i^F & y_i^F \leftarrow not\ y_i^T & i \in \{1,\ldots,n\} & (5)\\
y_i^T \leftarrow sat & y_i^F \leftarrow sat & i \in \{1,\ldots,n\} & (6)\\
sat \leftarrow \mu(E') & & & (7)\\
w \leftarrow not\ sat & & & (8)
\end{array}
$$

where μ is defined recursively as follows:

- $\mu(E') := (\mu(l_{1,1}) \wedge \mu(l_{1,2}) \wedge \mu(l_{1,3})) \vee \cdots \vee (\mu(l_{k,1}) \wedge \mu(l_{k,2}) \wedge \mu(l_{k,3}))$;
- $\mu(x_i) := x_i^T$ and $\mu(\neg\, x_i) := x_i^F$ for all $i = 1, \ldots, m$;
- $\mu(y_i) := y_i^T$ and $\mu(\neg\, y_i) := y_i^F$ for all $i = 1, \ldots, n$.

Note that structure $\mu(E')$ can also be encoded by means of a *sum* aggregate as shown in [1].

Rules (4)–(5) force each PSP answer set of P_Ψ to contain at least one of x_i^T, x_i^F ($i \in \{1, \ldots, m\}$), and one of y_j^T, y_j^F ($j \in \{1, \ldots, m\}$), respectively, encoding an assignment of the propositional variables in Ψ'. Rules (6) are used to simulate universality of the y variables, as described later. Having an assignment, rule (7) derives sat if the assignment satisfies some disjunct of E' (and hence also E' itself). Finally, rule (8) derives w if sat is false.

We first show that Ψ not valid implies $P_\Psi \not\models_c^{PSP} w$. If Ψ is not valid, Ψ' is valid. Hence, there is an assignment ν for $x_1, \ldots, x_m$ such that no extension to $y_1, \ldots, y_n$ satisfies E, i.e., all these extensions satisfy E'. Let us consider the following interpretation

(which is also a model of P_Ψ):

$$M = \{x_i^T \mid \nu(x_i) = 1,\ i = 1, \ldots, m\} \cup \{x_i^F \mid \nu(x_i) = 0,\ i = 1, \ldots, m\}$$
$$\cup \ \{y_i^T, y_i^F \mid i = 1, \ldots, n\} \cup \{sat\}$$

We claim that M is a PSP answer set of P_Ψ. In fact, $K_M^{P_\Psi}(\emptyset) \supseteq \{x_i^T \mid \nu(x_i) = 1,\ i = 1, \ldots, m\} \cup \{x_i^F \mid \nu(x_i) = 0,\ i = 1, \ldots, m\}$ because of rules (4) in P_Ψ^M. Since any assignment for the ys satisfies at least a disjunct of E', from rule (7) we derive $sat \in K_M^{P_\Psi}(K_M^{P_\Psi}(\emptyset))$. Hence, rules (6) force all y atoms to belong to $K_M^{P_\Psi}(K_M^{P_\Psi}(K_M^{P_\Psi}(\emptyset)))$, which is thus the least fixpoint of $K_M^{P_\Psi}$ and coincides with M.

Now we show that $P_\Psi \not\models_c^{PSP} w$ implies that Ψ is not valid. To this end, let M be a PSP answer set of P_Ψ such that $w \notin M$. Hence, by rule (8) we have that $M \models sat$. From $sat \in M$ and rules (6), we have $y_i^T, y_i^F \in M$ for all $i = 1, \ldots, n$. And M contains either x_i^T or x_i^F for $i = 1, \ldots, m$ because of rules (4). Suppose by contradiction that Ψ is valid. Thus, for all assignments of $x_1, \ldots, x_m$, there is an assignment for $y_1, \ldots, y_n$ such that E is true, i.e., E' is false. We can show that the least fixpoint of $K_M^{P_\Psi}$ is $K_M^{P_\Psi}(\emptyset) = \{x_i^T \mid \nu(x_i) = 1,\ i = 1, \ldots, m\} \cup \{x_i^F \mid \nu(x_i) = 0,\ i = 1, \ldots, m\}$. In fact, sat cannot be derived because $K_M^{P_\Psi}(\emptyset) \not\models \mu(E')$. We thus have a contradiction with the assumption that M is a PSP answer set of P_Ψ. $\qquad\square$

It is also known that the complexity drops to $coNP$ if structures in body rules are constrained to be convex. This appears to be "folklore" knowledge and can be argued to follow from results in [10]. An easy way to see membership in $coNP$ is that all convex structures can be decomposed into a conjunction of a monotonic and an antimonotonic structure, for which membership in $coNP$ has been shown in [7].

It is instructive to note a crucial difference between the Π_2^P-hardness proofs in [7] (and a similar one in [8]) and the proofs for Theorem 2 and the Σ_2^P result for PSP in [13].

The fundamental tool in the FLP hardness proofs is the availability of structures S_1, S_2 that allow for encoding "need to have either atom x^T or x^F, or both of them, but the latter only upon forcing the truth of both atoms." S_1, S_2 have domains $D_{S_1} = D_{S_2} = \{x^T, x^F\}$ and the following satisfaction patterns:

$$\emptyset \models S_1 \qquad \{x^T\} \models S_1 \qquad \{x^F\} \not\models S_1 \qquad \{x^T, x^F\} \models S_1$$
$$\emptyset \models S_2 \qquad \{x^T\} \not\models S_2 \qquad \{x^F\} \models S_2 \qquad \{x^T, x^F\} \models S_2$$

The reductions then use these structures in a similar way than disjunction is used in the classic Σ_2^P-hardness proofs in [3]. In particular, the same structures are used for all instances to be reduced.

On the other hand, in the PSP hardness proofs, one dedicated structure is used for each instance of the problem reduced from (2QBF in Theorem 2). Indeed, a construction using structures S_1, S_2 as described earlier is not feasible for PSP, because $(\emptyset, \{x^T, x^F\}) \not\models S_1$ and $(\emptyset, \{x^T, x^F\}) \not\models S_2$. This is because there is one satisfaction "hole" between $\emptyset$ and $\{x^T, x^F\}$ for both S_1 and S_2. In the next section, we will exploit this difference.

3.3 Dissonance: Complexity of Non-convex Structures with Bounded Domains

In this section, we look more carefully at programs with non-convex structures and identify computational differences between the FLP and PSP semantics. In [2] it has been shown that any non-convex structure (plus all of its variants) can be used in order to implement S_1 and S_2. This result makes it clear that the presence of any non-convex structure that is closed under variants causes a complexity increase for the FLP semantics (unless the polynomial hierarchy collapses). From the above considerations, it is immediately clear that the same construction is not feasible for PSP. It turns out that also no alternative way exists to obtain a similar result, and that the difference in the Π_2^P-hardness proofs for FLP and PSP is intrinsic.

We start by considering a simple non-convex structure $\mathring{A}$ with $D_{\mathring{A}} = \{x, y\}$ and $I \models \mathring{A}$ if and only if $|I \cap D_{\mathring{A}}| \neq 1$. Therefore, $\mathring{A}$ behaves like a cardinality constraint $COUNT(\{x, y\}) \neq 1$.

Proposition 1. *Deciding whether an interpretation M is a PSP answer set of a datalog$^{\{\mathring{A}\}}$ program P is feasible in polynomial time, in particular $DTIME(m^2)$, where m is the number of rules in P.*

Proof. For any interpretation, testing whether $(I, M) \models \mathring{A}\sigma$ (for a variant $\mathring{A}\sigma$ of $\mathring{A}$) can be done by examining $|I \cap D_{\mathring{A}\sigma}| = i$ and $|M \cap D_{\mathring{A}\sigma}| = j$ and returning false if either one of i, j is 1, or if $i = 0$ and $j = 2$. Alternatively, in a less syntax dependent way, one can test whether $M \models \mathring{A}\sigma$ and $(I \cup J) \models \mathring{A}\sigma$ for each $J \subseteq (M \cap D_{\mathring{A}\sigma}) \setminus (I \cap D_{\mathring{A}\sigma})$. Since there are at most 4 different J for each I, either method is feasible in constant time.

For determining whether M is a PSP answer set of P, we can check whether it is the least fixpoint of K_M^P. Computing the least fixpoint takes at most m applications of K_M^P (where m is the number of rules in P). Each application of K_M^P involves in turn at most m tests for $(I, M) \models \mathring{A}\sigma$. $\square$

Given Proposition 1 it follows that cautious reasoning is still in $coNP$ for datalog$^{\{\mathring{A}\}}$ programs under the PSP semantics.

Proposition 2. *Given a datalog$^{\{\mathring{A}\}}$ program P and an atom a, deciding $P \models_c^{PSP} a$ is in coNP.*

Proof. The complement has an immediate nondeterministic polynomial time algorithm: guess an interpretation M and verify in polynomial time that $a \notin M$ and that M is a PSP answer set of P (by virtue of Proposition 1). $\square$

It follows that for datalog$^{\{\mathring{A}\}}$ cautious reasoning (and also answer set existence and brave reasoning) is more complex for the FLP semantics than for the PSP semantics (unless the polynomial hierarchy collapses to its first level).

Examining this result and its proof carefully, we can see that it depends on the fact that each $D_{\mathring{A}\sigma}$ contains 2 elements and therefore at most 4 satisfaction tests are needed to determine $(I, M) \models \mathring{A}\sigma$. Indeed, we can apply similar reasoning whenever the domains of involved structures are smaller than a given bound.

Theorem 3. *Let P be a program. If k is an upper bound for the domain size of any structure occurring in P, then checking whether a given interpretation M is a PSP answer set of P is decidable in $DTIME(2^k m^2 p(n))$, where m is the number of rules in P and $p(n)$ is the polynomial function (in terms of the input size n) bounding determining satisfaction of any aggregate in P.*

Proof. We show that the least fixpoint of K_M^P can be computed in time $O(2^k m^2 p(n))$. In the worst case, each application of the operator derives at most one new atom, and thus the fixpoint is reached after at most m applications of the operator. Each application requires at most the evaluation of all rules of P, and thus at most m^2 rule evaluations are sufficient. To evaluate a rule, the truth of the body has to be checked w.r.t. at most 2^k interpretations (similar to Proposition 1, in which $k = 2$), each requiring $p(n)$ time. We thus obtain the bound $O(2^k m^2 p(n))$. $\square$

This means that actually most languages with non-convex structures exhibit a complexity gap between the FLP and PSP semantics. There is a uniformity issue here, which we informally noted earlier when examining the Π_2^P-hardness proof for cautious reasoning under PSP. We can now formalize this, as it follows from Theorem 3 that we need an infinite number of inherently different non-convex structures in order to obtain Π_2^P hardness.

Corollary 1. *Let $\mathbf{S}$ be any finite set of structures, possibly including non-convex structures. Cautious reasoning over datalog$^{\mathbf{S}}$ is in coNP under the PSP semantics.*

This means that there is also a clear difference in uniformity between the complexity boundary of the FLP and the PSP semantics, respectively. It also means that it is impossible to simulate the FLP semantics in a compact way using the PSP semantics on the class of programs with bounded domain structures, unless the polynomial hierarchy collapses to its first level. The general picture of our complexity results is shown in Figure 1. We can see that the complexity transition from $coNP$ to Π_2^P is different for the FLP and PSP semantics, respectively. The solid line between convex and non-convex structures denotes a crisp transition for FLP, while the dashed line between bounded non-convex and unbounded non-convex structures is a rougher transition.

4 Discussion

Looking at Figure 1, the transition from $coNP$ to Π_2^P appears somewhat irregular for PSP, as the availability of single non-convex structures does not cause the transition, but only their union. However, in practice the availability of an infinite number of different structures is not unusual: indeed, if aggregates are considered, the presence of one aggregate function and suitable comparison relations usually gives rise to such an infinite repertoire of structures.

Example 10. Consider the availability of $COUNT$ over any set of atoms and the comparison relation $\neq$. The structures generated by aggregates of the form $COUNT(S) \neq i$ do not have a bound on the domains of non-convex aggregates. Indeed, for any structure $COUNT(\{a_1, \ldots, a_k\}) \neq 1$, which is non-convex and for which the domain size is k,
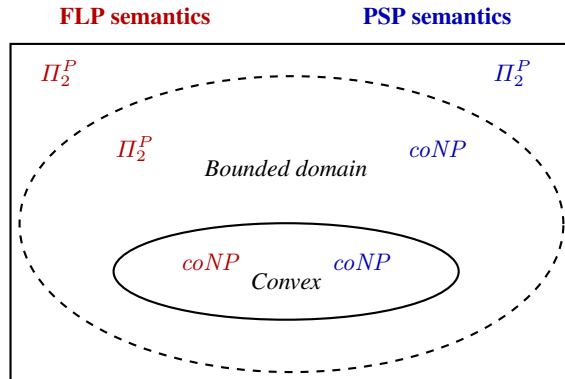
Fig. 1. Complexity of cautious reasoning

one can formulate also $COUNT(\{a_1, \ldots, a_{k+1}\}) \neq 1$, which is also non-convex and has a larger domain.

However, as noted earlier, for expressing Π_2^P-hard problems, one needs a non-uniform approach for PSP, in the sense that a dedicated aggregate has to be formulated for each problem instance, whereas for FLP one can re-use the same aggregates for all problem instances.

In practical terms, our results imply that for programs containing only convex structures, techniques as those presented in [1] for FLP can be used for computing answer sets also for PSP, and techniques presented for PSP can be used for FLP in turn. It also means that this is the largest class for which this can be done with currently available methods in an efficient way. There are several examples for convex structures that are easy to identify syntactically: count aggregates with equality guards, sum aggregates with positive summands and equality guards, dl-atoms that do not involve ⩀ and rely on a tractable Description Logic [4]. However many others are in general not convex, for example sum aggregates that involve both positive and negative summands, times aggregates that involve the factor 0, average aggregates, dl-atoms with ⩀, and so on. It is still possible to find special cases of such structures that are convex, but that requires deeper analyses.

The results also immediately imply impossibility and possibility results for rewritability: unless the polynomial hierarchy collapses to its first level, it is not possible in the FLP semantics to rewrite a program with non-convex structures into one containing only convex structures (for example, a program not containing any generalized atoms), unless disjunction or similar constructs are allowed in rule heads. On the other hand, such rewritings are possible for the PSP semantics if the non-convex structures are guaranteed to have bounded domains. This seems to be most important for dl-programs, where such rewritings are sought after.

The semantics considered in this paper encompass several approaches suggested for programs that couple answer set programming with description logics. The approaches presented in [5] and [11] directly employ the FLP semantics, while the approach of [15]

is shown to be equivalent to the PSP semantics. There are other proposals, such as [4], which appears to be different from both FLP and PSP already on convex structure. In future work we plan to relate also these other semantics with FLP and PSP and attempt to identify the largest coinciding classes of programs.

References

1. Alviano, M., Calimeri, F., Faber, W., Leone, N., Perri, S.: Unfounded Sets and Well-Founded Semantics of Answer Set Programs with Aggregates. Journal of Artificial Intelligence Research 42, 487–527 (2011)
2. Alviano, M., Faber, W.: The complexity boundary of answer set programming with generalized atoms under the flp semantics. In: Cabalar, P., Tran, S.C. (eds.) Logic Programming and Nonmonotonic Reasoning — 12th International Conference (LPNMR 2013). Springer Verlag (Sep 2013), accepted for publication
3. 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)
4. Eiter, T., Ianni, G., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the semantic web. Artif. Intell. 172(12–13), 1495–1539 (2008)
5. Eiter, T., Ianni, G., Schindlauer, R., Tompits, H.: A Uniform Integration of Higher-Order Reasoning and External Evaluations in Answer Set Programming. In: International Joint Conference on Artificial Intelligence (IJCAI) 2005. pp. 90–96. Edinburgh, UK (Aug 2005)
6. Faber, W., Leone, N., Pfeifer, G.: Recursive aggregates in disjunctive logic programs: Semantics and complexity. In: Alferes, J.J., Leite, J. (eds.) Proceedings of the 9th European Conference on Artificial Intelligence (JELIA 2004). Lecture Notes in AI (LNAI), vol. 3229, pp. 200–212. Springer Verlag (Sep 2004)
7. Faber, W., Leone, N., Pfeifer, G.: Semantics and complexity of recursive aggregates in answer set programming. Artificial Intelligence 175(1), 278–298 (2011), special Issue: John McCarthy's Legacy
8. Ferraris, P.: Answer Sets for Propositional Theories. 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. Lecture Notes in Computer Science, vol. 3662, pp. 119–131. Springer Verlag (2005)
9. Liu, L., Pontelli, E., Son, T.C., Truszczyński, M.: Logic programs with abstract constraint atoms: The role of computations. Artificial Intelligence 174(3–4), 295–315 (2010)
10. Liu, L., Truszczyński, M.: Properties and applications of programs with monotone and convex constraints. Journal of Artificial Intelligence Research 27, 299–334 (2006)
11. Lukasiewicz, T.: A novel combination of answer set programming with description logics for the semantic web. IEEE Transactions on Knowledge and Data Engineering 22(11), 1577–1592 (2010)
12. Marek, V.W., Niemelä, I., Truszczyński, M.: Logic Programming with Monotone Cardinality Atom. In: Lifschitz, V., Niemelä, I. (eds.) Proceedings of the 7th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR-7). LNAI, vol. 2923, pp. 154–166. Springer (Jan 2004)
13. Pelov, N.: Semantics of Logic Programs with Aggregates. Ph.D. thesis, Katholieke Universiteit Leuven, Leuven, Belgium (Apr 2004)
14. Pelov, N., Denecker, M., Bruynooghe, M.: Well-founded and Stable Semantics of Logic Programs with Aggregates. Theory and Practice of Logic Programming 7(3), 301–353 (2007)

15. Shen, Y.D., Wang, K.: FLP semantics without circular justifications for general logic programs. In: Hoffmann, J., Selman, B. (eds.) Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI-12) (Jul 2012)
16. Simons, P., Niemelä, I., Soininen, T.: Extending and Implementing the Stable Model Semantics. Artificial Intelligence 138, 181–234 (Jun 2002)
17. Son, T.C., Pontelli, E.: A Constructive Semantic Characterization of Aggregates in ASP. Theory and Practice of Logic Programming 7, 355–375 (May 2007)
18. Son, T.C., Pontelli, E., Tu, P.H.: Answer Sets for Logic Programs with Arbitrary Abstract Constraint Atoms. Journal of Artificial Intelligence Research 29, 353–389 (2007)