Chapter 9

Semantic Contents

In this chapter, we introduce the notion of *Semantic Contents of a program* [47, 37, 48] as an alternative point of view to obtain different answer set semantics of a program.

With the aim of having a mathematical structure that may express in a uniform way different answer sets semantics, we introduce the notion of *Semantic Contents of a program*. Given two set of formulas called hypotheses and scenario, the intuition behind the Semantic Contents of a program can be divided into two parts: The former aims at obtaining an infinite set of scenarios using a logic framework such that, the original program and a different set of hypotheses entail each scenario. Additionally, each scenario satisfies certain properties with respect to the original program and the hypotheses. The latter aims at obtaining different answer set semantics of the program by applying filters and/or orders over the set of scenarios. It is worth mentioning that this second part does not needs of a logic framework since only filters and/or orders over the set of scenarios are used. Then, since the point of view of Semantic Contents we could define Answer Set semantics as follows:

Answer Set Semantics = Logic + Filters and/or Orders.

In particular, in this chapter, we show how to obtain the *standard answer sets* (see Theorem 4.1), the *generalized answer sets* (see Definition 4.2), the *minimal generalized*

answer sets (see Definition 4.4) and a new answer set semantic introduced in this section called *partial answer sets*. In particular we present an example in a planning domain where partial answer sets could be useful.

Finally, one of our theorems says that we can have compositionality in answer sets via its semantic contents.

9.1 Semantic Contents definition

The Semantic Contents of a program could be described as a mathematical structure useful to express in a uniform way different semantics of the program. Informally, the Semantic Contents of a program is the set of pairs such that:

(1) the first entry corresponds to a set of formulas that should be added to the original program in order to entail a set of formulas corresponding to the second entry of the pair, and

(2) the second entry of the pair is a consistent extension of the original program.[1]

Definition 9.1. Let P be a program with a finite set of formulas. We define the *semantic contents* of P, denoted by $SC(P)$, as the set of all pairs $\langle S, T\rangle$ satisfying the following properties:

1. S is a set of formulas such that $T := \{X|\ P \cup S \vdash X\}$, and
2. $P \cup S$ is consistent. □

For instance, Table 9.1 considers program $P = \{\neg b \rightarrow a, \neg a \rightarrow b\}$ and shows some of the scenarios entailed by the program P together with a different set of hypotheses. Let

[1] The following definitions were given in the Background Section, however we repeat them here in order to make an easier lecture of this document: A set of formulas P is *consistent* [27] if there is not a formula A such that both A and $\neg A$ are theorems of P. A set of formulas P' is said to be an *extension* of a set of formulas P [27] if every theorem of P is a theorem of P'.

S (hypotheses)	T (Scenarios)
$\{\ \}$	$\{a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg b\}$	$\{\neg b, a, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg b, a\}$	$\{\neg b, a, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg a\}$	$\{\neg a, b, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg a, b\}$	$\{\neg a, b, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{a, b\}$	$\{a, b, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\vdots$	$\vdots$

Table 9.1: Some hypotheses and scenarios of program $P = \{\neg b \rightarrow a, \neg a \rightarrow b\}$.

us remark that the three dots $(\ldots)$ represent the rest of the formulas in each scenario, including the program P itself and the rest of theorems that P entails. We also can see that if we consider the set of hypotheses $\{\neg b\}$ then a subset of its scenario coincides with one of the answer sets of P: $\{a\}$. Additionally, if we consider the set of hypotheses $\{\neg a\}$ then a subset of its scenario coincides with the other answer sets of P: $\{b\}$. Let us notice that if we consider the set of hypotheses equal to the empty set then the scenario corresponds to the original program together with the set of theorems that the program entails.

The set S is called *an abductive* and the set T is called *a scenario*. If $SC(P)$ is a semantic contents of program P, then $SC_S(P) := \{S \mid \langle S, T\rangle \in SC(P)\}$ and $SC_T(P) := \{T \mid \langle S, T\rangle \in SC(P)\}$. Note that if P is inconsistent then $SC(P)$ is the empty contents. Sometimes, when no ambiguity arises, we will write SC_S to denote $SC_S(P)$ and SC_T to denote $SC_T(P)$.

9.2 Some properties of the semantic contents

In this section we introduce some properties about the semantic contents.

The following lemma shows that if the union of two programs entail a set of scenarios

then, this set of scenarios is the same set of scenarios that we get as result of intersect the sets of scenarios that each program entails.

Lemma 9.1. *Let P_1 and P_2 be two programs over a signature $\mathcal{L}$. Then $SC_T(P_1 \cup P_2) := SC_T(P_1) \cap SC_T(P_2)$.* □

Proof of Lemma 9.1.

We preset this proof in two parts:

1. We start proving that $SC_T(P_1 \cup P_2) \subseteq SC_T(P_1) \cap SC_T(P_2)$.

   Let $T \in SC_T(P_1 \cup P_2)$. Then by definition of semantic contents of $(P_1 \cup P_2)$, $\exists S$ such that $\langle S, T\rangle \in SC(P_1 \cup P_2)$, i.e., $\exists S$ such that

   $$T := \{X | P_2 \cup P_1 \cup S \vdash X\} \tag{9.1}$$

   We have to prove that $\exists S_1, S_2$ such that $\langle S_1, T\rangle \in SC(P_1)$ and $\langle S_2, T\rangle \in SC(P_2)$.

   Let $S_1 = (P_2 \cup S)$ and $S_2 = (P_1 \cup S)$ then by the associative property of set theory we can rewrite (9.1) as $T := \{X | P_1 \cup (P_2 \cup S) \vdash X\}$ and $T := \{X | P_2 \cup (P_1 \cup S) \vdash X\}$. Hence $\langle S_1, T\rangle \in SC(P_1)$ and $\langle S_2, T\rangle \in SC(P_2)$.

2. Now we are going to prove that $SC_T(P_1) \cap SC_T(P_2) \subseteq SC_T(P_1 \cup P_2)$.

   Let $T \in (SC_T(P_1) \cap SC_T(P_2))$, i.e., $\exists S_1$ such that $\langle S_1, T\rangle \in SC(P_1)$ and $\exists S_2$ such that $\langle S_2, T\rangle \in SC(P_2)$. We have to prove that $\exists S$ such that $\langle S, T\rangle \in SC(P_1 \cup P_2)$, i.e., $T := \{X | P_2 \cup P_1 \cup S \vdash X\}$.

   By definition of semantic contents of P_1 and P_2 we have that $T := \{X | P_1 \cup S_1 \vdash X\}$ and $T := \{X | P_2 \cup S_2 \vdash X\}$. If $S = (S_1 \cup S_2)$ then $T := \{X | P_2 \cup P_1 \cup S \vdash X\}$. Hence, $\langle S, T\rangle \in SC_T(P_1 \cup P_2)$.

□

S (hypotheses)	T (Scenarios)
$\{\ \}$	$\{a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg b\}$	$\{\neg b, a, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg b, a\}$	$\{\neg b, a, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\vdots$	$\vdots$

Table 9.2: Some hypotheses and scenarios of program $P_1 = \{\neg b \rightarrow a\}$.

S (hypotheses)	T (Scenarios)
$\{\ \}$	$\{a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg a\}$	$\{\neg a, b, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\{\neg a, b\}$	$\{\neg a, b, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\}$
$\vdots$	$\vdots$

Table 9.3: Some hypotheses and scenarios of program $P_2 = \{\neg a \rightarrow b\}$.

Example 9.1. Let us consider $P_1 = \{\neg b \rightarrow a\}$ and $P_1 = \{\neg a \rightarrow b\}$. Table 9.2 shows some of the entries of the semantic contents of P_1, and Table 9.3 shows some of the entries of the semantic contents of P_2. Additionally, Table 9.1 shows some of the entries of the semantic contents of $P_1 \cup P_2$. Then, we can see that $SC_T(P_1 \cup P_2) = SC_T(P_1) \cap SC_T(P_2) = \{\{a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a \ldots\}, \ldots\}$. □

Following the definition of the semantic contents of a program, $SC(P)$ (Definition 9.1), we can verify that the entry $\langle \emptyset, th(P) \rangle$ is one of the entries of $SC(P)$.

Lemma 9.2. *Let P be a program over a signature $\mathcal{L}$. If $SC(P)$ is the semantic contents of P then $\langle \emptyset, th(P) \rangle \in SC(P)$.* □

Proof of Lemma 9.2.

If $SC(P)$ is the semantic contents of P then $SC(P) = \{\langle S, T \rangle | S\ is\ a\ set\ of\ formulas\ such\ that\ T := \{X | P \cup S \vdash X\}\ and\ (P \cup S)\ is\ consistent\}$. In particular, if $S = \emptyset$ then $T = \{X | P \vdash X\}$. We know that $th(P) = \{X | P \vdash X\}$. Hence, $\langle \emptyset, th(P) \rangle \in SC(P)$. □

It is easy verify that if $\langle \emptyset, th(P) \rangle$ is an entry of the semantic contents of a program then $th(P)$ is the minimal scenario w.r.t. set inclusion among the set of scenarios of $SC(P)$.

Lemma 9.3. *Let P be a program over a signature $\mathcal{L}$. If $SC(P)$ is the semantic contents of P then $th(P)$ is the minimal set (w.r.t. set inclusion) in $SC_T(P)$.* □

The following lemma describes the relationship between two entries of the semantic contents of a program. This lemma shows that if the set of hypotheses of one of the entries is a subset of the set of hypotheses of the other entry then the scenario of the first entry is a subset of the scenario of the second entry.

Lemma 9.4. *Let P be a program over a signature $\mathcal{L}$. If $\langle S_1, T_1 \rangle \in SC(P)$ and $\langle S_2, T_2 \rangle \in SC(P)$ such that $S_1 \subseteq S_2$ then $T_1 \subseteq T_2$.* □

Proof of Lemma 9.4.

Let $\langle S_1, T_1 \rangle \in SC(P)$ and $\langle S_2, T_2 \rangle \in SC(P)$ such that $S_1 \subseteq S_2$, i.e., $T_1 = \{X | P \cup S_1 \vdash X\}$ and $T_2 = \{X | P \cup S_2 \vdash X\}$ such that $S_1 \subseteq S_2$.

We have to prove that $T_1 \subseteq T_2$, i.e., if $X \in T_1$ then $X \in T_1$.

By hypothesis, if X in T_1 then $P \cup S_1 \vdash X$. By monotonicity, $P \cup S_2 \vdash X$ since $S_1 \subseteq S_2$. Hence, $X \in T_1$. □

Corolary 1. Let P be a program over a signature $\mathcal{L}$. If $\langle \emptyset, T_1 \rangle \in SC(P)$ and $\langle S_2, T_2 \rangle \in SC(P)$ then $T_1 \subseteq T_2$. □

Example 9.2. Let $\langle S_1, T_1 \rangle = \langle \{\ \}, \{a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\} \rangle$ and $\langle S_2, T_2 \rangle = \langle \{\neg b\}, \{\neg b, a, a \wedge a, a \rightarrow a, b \rightarrow b, a \rightarrow (b \rightarrow a), (a \wedge a) \rightarrow a, \ldots\} \rangle$ be two entries of the semantic contents of program P_1 of Example 9.1 such that $S_1 \subseteq S_2$. We can see in Table 9.2 that $T_1 \subseteq T_2$. □

The following two lemmas describe the relationship between the semantic contents of a program and the set of theorems of the same program. These lemmas use definitions of $th(P)$ and $K(R)$ that where given in Section 4.3. However we recall them here: $th(P) = \{\alpha \mid \alpha \text{ is a formula over } \mathcal{L}_P \text{ and } P \vdash \alpha\}$ and $K(R) = \bigcap_{S \in R} S$ for a given set of sets of formulas R.

Lemma 9.5. *Let P be a program over a signature $\mathcal{L}$. Then $th(P) = K(SC_T(P)) = \bigcap_{T \in SC_T(P)} T$.* □

Proof of Lemma 9.5.

We know that $K(R) = \bigcap_{S \in R} S$ then $K(SC_T(P)) = \bigcap_{S \in SC_T(P)} S = th(P)$ since $th(P)$ is the minimal set contained in each set of $SC_T(P)$ (see Lemma 9.3). □

Example 9.3. Let us consider P_1 of Example 9.1. We can see in Table 9.2 that $th(P_1) = K(SC_T(P_1)) = \bigcap_{T \in SC_T(P_1)} T = \{a \wedge a, a \to a, b \to b, a \to (b \to a), (a \wedge a) \to a, \ldots\}$. □

Lemma 9.6. *Let P_1 and P_2 be two programs over a signature $\mathcal{L}$. Then $th(P_1 \cup P_2) := K(SC_T(P_1) \cap SC_T(P_2)) = \bigcap_{T \in SC_T(P_1) \cap SC_T(P_2)} T$.* □

Proof of Lemma 9.6.

By Lemma 9.5, $th(P_1 \cup P_2) = K(SC_T(P_1 \cup P_2))$. However, by Lemma 9.1, $SC_T(P_1 \cup P_2) = SC_T(P_1) \cap SC_T(P_2)$, then by substitution, $th(P_1 \cup P_2) = K(SC_T(P_1) \cap SC_T(P_2))$. □

Example 9.4. Let us consider Example 9.1. We can verify that $th(P_1 \cup P_2) = K(SC_T(P_1) \cap SC_T(P_2)) = \bigcap_{T \in SC_T(P_1) \cap SC_T(P_2)} T = \{a \wedge a, a \to a, b \to b, a \to (b \to a), (a \wedge a) \to a, \ldots\}$.

9.3 Finding variants of answer sets from semantic contents

In order to find some variants of answer sets from the semantic contents of a program, $SC(P)$, we consider a particular subset of entries of $SC(P)$, denoted by $SC(P)^R$. The pairs of this particular subset are selected from $SC(P)$ based on a subset R of $\mathcal{L}_P$ together with the sets $\neg\mathcal{L}_P$ and $\neg\neg\mathcal{L}_P$. Then, the first entry of each pair in $SC(P)^R$ should be a subset of $(R \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$ and the second entry corresponds to the set of literals in the second entry of the original entry in $SC(P)$. Additionally, this particular subset, $SC(P)^R$, will be called *restricted semantic contents of a program P w.r.t. the set R.*

Definition 9.2. Let P be a program. Let $SC(P)$ be the semantic contents of P and $R \subseteq \mathcal{L}_P$. We define the *restricted semantic contents of a program P w.r.t. the set R*, denoted as $SC(P)^R$, as follows: $SC(P)^R = \{\langle S, L\rangle \mid \langle S, T\rangle \in SC(P) \ \wedge \ S \subseteq (R \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P) \ \wedge \ L = literals(T)\}$.

Example 9.5. Let P be the program:

$$e \leftarrow \neg d.$$
$$d \leftarrow \neg e.$$
$$a \leftarrow \neg b.$$

Then $\mathcal{L}_P = \{a, b, d, e\}$ and we are going to consider $R = \emptyset$. Then $SC(P)^\emptyset = \{\langle S, L\rangle \mid \langle S, T\rangle \in SC(P) \ \wedge \ S \subseteq \{\neg a, \neg b, \neg d, \neg e, \neg\neg a, \neg\neg b, \neg\neg d, \neg\neg e\} \ \wedge \ L = literals(T)\}$. Table 9.4 shows some of the entries of the pairs of $SC(P)^\emptyset$.

Based on the restricted semantic contents of a program P w.r.t. the set R, $SC(P)^R$, we introduce the definitions of partial pair, partial model, model pair and model. Then, we shall show how to obtain some answer set semantics based on these definitions.

S	L
$\{\neg b, \neg e\}$	$\{\neg b, \neg e, a, d\}$
$\{\neg b, \neg d\}$	$\{\neg b, \neg d, a, e\}$
$\{\neg b\}$	$\{\neg b, a\}$
$\{\neg a, \neg e\}$	$\{\neg a, \neg e, d\}$
$\{\neg a, \neg d\}$	$\{\neg a, \neg d, e\}$
$\{\neg d\}$	$\{\neg d, e\}$
$\{\neg e\}$	$\{\neg e, d\}$
$\{\neg a\}$	$\{\neg a\}$

Table 9.4: $SC(P)^{\emptyset}$ of program $P = \{a \leftarrow \neg b,\ e \leftarrow \neg d,\ d \leftarrow \neg e\}$.

Definition 9.3. Let P be a program, $R \subseteq \mathcal{L}_P$ and $SC(P)^R$ be the restricted semantic contents of a program P w.r.t. the set R. We define the following:

1. $\langle S, M\rangle$ is a $partialPair(R)$ of P if $\langle S, M\rangle \in SC(P)^R$.
2. M is a $partialModel(R)$ of P if $\langle S, M\rangle \in SC(P)^R$.
3. $\langle S, M\rangle$ is a $modelPair(R)$ of P if $\langle S, M\rangle \in SC(P)^R$ such that M is complete.[2]
4. M is a $model(R)$ of P if $\langle S, M\rangle \in SC(P)^R$ and M is complete.
5. A *semantics* of a program P, denoted as $SEM(P)$, is a set of $partiaModel(R)$ of P □

Example 9.6. We can verify from Table 9.4 of Example 9.5 that,

1. some of the $partialPair(\emptyset)$'s of P are : $\langle\{\neg b, \neg e\}, \{\neg b, \neg e, a, d\}\rangle$, $\langle\{\neg b\}, \{\neg b, a\}\rangle$ $\langle\{\neg a, \neg e\}, \{\neg a, \neg e, d\}\rangle$,
2. the $partialModel(\emptyset)$'s of P are: $\{\neg b, \neg e, a, d\}, \{\neg b, \neg d, a, e\}, \{\neg b, a\}, \{\neg a, \neg e, d\}$, $\{\neg a, \neg d, e\}, \{\neg d, e\}, \{\neg e, d\}, \{\neg a\}$,
3. the $modelPair(\emptyset)$'s of P are: $\langle\{\neg b, \neg e\}, \{\neg b, \neg e, a, d\}\rangle$ and $\langle\{\neg b, \neg d\}, \{\neg b, \neg d, a, e\}\rangle$,

[2] A finite set of formulas P is said to be *complete* [27] if, for any formula A of P, either $P \vdash A$ or $P \vdash \neg A$.

4. the $model(\emptyset)$'s of P are: $\{\neg b, \neg e, a, d\}$ and $\{\neg b, \neg d, a, e\}$. □

Now, in the following subsections we are going to show how to obtain some answer set semantics based on Definition 9.3.

9.3.1 Answer sets

If a program has at least one answer set then, in order to obtain the answer sets of a program, we shall use the definition of $model(R)$ given in Definition 9.3 such that the set R is equal to the empty set.

Lemma 9.7. *Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. Then M is an answer set of P iff there exists M', a $model(\emptyset)$ of P, such that $M = pos(M')$.* □

Proof of Lemma 9.7.
We split the proof into two parts:

1. Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. If M is an answer set of P then there exists M', a $model(\emptyset)$ of P, such that $M = pos(M')$.

2. Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. If there exists M', a $model(\emptyset)$ of P, then $pos(M')$ is an answer set of P.

We start proving **(1)**:

Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. Let M an answer set of P (see Theorem 4.1), i.e.,

$$P \cup \neg(\mathcal{L}_P \setminus M) \cup \neg\neg M \Vdash_{\mathrm{I}} M \tag{9.2}$$

Then by definition of $model(\emptyset)$ of P (Definition 9.3) we have to prove that $\exists\langle S, M'\rangle \in SC(P)^{\emptyset}$ such that M' is complete and $M = pos(M')$,

i.e., by definition of $SC(P)^{\emptyset}$ (Definition 9.2) we have to prove that $\exists\langle S, M'\rangle$ such that $\exists\langle S, T\rangle \in SC(P)$ where $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete and $M = pos(M')$,

i.e., by definition of $SC(P)$ (Definition 9.1) we have to prove that $\exists\langle S, M'\rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ where $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ is consistent, M' is complete and $M = pos(M')$.

Since $M = pos(M)$ and $\neg(\mathcal{L}_P \setminus M) = neg(M)$ then we can rewrite (9.2) as follows:

$$P \cup neg(M) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$$

By monotonicity, we can have

$$P \cup neg(M) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M) \cup neg(M) \tag{9.3}$$

Let $S = neg(M) \cup \neg\neg pos(M)$ and $M' = pos(M) \cup neg(M)$ then we can rewrite (9.3) as follows

$$P \cup S \Vdash_{\mathrm{I}} M' \tag{9.4}$$

Let $T = \{X | P \cup S \vdash X\}$, then by (9.4) $M' \subseteq T$. Additionally, by construction $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete and $pos(M') = M$. Finally, by definition of $\Vdash_{\mathrm{I}}$, $P \cup S$ is consistent.

Now, we prove **(2)**:

Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. Let M' a $model(\emptyset)$ of P, i.e., by definition of $model(\emptyset)$ of P (Definition 9.3) $\exists\langle S, M'\rangle \in SC(P)^{\emptyset}$ such that M' is complete,

i.e., by definition of $SC(P)^{\emptyset}$ (Definition 9.2) $\exists\langle S, M'\rangle$ such that $\exists\langle S, T\rangle \in SC(P)$ with $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, and M' complete,

i.e., by definition of $SC(P)$ (Definition 9.1) $\exists\langle S, M'\rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ with $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete.

Then, by definition of answer set of P (see Theorem 4.1) we have to prove that

$$P \cup \neg(\mathcal{L}_P \setminus pos(M')) \cup \neg\neg pos(M') \Vdash_{\mathrm{I}} pos(M')$$

By hypothesis $\exists\langle S, M'\rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ with $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete. Then in particular, $P \cup S \vdash M'$ with $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete.

Since $pos(M') \subseteq (M')$, then $P \cup S \vdash pos(M')$ with $S \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete.

Clearly, $P \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \vdash pos(M')$. Finally, since $P \cup S$ is consistent we have that $P \cup \neg(\mathcal{L}_P \setminus pos(M')) \cup \neg\neg pos(M') \Vdash_{\mathrm{I}} pos(M')$. □

For instance, in Example 9.6 the $model(\emptyset)$'s of P are $\{\neg b, \neg e, a, d\}$ and $\{\neg b, \neg d, a, e\}$ then the answer sets of P are $\{a, d\}$ and $\{a, e\}$.

9.3.2 Partial answer sets

We know that some programs do not have answer sets. However, we would like to know at least "the greatest amount of information about the literals" that this program entail. For this reason, we introduce the *partial answer sets* of a program. The partial answer sets of a program are based on the $partial\_model(R)$'s of the program such that the set R is equal to the empty set. Moreover, partial answer sets capture the idea of "the greatest amount of information about the literals" by using the concept of set cardinality. Hence, a *partial answer set* of a program P will be a $partialModel(\emptyset)$ of P with maximum cardinality among the $partialModel(\emptyset)$'s of P.

S	L
$\{\neg b\}$	$\{\neg b, a\}$
$\{\neg a\}$	$\{\neg a\}$

Table 9.5: $SC(P)^{\emptyset}$ of program $P = \{a \leftarrow \neg b,\ c \leftarrow \neg c\}$.

Definition 9.4. Let P be a program and M a $partialModel(\emptyset)$ of P. Then $pos(M)$ is a partial answer set of P iff there does not exists a $partialModel(\emptyset)$ of P, such that $|M'| > |M|$. □

Example 9.7. Let P be the program:

$$a \leftarrow \neg b.$$

$$c \leftarrow \neg c.$$

We can see that $\mathcal{L}_P = \{a, b, c\}$ and that program P does not have answer sets. However, we will obtain its partial answer sets in order to know at least "the greatest amount of information about the literals" that program P entail. Table 9.5 shows the entries for some pairs of $SC(P)^{\emptyset}$. We can verify from Table 9.5 that $\{\neg a\}$ and $\{\neg b, a\}$ are the $partialModel(\emptyset)$'s of P but the only partial answer set of P is $\{a\}$ since it corresponds to the $partialModel(\emptyset)$ $\{\neg b, a\}$ and $|\{\neg b, a\}| > |\{\neg a\}|$. As the intuition about partial answer sets suggest, $\{\neg b, a\}$ gives us more information about the literals of the program than the rest of $partialModel(\emptyset)$'s. □

Another example about partial answer sets results if we consider again Example 9.5. We can see that the $partialModel(\emptyset)$'s with maximum cardinality are $\{\neg b, \neg e, a, d\}$ and $\{\neg b, \neg d, a, e\}$, hence $\{a, d\}$ and $\{a, e\}$ are the partial answer sets of program P. As we mentioned before, we are using the concept of set cardinality to select which $partialModel(\emptyset)$ corresponds to a partial answer set. We also could suggest to use the concept of set inclusion to obtain the partial answer sets. However, since we are interested in the greatest amount of information about the literals, set inclusion does

not work as we would expect. For instance, the $partialModel(\emptyset)$ $\{\neg b, \neg e, a, d\}$ (one of the partial answer sets) is not comparable to the $partialModel(\emptyset)$ $\{\neg a, \neg e, d\}$ when we use set inclusion. Then $\{a, d\}$ and $\{d\}$ would be partial answer sets. However, the second one gives less information about the literals of the program than the first one.

We also can see that the partial answer sets of P in Example 9.5 correspond to the answers sets of P. Then this example shows how our definition of partial answer sets agrees with the definition of answer sets when the program has at least one answer set. The following lemma shows that our definition of partial answer sets reflects naturally its relationship with the definition of answer sets.

Lemma 9.8. *Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. If M is an answer set of P then M is a partial answer set of P.* □

Proof of Lemma 9.8.

Let P be a stable consistent program (P has at least one answer set) and $M \subseteq \mathcal{L}_P$. Let M an answer set of P, i.e., by Lemma 9.7 exists M' a $model(\emptyset)$ of P such that $M = pos(M')$, i.e., by definition of $model(\emptyset)$ of P (Definition 9.3) exists $\langle S, M' \rangle \in SC(P)^{\emptyset}$ with M' complete such that $M = pos(M')$.

By definition of partial answer set of P (Definition 9.4), we have to prove that

$\exists M'$ a $partialModel(\emptyset)$ of P such that $M = pos(M')$, and

$\nexists$ a M'', a $partialModel(\emptyset)$ of P, such that $|M''| > |M'|$, i.e.,

by definition of $partialModel(\emptyset)$ of P (Definition 9.3), we have to prove that

$\exists \langle S, M' \rangle \in SC(P)^{\emptyset}$ such that $M = pos(M')$, and

$\nexists$ a M'', a $partialModel(\emptyset)$ of P, such that $|M''| > |M'|$.

By hypothesis, we have seen that exists $\langle S, M' \rangle \in SC(P)^{\emptyset}$ such that $M = pos(M')$. We are going to prove by contradiction that $\nexists$ a M'', a $partialModel(\emptyset)$ of P, such that $|M''| > |M'|$. Let M'' be a $partialModel(\emptyset)$ of P, such that $|M''| > |M'|$. Then

$|M''| > |\mathcal{L}_P|$ since by hypothesis M' is complete, a contradiction. □

We also want to remark that our definition of partial answer sets is different from the definition of partial stable model semantics introduced in [39] and it is also different from the definition of partial models given in [44].

Using Partial answer sets to restore consistency

Now we want to present an example in a planning domain where partial answer sets can be useful. We think that in case that a program has no answer sets, partial answer sets could help to detect the cause of inconsistency. In particular, in planning domains partial answer sets could correspond to partial plans. Then, the idea is to analyze the partial plans and review the possible causes of inconsistency. The following example illustrates this idea presenting a short planing problem.

Example 9.8. Somebody plans to submit his/her paper to a congress and present it. Additionally, this person receives an invitation to present his/her paper. Then we can express this problem as a planning problem where there are three fluents: $noMoneyProblems$, $invitedPaper$ and $inCongress$; there are only one action: $makeInscription$; and the goal is that the person assists to the congress, i. e., the fluent $inCongress$ holds. Then a brief description of the planning problem could be the following:

% If it is an invitedPaper at time 1 then inCongress holds at time 2

$holds(inCongress, 2) \leftarrow holds(invitedPaper, 1).$

%When action payInscription occurs at time 1 causes inCongress at time 2.

$holds(inCongress, 2) \leftarrow occurs(makeInscription, 1).$

%Normally, if there is not evidence about action makeInscription

%occurs at time 1 then inCongress does not holds at time 2.

$holds(neg(inCongress), 2) \leftarrow \neg occurs(makeInscription, 1).$

%InvitedPaper holds at time 1.

$holds(invitedPaper, 1).$

%If there is no money problems then action makeInscription is possible.

$occurs(makeInscription, 1) \leftarrow holds(noMoneyProblems, 1).$

We can verify that the program does not has answer sets, i.e., there is an inconsistency. In order to detect the inconsistency we could analyze the partial answer sets of this program:

$$\{holds(invitedPaper, 1), \quad holds(inCongress, 2)\}$$

The partial answer set could suggest that there is a contradiction with

$$holds(invitedPaper, 1) \quad or \quad holds(inCongress, 2).$$

Then, in order to detect the possible causes of contradiction we could review the program and see that normally, if there is not evidence about action *makeInscription* occurs at time 1 then *inCongress* does not hold at time 2. Moreover, since *invitedPaper*

holds at time 1 then *inCongress* also holds at time 2. Hence, we can see that there is an inconsistency with $holds(neg(inCongress), 2)$ and $holds(inCongress, 2)$.

Then we could add new information to the program about $noMoneyProblems$ since we know that a person who has an invited paper does not have to pay inscription and his/her inscription is generated automatically. Then, the program is the following:

$$
\begin{aligned}
holds(inCongress, 2) &\leftarrow holds(invitedPaper, 1).\\
holds(inCongress, 2) &\leftarrow occurs(makeInscription, 1).\\
-holds(inCongress, 2) &\leftarrow \neg occurs(makeInscription, 1).\\
holds(invitedPaper, 1).&\\
occurs(makeInscription, 1) &\leftarrow holds(noMoneyProblems, 1).\\
holds(noMoneyProblems, 1).&
\end{aligned}
$$

This new program has an answer set that corresponds to the desired plan:
$\{holds(noMoneyProblems, 1).\ holds(invitedPaper, 1).\ occurs(makeInscription, 1).$
$holds(inCongress, 2)\}$ □

9.3.3 Generalized answer sets

We have presented in the Background section the Definition of *Generalized Answer Sets* (see Definition 4.2). As we have described, a generalized answer sets of a program is the answer set obtained by adding a set of atoms to the program.

Now, we are going to show how to obtain the generalized answer sets of a program in terms of the *generalized answer set pair*'s of the program. The intuition behind a *generalized answer set pair* of a program is to denote explicitly the set of abducible atoms added to the program and the generalized answer set of the program as a pair. Moreover, a *generalized answer set pair* of a program P is based on a $modelPair(R)$ of P where the set R includes a subset of $\mathcal{L}_P$. We denote this subset of $\mathcal{L}_P$ as EA and we

call it the *explicit abducibles*.

Definition 9.5. Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Then $\langle S, M\rangle$ is a *generalized answer set pair* of P w.r.t. EA if $\langle S, M\rangle$ is a $modelPair(EA)$ of P. □

In Example 4.11 of Section 4.7 we presented the program P_1 to illustrate the abductive logic programs and the minimal generalized answer sets. Now in this example, we consider again program P_1 in order to see if the results using the semantic contents of the program agree with the results of Example 4.11.

Example 9.9. Let P_1 be the program:

$$p \leftarrow \neg q.$$
$$r \leftarrow \neg s.$$
$$q \leftarrow t.$$
$$s \leftarrow t.$$
$$\leftarrow p, r.$$

We can see that program P_1 does not have answer sets. However, if we consider $EA = \mathcal{L}_{P_1} = \{p, q, r, s, t\}$ then Table 9.6 shows some of the entries of $SC(P_1)^{EA}$. Additionally, all of these pairs correspond to $modelPair(EA)$'s of P_1. Hence, all of them are *generalized answer set pair*'s of P w.r.t. EA. □

Then, it is possible to use Definition 9.5 to obtain the generalized answer sets of a program.

Lemma 9.9. *Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Then M is a generalized answer set of P w.r.t. EA iff there exists $\langle S, M'\rangle$, a generalized answer set pair of P w.r.t. EA, such that $M = pos(M')$.* □

Proof of Lemma 9.9.

We split the proof into two parts:

1. Let P be program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. If M is a generalized answer set of P w.r.t. EA then there exists $\langle S, M' \rangle$, a *generalized answer set pair* of P w.r.t. EA, such that $M = pos(M')$.

2. Let P be program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. If $\langle S, M' \rangle$ is a *generalized answer set pair* of P w.r.t. EA then $pos(M')$ is a generalized answer set of P w.r.t. EA.

We start proving **(1)**:

Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let M a generalized answer set of P w.r.t. EA, i.e., by definition of generalized answer set of P w.r.t. EA (see Definition 4.2), $\exists \Delta \subseteq EA$ such that M is an answer set of $P \cup \Delta$, i.e., by definition of answer set of P (see Theorem 4.1), $\exists \Delta \subseteq EA$ such that

$$P \cup \Delta \cup \neg(\mathcal{L}_P \setminus M) \cup \neg\neg M \Vdash_{\mathrm{I}} M \tag{9.5}$$

We have to prove that there exists $\langle S, M' \rangle$, a *generalized answer set pair* of P w.r.t. EA, such that $M = pos(M')$, i.e.,

by definition of *generalized answer set pair* of P w.r.t. EA (Definition 9.5), we have to prove that exists $\langle S, M' \rangle$, a $modelPair(EA)$ of P, such that $M = pos(M')$, i.e.,

by definition of $modelPair(EA)$ of P w.r.t. EA (Definition 9.3), we have to prove that $\exists \langle S, M' \rangle \in SC(P)^{EA}$ such that M' is complete and $M = pos(M')$,

i.e., by definition of $SC(P)^{EA}$ (Definition 9.2) we have to prove that $\exists \langle S, M' \rangle$ such that $\exists \langle S, T \rangle \in SC(P)$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete and $M = pos(M')$,

i.e., by definition of $SC(P)$ (Definition 9.1) we have to prove that $\exists \langle S, M' \rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ is consistent, M' is complete and $M = pos(M')$.

Since $M = pos(M)$ and $\neg(\mathcal{L}_P \setminus M) = neg(M)$ then we can rewrite (9.5) as follows:

$$P \cup \Delta \cup neg(M) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$$

By monotonicity, we can have

$$P \cup \Delta \cup neg(M) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M) \cup neg(M) \tag{9.6}$$

Let $S = \Delta \cup neg(M) \cup \neg\neg pos(M)$ and $M' = pos(M) \cup neg(M)$ then we can rewrite (9.6) as follows

$$P \cup S \Vdash_{\mathrm{I}} M' \tag{9.7}$$

Let $T = \{X | P \cup S \vdash X\}$, then by (9.7) $M' \subseteq T$. Additionally, by construction $S \subseteq (\Delta \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete and $pos(M') = M$. Finally, by definition of $\Vdash_{\mathrm{I}}$, $P \cup S$ is consistent.

Now, we prove **(2)**:

Let P be program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let $\langle S, M' \rangle$ a *generalized answer set pair* of P w.r.t. EA,

i.e., by definition of *generalized answer set pair* of P w.r.t. EA (Definition 9.5) exists $\langle S, M' \rangle$, a $modelPair(EA)$ of P,

i.e., by definition of $modelPair(EA)$ of P w.r.t. EA (Definition 9.3), $\exists \langle S, M' \rangle \in SC(P)^{EA}$ such that M' is complete,

i.e., by definition of $SC(P)^{EA}$ (Definition 9.2), $\exists \langle S, M' \rangle$ such that $\exists \langle S, T \rangle \in SC(P)$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$ and M' is complete,

i.e., by definition of $SC(P)$ (Definition 9.1), $\exists \langle S, M' \rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent and M' complete.

Then by definition of generalized answer set of P w.r.t. EA (see Definition 4.2), we have to prove that $\exists \Delta \subseteq EA$ such that $pos(M')$ is an answer set of $P \cup \Delta$,

$\langle S$	,	$L\rangle$	$pos(L)$
$\langle\{q, \neg s\}$	,	$\{q, \neg s, r, \neg t, \neg p\}\rangle$,	$\{q, r\}$
$\langle\{s, \neg q\}$	,	$\{s, \neg q, p, \neg r, \neg t\}\rangle$,	$\{s, p\}$
$\langle\{p, s, \neg q\}$	,	$\{s, \neg q, p, \neg r, \neg t\}\rangle$,	$\{s, p\}$
$\langle\{t, \neg r, \neg p\}$	,	$\{t, \neg r, \neg p, s, q\}\rangle$,	$\{t, s, q\}$

Table 9.6: Some entries of $SC(P_1)^{EA}$ of program P_1 with $EA = \mathcal{L}_{P_1}$.

i.e., by definition of answer set of P (see Theorem 4.1), we have to prove that $\exists\Delta \subseteq EA$ such that

$$P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M')) \cup \neg\neg pos(M') \Vdash_{\mathrm{I}} pos(M')$$

By hypothesis, $\exists\langle S, M'\rangle$ such that $\exists T = \{X | P \cup S \vdash X\}$ with $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete. Then in particular, $P \cup S \vdash M'$ with $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, and M' complete.

Since $pos(M') \subseteq (M')$ then $P \cup S \vdash pos(M')$ with $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ in consistent, and M' complete.

Let $\Delta \subseteq EA$ be the minimal set such that $P \cup \Delta \cup S' \vdash pos(M')$ with $S' \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ in consistent, and M' complete.

Clearly, $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M')) \cup \neg\neg pos(M') \vdash pos(M')$. Finally, since $P \cup S$ is consistent then whe have $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M')) \cup \neg\neg pos(M') \Vdash_{\mathrm{I}} pos(M')$. □

For instance, if we consider again the program of Example 9.9, we can verify for each entry in Table 9.6 that $pos(L)$ is a generalized answer sets of program P_1.

9.3.4 Minimal generalized answer sets

We have presented in the Background section the Definition of *Minimal Generalized Answer Sets* (see Definition 4.4). Now, we are going to show how we can obtain the minimal generalized answer sets in terms of the generalized answer set pairs of a program

w.r.t. the set EA.

First, we introduce an order among the generalized answer set pairs of a program w.r.t. the set EA, denoted as $<_{EA}$. Then, we show how to obtain the minimal generalized answers sets of the program using this order.

Definition 9.6. Let P be a program and $EA \subseteq \mathcal{L}$. Let $\langle S, L\rangle$ and $\langle S', L'\rangle$ be two *generalized answer set pairs* of P w.r.t. EA. Then, we define $\langle S, L\rangle <_{EA} \langle S', L'\rangle$ if $pos(S) \subset pos(S')$. □

Note that $<_{EA}$ is a strict order over SC.

Lemma 9.10. *Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Then M is a minimal generalized answer set of P w.r.t. EA iff there exists $\langle S, M'\rangle$, a minimal generalized answer set pair of P w.r.t. the ordering $<_{EA}$, such that $M = pos(M')$.* □

Proof of Lemma 9.10.
We split the proof into two parts:

1. Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let M be a minimal generalized answer set of P w.r.t. EA then exists $\langle S, M'\rangle$ a minimal generalized answer set pair of P w.r.t. the ordering $<_{EA}$ such that $M = pos(M')$.

2. Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let $\langle S, M'\rangle$ be a minimal generalized answer set pair of P w.r.t. EA and minimal w.r.t. the ordering $<_{EA}$ then $pos(M')$ is a minimal generalized answer set of P w.r.t. EA.

We start proving **(1)**:

Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let M be a minimal generalized answer set of P w.r.t. EA,

i.e, by definition of minimal generalized answer sets (see Definition 4.4), $\exists \Delta \subseteq EA$ such that M is an answer set of $P \cup \Delta$ and $\nexists \Delta' \subset \Delta$ such that $P \cup \Delta'$ has an answer set of $P \cup \Delta'$,

i.e., by definition of answer set of $P \cup \Delta$ (see Theorem 4.1),

$\exists \Delta \subseteq EA$ such that $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$

and

$\nexists \Delta' \subseteq \Delta$ such that $P \cup \Delta' \cup \neg(\mathcal{L}_P \setminus pos(M'')) \cup \neg\neg pos(M'') \Vdash_{\mathrm{I}} pos(M'')$.

By definition of minimal generalized answer set pair (Definition 9.6), we have to prove two things:

- **(a)** exists $\langle S, M' \rangle$, a generalized answer set pair of P w.r.t. EA, such that $pos(M')$ is a generalized answer set of P w.r.t. EA, and
- **(b)** $\nexists \langle S', M'' \rangle$, a generalized answer set pair of P w.r.t. EA, such that $pos(S') \subset pos(S)$.

In order to prove **(a)**, we know by hypothesis that M is a generalized answer set of P w.r.t. EA. Then by Lemma 9.9, M is a generalized answer set of P w.r.t. EA iff there exists $\langle S, M' \rangle$, a *generalized answer set pair* of P w.r.t. EA, such that $M = pos(M')$.

We are going to prove **(b)** by contradiction. Let $\langle S, M' \rangle$ be a generalized answer set pair of P w.r.t. EA, such that $pos(M')$ is a generalized answer set of P w.r.t. EA, and let $\langle S', M'' \rangle$ be a generalized answer set pair of P w.r.t. EA, such that $pos(S') \subset pos(S)$.

Then by definition of generalized answer set pair of P w.r.t. EA (Definition 9.5): $\langle S, M' \rangle$ is a $modelPair(EA)$ of P and $\langle S', M'' \rangle$ is a $modelPair(EA)$ of P such that $pos(S') \subset pos(S)$.

Then by definition of $modelPair(EA)$ of P (Definition 9.3): $\langle S, M' \rangle \in SC(P)^{EA}$ such that M' is complete and $\langle S', M'' \rangle \in SC(P)^{EA}$ such that M'' is complete and $pos(S') \subset pos(S)$.

Then by definition of $SC(P)^{EA}$ (Definition 9.2):

$\exists\langle S, T\rangle \in SC(P)$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete, and

$\exists\langle S', T'\rangle \in SC(P)$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, M'' is complete, and $pos(S') \subset pos(S)$.

Then by definition of $SC(P)$ (Definition 9.1):

$\exists T = \{X | P \cup S \vdash X\}$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and

$\exists T' = \{X | P \cup S' \vdash X\}$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, and $pos(S') \subset pos(S)$.

In particular:

$P \cup S \vdash M'$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and

$P \cup S' \vdash M''$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, and $pos(S') \subset pos(S)$.

Let $\Delta = pos(S)$ and $\Delta' = pos(S')$ then:

$P \cup \Delta \cup R \vdash M'$ where $R \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and

$P \cup \Delta' \cup R' \vdash M''$ where $R' \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, and $\Delta' \subset \Delta$.

Clearly, $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$ and

$P \cup \Delta' \cup \neg(\mathcal{L}_P \setminus pos(M'')) \cup \neg\neg pos(M'') \Vdash_{\mathrm{I}} pos(M'')$ such that $\Delta' \subset \Delta$, a contradiction of hypothesis.

Now, we prove **(2)**:

Let P be a program, $M \subseteq \mathcal{L}_P$ and $EA \subseteq \mathcal{L}_P$. Let $\langle S, M'\rangle$ be a minimal generalized answer set pair of P w.r.t. EA and the ordering $<_{EA}$.

By definition of minimal generalized answer set of P (Definition 4.4), we have to prove two things:

- **(a)** $\exists \Delta \subseteq EA$ such that $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$, i.e. $pos(M)$ is an answer set of $P \cup \Delta$,
- **(b)** $\nexists \Delta' \subseteq \Delta$ such that $P \cup \Delta' \cup \neg(\mathcal{L}_P \setminus pos(M'')) \cup \neg\neg pos(M'') \Vdash_{\mathrm{I}} pos(M'')$.

In order to prove **(a)**, we know by hypothesis that $\langle S, M' \rangle$ is a minimal generalized answer set pair of P w.r.t. EA and the ordering $<_{EA}$. Then by Lemma 9.9, $pos(M')$ is a generalized answer set of P .

We are going to prove **(b)**.

By hypothesis and definition of order $<_{EA}$ (Definition 9.6), if $\langle S, M' \rangle$ is a minimal generalized answer set pair of P w.r.t. EA and the ordering $<_{EA}$ then $\langle S, M' \rangle$ is a generalized answer set pair of P w.r.t. EA, and $\nexists \langle S', M'' \rangle$ generalized answer set pair of P w.r.t. EA, such that $pos(S') \subset pos(S)$,

i.e., if $\langle S, M' \rangle$ is a minimal generalized answer set pair of P w.r.t. EA and the ordering $<_{EA}$ then $\langle S, M' \rangle$ is a generalized answer set pair of P w.r.t. EA, and if exists $\langle S', M'' \rangle$ a generalized answer set pair of P w.r.t. EA, then $pos(S) \subset pos(S')$,

i.e., by definition of generalized answer set pair of P w.r.t. EA (Definition 9.5), $\langle S, M' \rangle$ is a $modelPair(EA)$ of P and if exists $\langle S', M'' \rangle$ is a $modelPair(EA)$ of P then $pos(S) \subset pos(S')$,

i.e., by definition of $modelPair(EA)$ of P (Definition 9.3), $\langle S, M' \rangle \in SC(P)^{EA}$ such that M' is complete and if exists $\langle S', M'' \rangle \in SC(P)^{EA}$ such that M'' is complete then $pos(S) \subset pos(S')$,

i.e., by definition of $SC(P)^{EA}$ (Definition 9.2), exists $\langle S, T \rangle \in SC(P)$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, M' is complete, and

if exists $\langle S', T' \rangle \in SC(P)$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, M'' is complete, then $pos(S) \subset pos(S')$,

i.e., by definition of $SC(P)$ (Definition 9.1), exists $T = \{X | P \cup S \vdash X\}$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and if exists $T' = \{X | P \cup S' \vdash X\}$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, then $pos(S) \subset pos(S')$.

Then, in particular:
$P \cup S \vdash M'$ where $S \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and if exists $P \cup S' \vdash M''$ where $S' \subseteq (EA \cup \neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, then $pos(S) \subset pos(S')$.

Let $\Delta = pos(S)$ and $\Delta' = pos(S')$ then:
$P \cup \Delta \cup R \vdash M'$ where $R \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M' = literals(T)$, $P \cup S$ consistent, M' is complete, and
if exists $P \cup \Delta' \cup R' \vdash M''$ where $R' \subseteq (\neg\mathcal{L}_P \cup \neg\neg\mathcal{L}_P)$, $M'' = literals(T')$, $P \cup S$ consistent, M'' is complete, then $\Delta \subset \Delta'$.

Clearly, $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$ and
if exists $P \cup \Delta' \cup \neg(\mathcal{L}_P \setminus pos(M'')) \cup \neg\neg pos(M'') \Vdash_{\mathrm{I}} pos(M'')$ then $\Delta \subset \Delta'$,

i.e., $P \cup \Delta \cup \neg(\mathcal{L}_P \setminus pos(M)) \cup \neg\neg pos(M) \Vdash_{\mathrm{I}} pos(M)$ and
$\nexists \Delta' \subset \Delta$ such that $P \cup \Delta' \cup \neg(\mathcal{L}_P \setminus pos(M'')) \cup \neg\neg pos(M'') \Vdash_{\mathrm{I}} pos(M'')$. □

For instance, $\{s\}$ is a minimal generalized answer set of program P_1 in Example 9.9 since $\{s, \neg q\}$ is a minimal entry.

9.3.5 Compositionality of programs

In this section we introduce *compositionality* of programs in terms of their Semantic Contents. The idea is to combine the knowledge represented in two diferents programs

to make conclusions of it. This result is not fundamental for this work and we can see it in detail in [47, 37, 48].

9.4 Conclusions

In order to have a mathematical structure useful to express from it in a uniform way the different answer set semantics, in this chapter we introduced the notion of Semantic Contents of a program. Once we have constructed the Semantic Contents of a program we show how to find from it in a uniform way the standard definition of answer sets and some of the variants of answer sets such as generalized answer sets, minimal generalized answer sets, and an answer set semantics introduced in this section called partial answer sets.

Future work could be about how to use the semantics contents to obtain other answer set semantics such as preferred answers sets in terms of ordered disjunction programs.