TPLP **24** (3): 482–504, 2024.

doi:10.1017/S1471068424000012 First published online 5 February 2024

Epistemic Logic Programs: A Study of Some Properties[*]

STEFANIA COSTANTINI
DISIM - Università dell'Aquila, via Vetoio, L'Aquila, Italy
Gruppo Nazionale per il Calcolo Scientifico - INdAM, Roma, Italy
(*e-mail:* stefania.costantini@univaq.it)

ANDREA FORMISANO
DMIF - Università di Udine, via delle Scienze 206, Udine, Italy
Gruppo Nazionale per il Calcolo Scientifico - INdAM, Roma, Italy
(*e-mail:* andrea.formisano@uniud.it)

submitted 31 December 2022; revised 29 September 2023; accepted 8 January 2024

Abstract

Epistemic logic programs (ELPs), extend answer set programming (ASP) with epistemic operators. The semantics of such programs is provided in terms of *world views*, which are sets of belief sets, that is, syntactically, sets of sets of atoms. Different semantic approaches propose different characterizations of world views. Recent work has introduced semantic properties that should be met by any semantics for ELPs, like the *Epistemic Splitting Property*, that, if satisfied, allows to modularly compute world views in a bottom-up fashion, analogously to "traditional" ASP. We analyze the possibility of changing the perspective, shifting from a bottom-up to a top-down approach to splitting. We propose a basic top-down approach, which we prove to be equivalent to the bottom-up one. We then propose an extended approach, where our new definition: (i) is provably applicable to many of the existing semantics; (ii) operates similarly to "traditional" ASP; (iii) provably coincides under any semantics with the bottom-up notion of splitting at least on the class of *Epistemically Stratified Programs* (which are, intuitively, those where the use of epistemic operators is stratified); (iv) better adheres to common ASP programming methodology.

KEYWORDS: answer set programming, epistemic logic programs, epistemic splitting

1 Introduction

Epistemic logic programs (ELPs, in the following just *programs*, if not explicitly stated differently), were first introduced in Gelfond and Przymusinska (1991) and Gelfond (1994), and extend Answer Set Programs, defined under the Answer Set Semantics

[*] Research partially performed within the activities of the Action COST CA17124 "DigForASP", the Interdepartmental Project on AI (Strategic Plan UniUD-22-25), and the INdAM-GNCS project CUP E53C22001930001.

(Gelfond and Lifschitz 1988), with *epistemic operators* that are able to introspectively "look inside" a program's own semantics, which is defined in terms of its *answer sets* (cf. Fandinno *et al.* (2022) for a historical review of research on this topic). In fact, $\mathbf{K}A$ means that (ground) atom A is true in every answer set of the program Π where $\mathbf{K}A$ occurs. Related operators that can be defined in terms of $\mathbf{K}$ are the *possibility operator* $\mathbf{M}$ (not treated in this paper) where $\mathbf{M}A$ means that A is true in some of the answer sets of Π, and the *epistemic negation operator* **not**, where **not** A expresses that A *is not provably true*, meaning that A is false in at least one answer set of Π.

The semantics of ELPs is provided in terms of *world views*: instead of a unique set of answer sets (a unique "world view" in the new terminology) like in answer set programming (ASP), there is now a set of such sets. Each world view consistently satisfies (according to a given semantics) the epistemic expressions that appear in a given program. Many semantic approaches for ELPs have been introduced beyond the seminal work of Gelfond and Przymusinska (1991), among which we mention (Gelfond 2011; Truszczynski 2011; Fariñas del Cerro *et al.* 2015; Shen and Eiter 2016; Kahl and Leclerc 2018; Su 2019; Cabalar *et al.* 2019; Costantini and Formisano 2022; Su 2021).

Recent work extends to epistemic logic programming notions that have already been defined for ASP and that might prove useful in ELPs as well. In particular, Cabalar et al. consider *splitting* (introduced for ASP in Lifschitz and Turner (1994)), which allows a program to be seen as divided ("split") into two parts, the "top" and "bottom" in a principled way, that is, atoms occurring in the bottom can occur only in the body of rules in the top. This allows the answer sets of the program to be computed incrementally, in the following way: compute the answer sets of the bottom part and use them (one by one) to simplify the top part; then, compute the answer sets of the simplified top part; finally, the answer sets of the overall program are obtained as the union of each answer set of the bottom with the corresponding answer sets of the simplified top (such a procedure can be iterated, that is, the top and the bottom could in turn be split). Cabalar et al. then extend to ELPs the concept of splitting and the method of incremental calculation of the semantics (here, it is the world views that must be calculated). This is achieved by defining a notion of *Epistemic Splitting*, where top and bottom are defined with respect to the occurrence of epistemic operators, and a corresponding *Epistemic Splitting Property* (ESP), which is fulfilled by a semantics if it allows the world views to be computed bottom-up (a precise definition is seen below). Further, Cabalar et al. adapt properties of ASP to ELPs, which are implied by this property, namely, the fact that adding constraints leads to reduce the number of answer sets (*Subjective Constraint Monotonicity*), and *Foundedness*, meaning that atoms composing answer sets cannot have been derived through cyclic positive dependencies. Finally, they define the class of *Epistemically Stratified Programs* that, according to Cabalar *et al.* (2021, Th. 2), admit a unique world view (these programs are those where, intuitively, the use of epistemic operators is stratified). In substance, Cabalar et al. establish the properties that in their view a semantics should fulfill, and then they compare the existing semantics with respect to these properties.

In this paper, we explore a different stance: we analyze the possibility of changing the perspective about how to exploit a splitting, shifting from a bottom-up to a top-down approach. This applies in the first place to the ESP, of which we propose a reformulation allowing world views to be computed top-down. We then propose a substantial extension of the ESP, leading to a new approach that:

(i) is applicable to many of the existing semantics, while few of them fulfill the ESP as originally formulated;
(ii) operates similarly to splitting in "traditional" ASP;
(iii) provably coincides under any semantics with the bottom-up notion of splitting on a significant class of programs, including at least those which are *epistemically stratified*;
(iv) is compatible with common ASP programming practice, where one defines a problem solution (that would constitute the top) that will be merged with a problem instance (that would constitute the bottom).

The paper is organized as follows. In Sections 2 and 3, we recall ASP and ELPs. Section 4 reports some definitions from Cabalar *et al.* (2021) concerning useful properties of ELPs. In Section 5, we introduce some observations on ELPs that lead to formulate our proposal, treated in detail in Section 6. In Section 7, we state our main theorem and a relevant corollary. Finally, in Section 8, we conclude.

2 Answer set programming and answer set semantics

One can see an answer set program (for short, ASP program) as a set of statements that specify a problem, where each answer set represents a solution compatible with this specification. A *consistent* ASP program has one or more answer sets, while an *inconsistent* one has no answer sets, meaning that no solution can be found. Several well-developed freely available *answer set solvers* exist that compute the answer sets of a given program. Syntactically, an ASP program Π is a collection of *rules* of the form

$$A_1|\ldots|A_g \leftarrow\ L_1,\ldots,L_n.$$

where each A_i, $0 \leq i \leq g$, is an atom and $|$ indicates disjunction, and the L_is, $0 \leq i \leq n$, are literals (i.e., atoms or negated atoms of the form *not* A). The left-hand side and the right-hand side of the rule are called *head* and *body*, respectively.

A rule with an empty body is called a *fact*. As usual, the symbols $\top$ and $\bot$ denote the true and the false Boolean constants, respectively. The notation $A\,|\,B$ indicates disjunction, usable only in rule heads and, so, in facts.

A rule with an empty head (or, equivalently, with head $\bot$), of the form $\leftarrow L_1,...,L_n.$ or $\bot \leftarrow L_1,...,L_n.$, is a *constraint*, stating that literals $L_1,\ldots,L_n$ are not allowed to be simultaneously true in any answer set; the impossibility of fulfilling such kind of requirement is one of the reasons that makes a program inconsistent.

All extensions of ASP not explicitly mentioned above are not considered in this paper. We implicitly refer to the *ground* version of Π, which is obtained by replacing in all possible ways the variables occurring in Π with the constants occurring in Π itself, and is thus composed of ground atoms, that is, atoms that contain no variables.

The *answer set* (or *stable model*) semantics can be defined in several ways (Lifschitz 2010; Costantini and Formisano 2015). However, answer sets of a program Π, if any exists, are the supported minimal classical models of the program interpreted as a first-order theory in an obvious way. The original definition from Gelfond and Lifschitz (1988), introduced for programs where rule heads were limited to be single atoms, was in terms of the *GL-Operator* Γ. Given set of atoms I and program Π, $\Gamma_\Pi(I)$ is defined as the least Herbrand model of the program Π^I, namely, the Gelfond–Lifschitz reduct of Π w.r.t. I. The program Π^I is obtained from Π by:

1. removing all rules which contain a negative literal $not\, A$ such that $A \in I$; and
2. removing all negative literals from the remaining rules.

Since Π^I is a positive program, the least Herbrand model is guaranteed to exist and can be computed via the standard immediate consequence operator (Lloyd 1987). Then, I is an answer set whenever $\Gamma_\Pi(I) = I$.

This definition is then extended to the general case, involving disjunctive heads, by defining I to be an answer set of Π if it is a minimal model (w.r.t. set inclusion) of Π^I.

3 Epistemic logic programs

ELPs extend the syntax of ASP programs by introducing, in the body of rules, so-called *subjective literals* (w.r.t. the usual *objective literals*).[1] Such new literals are constructed via the *epistemic operator* **K** (disregarding without loss of generality the other epistemic operators). An ELP program is called *objective* if no subjective literals occur therein, that is, it is an ASP program. A constraint involving (also) subjective literals is called a *subjective constraint*, whereas one involving objective literals only is an *objective constraint*.

Let At be the set of atoms occurring (within either objective or subjective literals) in a given program Π, and $Atoms(r)$ be the set of atoms occurring in rule r. By some abuse of notation, we denote by $Atoms(X)$ the set of atoms occurring in X, whatever X is (a rule, a program, an expression, etc.). Let $Head(r)$ be the head of rule r and $Body_{obj}(r)$ (resp., $Body_{subj}(r)$) be the (possibly empty) set of objective (resp., subjective) literals occurring in the body of r. For simplicity, we often write $Head(r)$ and $Body_{obj}(r)$ in place of $Atoms(Head(r))$ and $Atoms(Body_{obj}(r))$, respectively, when the intended meaning is clear from the context. We call *subjective rules* those rules whose body is made of subjective literals only.

Literal $\mathbf{K}A$ intuitively means that the (ground) atom A is true in every answer set of the given program Π (it is a *cautious consequence* of Π). Since, as it turns out, whatever the semantic account one will choose there can be several sets of answer sets (called *world views*), the actual meaning of $\mathbf{K}A$ is that A is true in every answer set of some world view of Π. Each world view thus determines the truth value of all subjective literals in a program. There are several semantic approaches to ELPs, dictating in different ways how one finds the world views of a given program. Although all such approaches provide the same results in a set of basic examples, they (obviously) differ in others.

Formally, a semantics $\mathcal{S}$ is a function mapping an ELP program into sets of world views, that is, sets of sets of objective literals, where if Π is an objective program, then the unique member of $\mathcal{S}(\Pi)$ is the set of stable models of Π. Otherwise, each member of

[1] Nesting of subjective literals is not considered here.

$\mathcal{S}(\Pi)$ is an *$\mathcal{S}$-world view* of Π. (We will often write "world view" in place of "$\mathcal{S}$-world view" whenever mentioning the specific semantics will be irrelevant.) For an $\mathcal{S}$-world view W and a literal $\mathbf{K}L$, we write $W \models \mathbf{K}L$ if L is true in all elements of W.

For instance, for program $\{a \leftarrow not\, b,\ b \leftarrow not\, a,\ e \leftarrow not\, \mathbf{K}f,\ f \leftarrow not\, \mathbf{K}e\}$, every semantics returns two world views: $\{\{a, e\}, \{b, e\}\}$, where $\mathbf{K}e$ is true and $\mathbf{K}f$ is false, and $\{\{a, f\}, \{b, f\}\}$ where $\mathbf{K}f$ is true and $\mathbf{K}e$ is false. The presence of two answer sets in each world view is due to the cycle on objective atoms, whereas the presence of two world views is due to the cycle on subjective atoms (in general, the existence and number of world views are related to such cycles, see Costantini (2019) for a detailed discussion).

4 Epistemic logic programs: Useful properties

As argued by Cabalar et al., it would be useful if ELPs would enjoy, *mutatis mutandis*, properties similar to those of ASP programs. Hence, in their works, such useful properties are outlined and adapted, as we report (almost literally) below.

Drawing inspiration from the *Splitting Theorem* (Lifschitz and Turner 1994), an analogous property is defined for ELPs:

Definition 4.1 (*Epistemic splitting set* (*Cabalar et al. 2021*, *Definition 4*))
A set of atoms $U \subseteq At$ is said to be an epistemic splitting set of a program Π if for any rule r in Π one of the following conditions hold:

1. $Atoms(r) \subseteq U$;
2. $(Body_{obj}(r) \cup Head(r))\ \cap\ U = \emptyset$.

An epistemic splitting of Π is a pair $\langle B_U(\Pi), T_U(\Pi)\rangle$ such that $B_U(\Pi) \cap T_U(\Pi) = \emptyset$ and $B_U(\Pi) \cup T_U(\Pi) = \Pi$, and also, such that all rules in $B_U(\Pi)$ satisfy condition (1) and all rules in $T_U(\Pi)$ satisfy condition (2).

Intuitively, condition (2) means that the top program $T_U(\Pi)$ may refer to atoms in U which occur as heads of rules in the bottom $B_U(\Pi)$, only through epistemic operators.

Epistemic splitting can be used, similarly to "traditional" Lifschitz&Turner splitting, for iterative computation of world views. Indeed, Cabalar et al. (2021) propose to compute first the world views of the bottom program $B_U(\Pi)$ and, for each of them, simplify the corresponding subjective literals in the top part. Given an epistemic splitting set U for Π and a set of interpretations W, they define the subjective reduct of the top with respect to W and signature U, denoted by $E_U(\Pi, W)$. This operator considers all subjective literals L occurring in $T_U(\Pi)$, such that the atoms occurring in them belong to $B_U(\Pi)$. In particular, L will be substituted by $\top$ in $E_U(\Pi, W)$ if $W \models L$, and by $\bot$ otherwise. Thus, $E_U(\Pi, W)$ is a version of $T_U(\Pi)$ where some subjective literal, namely those referring to the bottom part of the program, have been simplified as illustrated.

Definition 4.2 ((*Cabalar et al. 2021*, *Definition 5*))
Given a semantics $\mathcal{S}$, a pair $\langle W_b, W_t\rangle$ is said to be an $\mathcal{S}$-solution of Π with respect to an epistemic splitting set U if W_b is an $\mathcal{S}$-world view of $B_U(\Pi)$ and W_t is an $\mathcal{S}$-world view of $E_U(\Pi, W_b)$.

The definition is parametric w.r.t. $\mathcal{S}$, as each different semantics $\mathcal{S}$ will define in its own way the $\mathcal{S}$-solutions for a given U and Π.

Definition 4.3
The WBT operation $W_b \sqcup W_t$ on sets of propositional interpretations W_b and W_t is defined as follows:

$$W_b \sqcup W_t = \{I_b \cup I_t | I_b \in W_b \wedge I_t \in W_t\}.$$

We report from Cabalar *et al.* (2021) the definition of the following property:

Property 4.1 (*Epistemic Splitting Property (ESP)*)
A semantics $\mathcal{S}$ satisfies the ESP if for any epistemic splitting set U of any program Π: W is an $\mathcal{S}$-world view of Π iff there is an $\mathcal{S}$-solution $\langle W_b, W_t \rangle$ of Π w.r.t. U such that $W = W_b \sqcup W_t$.

Then, under a semantics that satisfies ESP, world views of the entire program are obtainable as the union of world views of the bottom with world views of a simplified version of the top. The ESP implies *Subjective Constraint Monotonicity*, that is, for any epistemic program Π and any subjective constraint r, W is a world view of $\Pi \cup \{r\}$ iff both W is a world view of Π and W satisfies r.

As discussed in Cabalar *et al.* (2021), many semantics do not satisfy the ESP property, which is in fact satisfied only by the very first semantics of ELPs, proposed in Gelfond and Przymusinska (1991) and thus called G91 (and in some of its generalizations), and by Founded Autoepistemic Equilibrium Logic (FAEEL), defined in Cabalar *et al.* (2019).

Another interesting property is *foundedness*. Again, such a notion has been extended from objective programs (see Cabalar *et al.* (2021, Definition 15)). Intuitively, a set X of atoms is *unfounded* w.r.t. an (objective) program Π and an interpretation I, if for every $A \in X$ there is no rule r in Π by which A might be derived, without incurring in positive circularities and without forcing the derivation of more than one atom from the head of a disjunctive rule (see, e.g., Leone *et al.* (1997) for a formal definition). For ELPs, one has to consider that unfoundedness can originate also from positive dependencies on positive subjective literals, like, for example, in the program $A \leftarrow \mathbf{K}A$. Among the existing semantics, only FAEEL satisfies foundedness.

An interesting class of programs admitting a unique world view is characterized by the following definition.

Definition 4.4 (*Epistemic Stratification (Cabalar et al. 2021, Definition 6)*)
We say that an ELP Π is epistemically stratified if we can assign an integer mapping $\lambda : At \rightarrow N$ to each atom (occurring in the program) such that:

- $\lambda(a) = \lambda(b)$ for any rule $r \in \Pi$ and atoms $a, b \in (Atoms(r) \setminus Body_{subj}(r))$, and
- $\lambda(a) > \lambda(b)$ for any pair of atoms a, b for which there exists a rule $r \in \Pi$ with $a \in (Head(r) \cup Body_{obj}(r))$ and $b \in Body_{subj}(r)$.

5 Observations

The subdivision of an ELP into layers suggests that, in the upper layer, epistemic literals referring to the lower layer may be aimed at performing some kind of meta-reasoning about that layer. If the ESP is enforced, however, meta-level reasoning is in practice prevented. This is so because if the semantics satisfies such property, then, it is the lower layer that determines the truth value of the subjective literals that connect the two layers.

In fact, according to Property 4.1, through the simplification w.r.t. the answer sets of the lower layer, the upper layer is strongly (maybe sometimes too strongly) constrained.

For instance, let us consider the program $\Pi_0 = \{a \,|\, b,\ \bot \leftarrow \mathit{not}\,\mathbf{K}a\}$. We can see that, while the lower level $\{a \,|\, b\}$, considered as a program *per se*, has the unique world view $\{\{a\},\{b\}\}$, the overall program has no world views. In fact, $\mathbf{K}a$ does not hold in $\{\{a\},\{b\}\}$, thus the constraint is violated.

Notice, however, that the world view $\{\{a\}\}$ is instead accepted by some semantics, such as those defined in Gelfond (2011) and in Shen and Eiter (2016), that do not satisfy the ESP. This world view may be seen as corresponding to an approach where the upper layer, in order to retain consistency, "requires" the lower layer to entail a, which is absolutely feasible by choosing a over b in the disjunction.

From this perspective, the knowledge modeled by the upper layer is not just used to reject potential world views of the bottom level, but, instead, can affect the way in which they are composed, by filtering out some of the answer sets. This situation is reminiscent of what actually happens for ASP: consider the plain ASP program $\{a \,|\, b,\ c \leftarrow a,\ \leftarrow \mathit{not}\, c\}$, which has unique answer set $\{a, c\}$, originating from the answer set $\{a\}$ of the lower layer $\{a \,|\, b\}$.

We follow (for a long time) the line, amply represented in the literature, in which meta-reasoning is aimed not only at "observing" lower layer(s) but also at trying to influence them (cf. Costantini (2002) for a survey on meta-reasoning in Computational Logic); this by suitably enlarging and/or restricting, as an effect of meta-rules application, the set of possible consequences of such layer(s). We discuss at length this point of view, also proposing technical solutions and several examples, in Costantini and Formisano (2021).

In addition, let us notice that a common approach in logical declarative modeling of a problem consists of formalizing the problem domain as the "top" part of a program/theory. Then, such top part will be joined with a specific "bottom", representing the problem instance at hand, that may vary and might be, in general, unknown while defining the top.

Below is an example of what we mean (over-simplified and in "skeletal form" for the sake of conciseness), taken from the realm of digital investigations, that the authors have been studying in the context of the Action COST CA17124 DIGital FORensics: evidence Analysis via intelligent Systems and Practices (DigForASP). In the example, an investigation **must** be concluded with a judgment, that can be:

- of innocence if in no plausible scenario (i.e., in no answer set) evidence can be found of an involvement;
- of demonstrable guilt if in every possible scenario, the evidence of guilt can be found;
- of presumed innocence otherwise.

Clearly, the specification of the legal rules that can be used to draw conclusions, and then the details of each specific case will be modularly added whenever needed to this general "top" part. Thus, one can see a program composed of three layers: the top, and a bottom that can be further split into a middle layer containing legal rules, and the lowest layer with details of the case (see Costantini (2019) for more examples taken from this field).

The top layer is as follows:

$$
\begin{array}{l}
\mathit{judgement} \leftarrow \mathit{guilty}.\\
\mathit{judgement} \leftarrow \mathit{presumed\_innocent}.\\
\mathit{judgement} \leftarrow \mathit{innocent}.\\
\leftarrow \mathit{not}\ \mathbf{K}\ \mathit{judgement}.\\
\mathit{guilty} \leftarrow \mathit{provably\_guilty}.\\
\mathit{presumed\_innocent} \leftarrow \mathit{not}\ \mathit{provably\_guilty}.\\
\mathit{provably\_guilty} \leftarrow \mathbf{K}\ \mathit{sufficient\_evidence\_against}.\\
\mathit{innocent} \leftarrow \mathbf{K}\ \mathit{not}\ \mathit{sufficient\_evidence\_against}.
\end{array}
$$

Hence, a study of how the semantics of any resulting overall program might be built is in order here, as in many other practical cases: think, for example, of a top part comprising ontological definitions reusable in several application contexts. In fact, being able to compute and check a program's semantics only in dependence on each specific instance, does not seem to be elaboration-tolerant.

Therefore, we tried to understand whether the concept of splitting might be applied top-down, and how the existing semantics would behave in the new perspective.

6 Our proposal

Let us proceed step by step toward the new definition of *Top-down Epistemic Splitting Property* (TDESP). We first reformulate definitions related to ESP so that it can be applied also top-down, to obtain what we call Top-down Epistemic Splitting Property – Basic (TDESPB), showing that a semantics satisfies TDESPB if and only if it satisfies ESP. Thus, TDESPB provides a way of coping with incremental computation of world views more suitable to the examples mentioned earlier. We then perform some extensions, to obtain a more general TDESP that holds for a wider range of semantic approaches.

6.1 Preliminaries and key definitions

In our approach, the notion of splitting set remains the same, save for some details concerning subjective constraints. We need, in fact, to introduce preliminary assumptions on constraints. Notice that subjective literals may either occur in a subjective constraint directly or affect constraint's satisfaction through indirect dependencies, such as, for example, in the program $\bot \leftarrow a.\ a \leftarrow \mathbf{K}p$ (see Dix (1995) for a formal definition of direct and indirect dependencies). Without loss of generality, we exclude here indirect dependencies concerning subjective literals involved in constraints. Also, notice that, as it is well-known, a constraint can be represented as a unary odd cycle, that, for example, for $\bot \leftarrow \mathbf{K}p$ would be of the form $a \leftarrow \mathit{not}\, a, \mathbf{K}p$ (with a introduced as a fresh atom), or even (as discussed in depth in Costantini (2006)) as an odd cycle of any arity, of which $\mathbf{K}p$ is the unique *handle*. For the sake of simplicity, we consider subjective constraints in their plain form, namely, as in $\bot \leftarrow \mathbf{K}p$. Notice also that, according to the definition of splitting provided in Cabalar *et al.* (2021), subjective constraints can be placed at either of two adjacent levels. For convenience concerning definitions that will be introduced later, we impose, again without loss of generality, that both subjective rules satisfying condition (2) of the definition of epistemic splitting set (Definition 4.1) and subjective constraints are put in $T_U(\Pi)$.

We now proceed to introduce the key definitions on which our approach is based.

Definition 6.1
Let be given a semantics $\mathcal{S}$, a program Π, and an epistemic splitting $\langle B_U(\Pi), T_U(\Pi)\rangle$ of Π, according to the definition of epistemic splitting set. Let $F_U(\Pi)$ denote the set of all subjective literals $\mathbf{K}L$ occurring in $T_U(\Pi)$ (even in negative form $\mathit{not}\,\mathbf{K}L$) and referring to $B_U(\Pi)$ (in the sense that the atom involved in $\mathbf{K}L$ occurs in $B_U(\Pi)$ but not in $T_U(\Pi)$), together with their negations $\mathit{not}\,\mathbf{K}L$.

Intuitively, subjective literals in $F_U(\Pi)$ constitute the "interface" between the top and bottom parts. Notice that $\mathit{Atoms}(F_U(\Pi)) \subseteq U$.

Definition 6.2
Let Π be a program and let $F_U(\Pi) = \{\mathbf{K}L_1, \ldots, \mathbf{K}L_z, \mathit{not}\,\mathbf{K}L_1, \ldots, \mathit{not}\,\mathbf{K}L_z\}$. Let, moreover, $f_U(\Pi) = \{kl_1, \ldots, kl_z, nkl_1, \ldots, nkl_z\}$ be a set of fresh atoms. The *detached version* $T'_U(\Pi)$ of $T_U(\Pi)$ is the program consisting of:

- the rules obtained from rules in $T_U(\Pi)$ by substituting each occurrence of the subjective literal $\mathbf{K}L_i \in F_U(\Pi)$ or $\mathit{not}\,\mathbf{K}L_i \in F_U(\Pi)$ by the corresponding fresh atom $kl_i \in f_U(\Pi)$ or $nkl_i \in f_U(\Pi)$, for each $i \in \{1, \ldots, z\}$ (where kl_i and nkl_i are in turn called the *detached form* of $\mathbf{K}L_i$ and $\mathit{not}\,\mathbf{K}L_i$, resp.); and
- the facts $kl_i \mid nkl_i$, for each $i \in \{1, \ldots, z\}$.

We introduced $T'_U(\Pi)$ in order to model the connection between $T_U(\Pi)$ and $B_U(\Pi)$ w.r.t. the top-down perspective. Thus, we need to define the notion of *world views of the detached version* $T'_U(\Pi)$ *of a program* under the assumption that the fresh atoms kl_i and nkl_i represent the epistemic literals connecting the top and bottom parts of the program. As seen below, these world views not necessarily coincide with the world views of $T'_U(\Pi)$ if considered as an epistemic program by itself.

Recall that a disjunction between an epistemic literal $\mathbf{K}L$ and its negation $\mathit{not}\,\mathbf{K}L$ determines, as discussed in Costantini (2019), two world views, one entailing $\mathbf{K}L$ and the other one entailing $\mathit{not}\,\mathbf{K}L$. With respect to the subjective literals in $F_U(\Pi)$, in defining the detached version $T'_U(\Pi)$ of a program $T_U(\Pi)$ we encoded the potential existence of such alternative world views by means of the disjunctions $kl_i \mid nkl_i$, for $i \in \{1, \ldots, z\}$.

In computing the world views of the detached version $T'_U(\Pi)$, we start by considering $T'_U(\Pi)$ as a regular epistemic program (forgetting for the moment that the fresh atoms kl_i and nkl_i stand for epistemic literals) thus obtaining the corresponding collection of world views $\mathcal{W}$. Note in fact that $T'_U(\Pi)$ does not contain subjective literals referring to the bottom $B_U(\Pi)$, but it may contain "local" epistemic literals that may determine the existence of several world views (or just one if there are no such local epistemic literals). The answer sets in each $W \in \mathcal{W}$ might however contain some of the atoms kl_is and nkl_is. In this case, each $W \in \mathcal{W}$ has to be split into two world views, say W_1 and W_2, the former composed of the answer sets in W that contain kl_1, and the latter composed by those answer sets of W that contain nkl_1. This step must be repeated by considering the pair kl_2/nkl_2 in order to split both W_1 and W_2, and so on, for each $i \in \{1, \ldots, z\}$. (Observe that the order of splits does not matter.) We consider the resulting collection of sets of atoms as the world views of the detached version $T'_U(\Pi)$. An example of this process will be given at the end of Section 6.2. In summary:

Definition 6.3 (*World views of* $T'_U(\Pi)$, *or Interface World Views*)
Let $W^1, \ldots, W^n$ be the world views of $T'_U(\Pi)$ according to a given semantics $\mathcal{S}$. The Interface World Views of $T'_U(\Pi)$ are obtained as follows: for every W^j, $j \leq n$, $W^j = \{S^j_1, \ldots, S^j_v\}$ for some $v \geq 0$, and for every disjunction $kl_i \mid nkl_i$, $i \in \{1, \ldots, z\}$ occurring in $T'_U(\Pi)$, split W^j into W^j_1 and W^j_2, the former composed of the sets $S^j_h \in W^j$ such that $kl_i \in S^j_h$, the latter composed of the of the sets $S^j_f \in W^j$ such that $nkl_i \in S^j_f$, $f \in \{1, \ldots, v\}$. Repeat the splitting over the resulting world views, and iterate the process until splitting is no longer possible, that is, no resulting world view contains both kl_r and nkl_r, for some $r \in \{1, \ldots, z\}$.

The denomination "Interface World Views" indicates that they have been obtained in the perspective of a merge with world views of the bottom, as seen below. For the sake of conciseness though by some abuse of notation, we will call Interface World Views simply 'world views'.

Proposition 6.1
There exists a bijection between world views of $T_U(\Pi)$ and world views of $T'_U(\Pi)$.

Proof
Given a world view (Interface World View, to be precise) W'_j of the epistemic program $T'_U(\Pi)$, a world view W_j for $T_U(\Pi)$ is equal to $W_j = \{X \setminus f_U(\Pi) \mid X \in W'_j\}$. In fact, the procedure for obtaining Interface World Views takes into account the fact that each epistemic literal represented by an atom in $f_U(\Pi)$ can be potentially either true or false. Vice versa, W'_j is obtained from W_j by adding to it some subset of $f_U(\Pi)$. □

For each of such world views W_j of $T_U(\Pi)$, Definition 6.4 below identifies the set of subjective literals that are relevant in extending W_j to a world view of the entire Π. These are those that in the detached version of $T_U(\Pi)$ have been assumed to be true to obtain W_j as a world view.

Definition 6.4 (*Epistemic Top-down Requisite Set*)
Let $\langle B_U(\Pi), T_U(\Pi)\rangle$ be an epistemic splitting for a program Π, W'_j be a world view of $T'_U(\Pi)$, and let $W_j = \{X \setminus f_U(\Pi) \mid X \in W'_j\}$.

The set $ES_{T_U(\Pi)}(W_j) = \{\mathbf{K}L_h \mid W'_j \models kl_h\} \cup \{not\, \mathbf{K}L_h \mid W'_j \not\models kl_h\}$ is the *(epistemic top-down) requisite set* for W_j (w.r.t. $\langle B_U(\Pi), T_U(\Pi)\rangle$).

Now we partition the *requisite set*, identifying two relevant subsets (technical reasons for doing so will be seen below).

Definition 6.5
Given $f_U(\Pi) = \{kl_1, \ldots, kl_z, nkl_1, \ldots, nkl_z\}$ and the above definition of requisite set $ES_{T_U(\Pi)}(W_j)$, w.r.t. an epistemic splitting $\langle B_U(\Pi), T_U(\Pi)\rangle$, let set S include those kl_i/nkl_i that occur in some constraints in $T'_U(\Pi)$.
We split the requisite set $ES_{T_U(\Pi)}(W_j)$ as the union of the following two (disjoint) sets:

- the *epistemic top-down constraint set*:

$$EC_{T_U(\Pi)}(W_j) = (\{\mathbf{K}L_i \mid kl_i \in S\} \cup \{not\, \mathbf{K}L_i \mid nkl_i \in S\}) \cap ES_{T_U(\Pi)}(W_j)$$

- the *requirement set*:

$$RQ_{T_U(\Pi)}(W_j) = \big(\{\mathbf{K}L_i \mid kl_i \in f_U(\Pi)\backslash S\} \cup \{not\, \mathbf{K}L_i \mid nkl_i \in f_U(\Pi)\backslash S\}\big) \\ \cap ES_{T_U(\Pi)}(W_j).$$

There is an important reason for distinguishing these two subsets. Namely, the literals in $EC_{T_U(\Pi)}(W_j)$, if not entailed in some world view of the bottom part of the program, lead to a constraint violation and cause the nonexistence of world views of Π extending W_j. Thus, $EC_{T_U(\Pi)}(W_j)$ expresses prerequisites on which epistemic literals must be entailed in a world view of $B_U(\Pi)$, so that such world view can be merged with W_j in order to obtain a world view of Π. Instead, literals in $RQ_{T_U(\Pi)}(W_j)$, can be usefully exploited, as seen below, to drive the selection of which world view of the bottom can be combined with a given world view of the top.

For all the three sets (requisite set, constraint set, and requirement set) one can possibly list only the epistemic literals of $F_U(\Pi)$ required to be true, all the others implicitly required to be false.

Given a world view W of $T_U(\Pi)$ and considering literals belonging to $EC_{T_U(\Pi)}(W)$ which occur in the bodies of rules in $B_U(\Pi)$, we introduce a simplification that can be performed and will turn out to be useful later on.

Definition 6.6 (*Top-down Influence*)
Given a world view W of $T_U(\Pi)$, and its corresponding top-down constraint set $EC_{T_U(\Pi)}(W)$, the W-tailored version $B_U^W(\Pi)$ of $B_U(\Pi)$ is obtained by substituting in $B_U(\Pi)$ all literals $\mathbf{K}L \in EC_{T_U(\Pi)}(W)$ by L.

The intuition behind the above definition is that, if $\mathbf{K}A$ is in $EC_{T_U(\Pi)}(W)$, then A must necessarily belong to every answer set of a world view of the bottom that can be possibly merged with W in order to obtain a world view of the overall program Π. Hence, it is indifferent that in the body of rules of $B_U(\Pi)$ it occurs A rather than $\mathbf{K}A$, if $\mathbf{K}A \in EC_{T_U(\Pi)}(W)$. Substituting $\mathbf{K}A$ with A can, however, be useful, as discovered during the development of the G11 (Gelfond 2011) and K15 semantics (Kahl *et al.* 2015), to "break" unwanted positive cycles among subjective literals, that might lead to *unfounded* world views (cf. Cabalar *et al.* (2021, Definition 15)).

In our approach, the notion of top-down influence provides, as seen by examples in the next section, an alternative perspective on how a world view of the bottom is obtained, and, in a sense, a re-interpretation of the notion of foundedness (to be formally elaborated in future work).

In the top-down approach that we are going to propose, the world views of a given program Π are obtained as a combination of world views of the top and world views of the bottom, like in the bottom-up approach. In the basic version of the TDESP, presented in Section 6.2, there is only a change of perspective and a simple condition to drive the combination via the WBT operation (cf. Definition 4.3).

In the definition of the more general TDESP, presented in Section 6.3, one can notice two relevant changes: (i) the notion of top-down influence is exploited in the definition of candidate world views; (ii) a subset of a world view of the bottom (i.e., some of the answer sets occurring therein) may be cut out, so as to enable the merging via WBT with a "compatible" world view of the top.

Preliminarily:

Definition 6.7
Given a set E of epistemic literals and a set of sets of atoms W, we say that W *fulfills* E iff $\forall \mathbf{K}L \in E, W \models L$ and $\forall\, not\, \mathbf{K}L \in E, W \not\models L$.

6.2 Top-down epistemic splitting property – Basic (TDESPB)

Definition 6.8 (*Candidate World View - Basic Version*)
Given an epistemic splitting $\langle B_U(\Pi), T_U(\Pi)\rangle$ for a program Π, let W_T be a world view of $T_U(\Pi)$ and let W_B be a world view of $B_U(\Pi)$ that fulfills $EC_{T_U(\Pi)}(W_T)$ such that W_B also fulfills $RQ_{T_U(\Pi)}(W_T)$ (overall, W_B fulfills the requisite set $ES_{T_U(\Pi)}(W_T)$). Then,

$$W = W_B \sqcup W_T = \{I_b \cup I_t | I_b \in W_B \wedge I_t \in W_T\}$$

is a *candidate world view* for Π (obtained from W_T and W_B).

It is possible that no world views of the bottom comply with the conditions posed by world views of the top: in such case, Π has no candidate world views.

We can now state a property that, if satisfied by a semantics, allows world views to be computed top-down:

Definition 6.9 (*Top-down Epistemic Splitting Property - Basic Version (TDESPB)*)
A semantics $\mathcal{S}$ satisfies *basic top-down epistemic splitting* if any candidate world view of Π according to Definition 6.8 is indeed a world view of Π under $\mathcal{S}$.

Below we show that TDESPB is equivalent to the ESP by Cabalar *et al.* (2021), in the sense that both definitions are satisfied by the same semantic approaches and thus characterize the same world views.

Theorem 6.1 (*Equivalence ESP - TDESPB*)
A semantics $\mathcal{S}$ satisfies TDESPB if and only if $\mathcal{S}$ satisfies the ESP as defined in Definition 4.1.

Proof
If part. Assume that a given semantics $\mathcal{S}$ satisfies TDESPB. To show that $\mathcal{S}$ satisfies ESP as well, it suffices to observe that the couple $\langle W_B, W_T\rangle$ according to Definition 6.8 is a $\mathcal{S}$-solution as required by the definition of ESP. In fact, W_B is an $\mathcal{S}$-world view of the bottom $B_U(\Pi)$. It remains to be seen that W_T is an $\mathcal{S}$-world view of $E_U(\Pi, W_B)$, that is, that, after simplifying $T_U(\Pi)$ w.r.t. the subjective literals entailed by W_B, one would have W_T among the world views. By Definition 6.8, W_B fulfills the requisite set $ES_{T_U(\Pi)}(W_T)$, leading $W_B \sqcup W_T$ to be a world view of the overall program. This means, according to Definition 6.4, that W_B entails all the subjective literals of the form $\mathbf{K}A$ and *not* $\mathbf{K}A$, that, in the detached version of $T_U(\Pi)$ (Definition 6.2) have been assumed to be true (in their detached form) in order to obtain W_T as a world view (according to $\mathcal{S}$). Thus, if one would simplify $T_U(\Pi)$ into $E_U(\Pi, W_B)$ by considering exactly those subjective literals as true and all the others as false, one would trivially obtain W_T as the world view of $E_U(\Pi, W_B)$.

Only if part. Assume that a given semantics $\mathcal{S}$ satisfies ESP. This means that there exists a $\mathcal{S}$-solution $\langle W_B, W_T\rangle$ that, via WBT, gives rise to the world views of the program.

To be an $\mathcal{S}$-solution, W_B must be a world view of the bottom, and W_T a world view of $E_U(\Pi, W_B)$, that is, of the top simplified w.r.t. W_B. To find the correspondence with TDESPB, we have to ascertain that $\langle W_B, W_T\rangle$ gives rise to candidate world views in the sense of Definition 6.8. To do so, we put into $ES_{T_U(\Pi)}(W_T)$ the subjective literals, among those entailed by W_B, that are employed to perform such simplification, so as to exactly fulfill the conditions posed in Definition 6.4. $\square$

The equivalence stated by Theorem 6.1 implies that the world views of a program can be determined by composing the world views of the various layers into which the program can be split, by proceeding either bottom-up, according to the original definition, or top-down, according to our new definition. We will now illustrate the approach, and its similarities and differences w.r.t. ASP, by means of an example.

Consider the following sample ASP program.

$$\begin{array}{l} f \leftarrow a. \\ e \leftarrow c. \\ \bot \leftarrow \mathit{not}\, p. \\ a \leftarrow p. \\ a \leftarrow q. \\ p \leftarrow \mathit{not}\, q. \\ q \leftarrow \mathit{not}\, p. \\ c. \end{array}$$

A possible split according to Lifschitz & Turner can be:

$$\begin{array}{l} \mathit{Top\ part} \\ f \leftarrow a. \\ e \leftarrow c. \\ \bot \leftarrow \mathit{not}\, p. \\ \\ \mathit{Bottom\ part} \\ a \leftarrow p. \\ a \leftarrow q. \\ p \leftarrow \mathit{not}\, q. \\ q \leftarrow \mathit{not}\, p. \\ c. \end{array}$$

Notice that the unique answer set of this program is $S = \{c, p, a, e, f\}$. The answer sets of the bottom part are: $S1 = \{c, p, a\}$, $S2 = \{c, q, a\}$. The answer set of the top part, assuming p true (otherwise the constraint is violated), is $S3? = \{e?, f?\}$, the question mark meaning that any of the two atoms can be true, according to the selected answer set of the bottom. In this simple case, we have to choose $S2$, which makes p true, and, by imagining adding atoms in $S2$ as new facts in the top part, we get both e and f, thus obtaining the answer set S. Let us now consider the top part as a standalone program:

$$\begin{array}{l} f \leftarrow a. \\ e \leftarrow c. \\ \bot \leftarrow \mathit{not}\, p. \end{array}$$

This program in itself is inconsistent, but knowing that it is intended as the top part of a wider program, we can set the requirements for any bottom part, in the form of what we can call *Epistemic top-down Constraint set* $EC = \{p\}$, that is, p must be true in an answer set of the bottom, for the top to be consistent. If we enrich the top as follows:

$$\begin{array}{l} f \leftarrow a. \\ e \leftarrow c. \\ \bot \leftarrow not\, p. \\ \\ p \mid nop. \\ a \mid noa. \\ c \mid noc. \end{array}$$

We can compute all possible answer sets for the top part, by simulating possible values for atoms coming from the (still unknown) bottom. Each such simulation, for example, assuming a true and c false, gives rise to a *Requisite Set RQ*. Then, given a specific bottom program that one intends to add to the top, each answer set M of the bottom that fulfills EC can be combined with all the answer sets of the top that are compatible, in the sense that M entails all literals in the corresponding RQ.

Let us now consider an ELP with a very similar structure.

$$\begin{array}{l} \textit{Top part} \\ f \leftarrow \mathbf{K}a. \\ e \leftarrow \mathbf{K}c. \\ \bot \leftarrow not\, \mathbf{K}p \\ \\ \textit{Bottom part} \\ a \leftarrow p. \\ a \leftarrow q. \\ p \leftarrow not\, \mathbf{K}q. \\ q \leftarrow not\, \mathbf{K}p. \\ c. \end{array}$$

Let us first proceed bottom-up, as dictated by the ESP definition. The world views of the bottom, according to any existing semantics, are: $W1 = \{\{c, p, a\}\}$, $W2 = \{\{c, q, a\}\}$.

Below is the top part simplified w.r.t. $W1$, with a unique resulting world view $\{\{e, f\}\}$.

$$\begin{array}{l} f. \\ e. \end{array}$$

The top part simplified w.r.t. $W2$ is reported below, with no world views as the constraint is violated:

$$\begin{array}{l} \textit{Top part w.r.t. W2} \\ f. \\ e. \\ \bot \leftarrow \top \end{array}$$

Therefore, the unique world view of the overall program is, by the WBT operation which reduces here to a simple union, $W = \{\{c, p, a, e, f\}\}$.

Let us now apply the notions related to the top-down splitting property TDESPB that we presented above. We have the following detached version of the top part:

$$\begin{array}{l} f \leftarrow ka. \\ e \leftarrow kc. \\ \bot \leftarrow nkp \\ ka \mid nka. \\ kc \mid nkc. \\ kp \mid nkp. \end{array}$$

Seen as an epistemic program by itself, this program has a unique world view (indeed, it is a standard ASP program), which is

$$\{\{kp, nka, nkc\},\ \{kp, ka, nkc, f\},\ \{kp, nka, kc, e\},\ \{kp, ka, kc, e, f\}\}.$$

By splitting this set of sets three times (w.r.t. the pairs ka/nka, kc/nkc, and kp/nkp) as described in Section 6.1, Definition 6.3, we obtain the world views of the detached version: $\{\{kp, nka, nkc\}\}$, $\{\{kp, nka, kc, e\}\}$, $\{\{kp, ka, nkc, f\}\}$, and $\{\{kp, ka, kc, e, f\}\}$.

From them, one determines the *epistemic top-down constraint set* which is, clearly, $EC = \{\mathbf{K}p\}$, stating that the unique constraint must be satisfied. Any **compatible** world view of a bottom should satisfy one of the RQ^i's, $i \leq 4$, that is, the requirement sets, listed below (cf. Definitions 6.4 and 6.5). To each RQ^i it corresponds a world view of the top (indicated on the right) to be united to those world views of the bottom that satisfy RQ^i (if any).

$$\begin{array}{lll} RQ^1 = \emptyset, & \textit{determines} & W_t^1 = \{\emptyset\} \\ RQ^2 = \{\mathbf{K}c\}, & \textit{determines} & W_t^2 = \{\{e\}\} \\ RQ^3 = \{\mathbf{K}a\}, & \textit{determines} & W_t^3 = \{\{f\}\} \\ RQ^4 = \{\mathbf{K}c, \mathbf{K}a\}, & \textit{determines} & W_t^4 = \{\{e, f\}\} \end{array}$$

Given the world views of the bottom, that is, $W1 = \{\{c, p, a\}\}$, $W2 = \{\{c, q, a\}\}$, we can see that $W2$ does not fulfill EC and so must be discarded, while $W1$ fulfills EC and also RQ^4, thus leading, by the WBT operation which reduces here to a simple union, the to (unique) world view of the overall program $W = \{\{c, p, a, e, f\}\}$.

It is immediate to verify that the result obtained via the bottom-up and the top-down approach is indeed the same.

6.3 Top-down epistemic splitting property (TDESP)

In this subsection, we will extend previous definitions to a more general form, so as to be able to characterize in a top-down fashion the world views obtained according to many semantic approaches presented in the literature, other than G91 and FAAEL, such as, for example, those proposed by Shen and Eiter (2016), Kahl and Leclerc (2018), and Su (2019); in fact, they do not enjoy the basic property TDESPB illustrated above. We introduce a different way of computing candidate world views, where, in the absence of a world view of the bottom that fulfills the set EC relative to the top, one can select a subset of such a world view. This, as we will demonstrate in our running example, is analogous to what is customarily done in ASP.

Definition 6.10 (*Candidate World View*)
Given an epistemic splitting $\langle B_U(\Pi), T_U(\Pi)\rangle$ for a program Π, let W_T be a world view of $T_U(\Pi)$ and let W_B be a subset of a world view of $B_U^{W_T}(\Pi)$ that fulfills $EC_{T_U(\Pi)}(W_T)$ (where, if EC is empty, W_B is the entire world view of the bottom) such that W_B fulfills $RQ_{T_U(\Pi)}(W_T)$. Then,

$$W = W_B \sqcup W_T = \{I_b \cup I_t | I_b \in W_B \wedge I_t \in W_T\}$$

is a *candidate world view* for Π (obtained from W_T and W_B).

Note that, candidate world views are now computed after applying top-down influence. It is possible that no subset of any world view of the bottom complies with the conditions posed by world views of the top. In such case, Π has no candidate world views.

We can now state another property concerning top-down epistemic splitting that a semantics might obey:

Definition 6.11 (*Top-down Epistemic Splitting Property (TDESP)*)
A semantics $\mathcal{S}$ satisfies *top-down epistemic splitting* if any candidate world view of Π according to Definition 6.10 is indeed a world view of Π under $\mathcal{S}$.

We can state the relationship among TDESP and ESP/TDESPB (that, as seen, are equivalent).

Theorem 6.2
Given a semantics $\mathcal{S}$ satisfies both foundedness and ESP/TDESPB, then $\mathcal{S}$ satisfies TBDESP.

Proof
If $\mathcal{S}$ satisfies TDESPB, this means that for every world view of given program Π obtained via the WBT operation, and thus composed of a world view W_T of the top and a world view W_B of the bottom, every $\mathbf{K}L \in EC_{T_U(\Pi)}(W)$ is entailed by W_B and, if $\mathcal{S}$ satisfies foundedness, this equates to say that L is entailed by the bottom part of the program. Thus, the application of Top-down Influence is irrelevant. We then notice that, according to Definitions 6.10 and 6.3 a candidate world view for TDESP can be obtained from an entire world view of the bottom, as done for TDESPB. This concludes the proof, showing that for this class of semantics TDESP and TDESPB are indeed equivalent. □

The above theorem is immediately applicable to the FAAEL semantics. For semantics which do not enjoy foundedness things are different, as seen in the examples below. We will now, in fact, experiment with our methodology on some relevant examples proposed in recent literature. Consider program Π_1, taken from (Shen and Eiter 2020):

$$\begin{array}{ll} p \mid q & (r1) \\ \bot \leftarrow \mathit{not}\, \mathbf{K}p & (C) \end{array}$$

Here, $B_U(\Pi_1)$ consists of rule (r1), and $T_U(\Pi_1)$ consists of constraint (C). So, $T'_U(\Pi_1)$ is (where kp and nkp are fresh atoms):

$$\begin{array}{ll} kp \mid nkp & (r1) \\ \bot \leftarrow nkp & \end{array}$$

whose unique world view is $\{\{kp\}\}$. After canceling kp, we obtain $W_T = \{\emptyset\}$ for $T_U(\Pi_1)$, with $ES_{T_U(\Pi_1)}(W_T) = EC_{T_U(\Pi_1)}(W_T) = \{\mathbf{K}p\}$ and $RQ_{T_U(\Pi_1)}(W_T) = \emptyset$. Regardless of the epistemic semantics $\mathcal{S}$, as no subjective literals occur therein, the unique world view of $B_U(\Pi_1)$ is $\hat{W} = \{\{p\},\{q\}\}$. Since $W_B = \{\{p\}\}$ is the only subset of $\hat{W}$ fulfilling $EC_{T_U(\Pi_1)}(W_T)$ (cf. Definition 6.10), then it is the one selected. It is also a world view for Π_1, as the unique world view of the top part is empty. This world view violates subjective constraint monotonicity, still, it is the one delivered by the semantics proposed in Shen and Eiter (2016) and, as noticed in Shen and Eiter (2020), by those proposed in Kahl and Leclerc (2018) and Su (2019).

In our opinion the world view $\{\{p\}\}$ captures the "intended meaning" of the program Π_1, where the top layer "asks" the bottom layer to support, if possible, $\mathbf{K}p$ (in order not to make the overall program inconsistent). Let us, in fact, introduce a simple variation, by adding a fact, say c, to the program, where c also occurs in the constraint, obtaining:

$$\begin{array}{ll} p \mid q & (r1) \\ c. & (f1) \\ \bot \leftarrow c, not\,\mathbf{K}p & (C) \end{array}$$

We would obtain, in this case, the world view $\{\{c,p\}\}$. Let us now reinterpret this program within the work of the COST Action DigForASP, that is, in the realm of investigations. A rephrasing could be the following:

$$\begin{array}{ll} at\_crime\_scene \mid not\_at\_crime\_scene & (r1) \\ reliable\_witness\_recognizes. & (f1) \\ \bot \leftarrow reliable\_witness\_recognizes, not\,\mathbf{K}\,at\_crime\_scene & (C) \end{array}$$

The meaning underlying the schematic formulation is that it is uncertain whether a suspect was or not at the crime scene. However, if a reliable witness recognized the suspect, then investigators can be certain that the suspect was indeed at the crime scene. The constraint could in fact be rephrased (although this is not legal syntax) into:

$$\mathbf{K}\,at\_crime\_scene \leftarrow reliable\_witness\_recognizes.$$

The use of the $\mathbf{K}$ is crucial here because one wants to distinguish between facts collected by the investigators and reliable conclusions derived by these facts. Thus, the world view $\{\{reliable\_witness\_recognizes, at\_crime\_scene\}\}$ makes perfect sense here.

In addition, one might consider the very similar ASP program:

$$\begin{array}{ll} p \mid q & (r1) \\ c. & (f1) \\ \bot \leftarrow c, not\,p & (C) \end{array}$$

with unique answer set $\{c,p\}$. The "bottom" program fragment consisting of (r1)+(f1) would also have answer set $\{c,q\}$, which is discarded since it would lead to violating the constraint. We may consider this program as an ELP, with unique world view $\{\{c,p\}\}$ obtained from a subset of the world view $\{\{c,p\},\{c,q\}\}$ of the bottom (union the empty world view of the top), exactly as specified in Definition 6.10.

Consider now the following program Π_2.

$$\begin{array}{ll} p \mid q & (r1) \\ \bot \leftarrow not\,\mathbf{K}p & (C) \\ p \leftarrow \mathbf{K}q & (r2) \\ q \leftarrow \mathbf{K}p & (r3) \end{array}$$

Here, $B_U(\Pi_2)$ consists of rules (r1-r3), and $T_U(\Pi_2)$ consists of constraint (C). So, $T'_U(\Pi_2)$ is (where kp and nkp are fresh atoms):

$$\begin{array}{l} kp \mid nkp \\ \bot \leftarrow nkp \end{array}$$

whose unique world view is $\{\{kp\}\}$. After canceling kp, we obtain world view $W_T = \{\emptyset\}$ for $T_U(\Pi_2)$ where $ES_{T_U(\Pi_2)}(W_T) = EC_{T_U(\Pi_2)}(W_T) = \{\mathbf{K}p\}$ and set RQ is empty. Regardless of the semantics $\mathcal{S}$, the potential world views of $B_U(\Pi_2)$ are $W_1 = \{\{p\}\}$, $W_2 = \{\{q\}\}$, $W_3 = \{\{p\},\{q\}\}$, $W_4 = \{\{p,q\}\}$. Actually, W_4 is the only one fulfilling $ES_{T_U(\Pi_2)}(W_T)$; W_1 has the problem that, having p and fulfilling $\mathbf{K}p$, (r3) might be applied thus getting q. Note that W_4 is in fact the world view returned by semantics proposed, for instance, in Kahl *et al.* (2015) and Shen and Eiter (2016). It is easy to see that W_4 violates foundedness. However, in our approach q is not derived via the positive cycle (extended to subjective literals), but from the $\mathbf{K}p$ "forced" by the upper layer via top-down influence, which substitutes $\mathbf{K}p$ with p in rule (r3) of $B_U(\Pi_2)$. This in a sense guarantees a form of foundedness, though not the formal one introduced in Cabalar *et al.* (2021, Definition 15). Since the unique world view for the top is empty, then the unique world view of the overall program is, indeed, according to our method, $W = W_4 = \{\{p,q\}\}$.

Let us now consider Π_3 to be the seminal example introduced in Gelfond and Przymusinska (1991), which is discussed in virtually every paper on ELP. Π_3 is epistemically stratified (see Definition 4.4 and Cabalar *et al.* (2021, Definition 6)). This formulation (variations have appeared over time) is from Cabalar *et al.* (2021).

$$\begin{array}{ll}
\mathit{eligible}(X) \leftarrow \mathit{high}(X) & (r1) \\
\mathit{eligible}(X) \leftarrow \mathit{minority}(X), \mathit{fair}(X) & (r2) \\
\mathit{noeligible}(X) \leftarrow \mathit{not\,fair}(X), \mathit{not\,high}(X) & (r3) \\
\mathit{fair}(\mathit{mike}) \mid \mathit{high}(\mathit{mike}) & (f1) \\
\mathit{interview}(X) \leftarrow \mathit{not}\,\mathbf{K}\,\mathit{eligible}(X), \mathit{not}\,\mathbf{K}\,\mathit{noeligible}(X) & (r4) \\
\mathit{appointment}(X) \leftarrow \mathbf{K}\,\mathit{interview}(X) & (r5)
\end{array}$$

Since in this version of the program we have only *mike* as an individual, we may obtain the following ground abbreviated version:

$$\begin{array}{ll}
e \leftarrow h & (r1) \\
e \leftarrow m, f & (r2) \\
ne \leftarrow \mathit{not}\,f, \mathit{not}\,h & (r3) \\
f \mid h & (f1) \\
in \leftarrow \mathit{not}\,\mathbf{K}e,\ \mathit{not}\,\mathbf{K}ne & (r4) \\
a \leftarrow \mathbf{K}in & (r5)
\end{array}$$

Here, we consider (r5) as the top $T_U(\Pi_3)$, and (r1-r4) plus (f1) as the bottom, which can be however in turn divided into the top $T1_U(\Pi_3)$ including (r4), and the bottom $B_U(\Pi_3)$, made of (r1-r3) and (f1). So, $T'_U(\Pi_3)$ is (with fresh atoms kin, $nkin$):

$$\begin{array}{ll}
a \leftarrow kin & (r5') \\
kin \mid nkin &
\end{array}$$

with two answer sets: $\{a, kin\}, \{nkin\}$. As explained in Section 6.1, $kin \mid nkin$ stands for a disjunction between the epistemic literal $\mathbf{K}in$ and its negation $not\,\mathbf{K}in$. This determines the existence of two world views, each entailing only one of these atoms, that is, epistemic literals, where atom a can, however, be derived only from the former. Thus, we have $W_{11} = \{\{a\}\}$ with $ES_{T_U(\Pi_3)}(W_{11}) = \{\mathbf{K}in\}$, and $W_{12} = \{\emptyset\}$ with $ES_{T_U(\Pi_3)}(W_{12}) = \{not\,\mathbf{K}in\}$. $EC_{T1_U(\Pi_3)}$ is empty for all world views, as no constraint is present in Π_3. Then, $T1'_U(\Pi_3)$ is (with $ke, nke, kne, nkne$ fresh atoms):

$$\begin{array}{ll} in \leftarrow nke, nkne & (r4') \\ ke \mid nke & \\ kne \mid nkne. & \end{array}$$

By the same reasoning as above, since there are two disjunctions among fresh atoms representing epistemic literals, four world views can be found. After canceling the fresh atoms, in fact we have $W_{21} = \{\{in\}\}$, with $ES_{T1'_U(\Pi_3)}(W_{21}) = \{not\,\mathbf{K}e, not\,\mathbf{K}ne\}$, and three empty world views $W_{22} = W_{23} = W_{24} = \{\emptyset\}$, with requisite sets $\{\mathbf{K}e, \mathbf{K}ne\}$, $\{\mathbf{K}ne, not\,\mathbf{K}e\}$, and $\{not\,\mathbf{K}ne, \mathbf{K}e\}$, respectively. Clearly, also $EC_{T1'_U(\Pi_3)}$ is empty.

Finally, $B_U(\Pi_3)$, which is made of the rules (r1-r3) and (f1), has the world view $W_3 = \{\{h, e\}, \{f\}\}$. Since the requirement set relative to world view W_{21} for the immediately upper level is satisfied in both answer sets of W_3, we can obtain an intermediate world view $W_{213} = \{\{h, e, in\}, \{f, in\}\}$ for the part of the program including (r1-r4). Considering also the top, it is easily seen that W_{213} is compliant with the requirement set of $W_{11} = \{a\}$. So, we can obtain for the overall program the unique candidate world view $W = \{\{h, e, in, a\}, \{f, in, a\}\}$, which is indeed a world view. Notice that, in fact, the world views that are part of the union, corresponding to the various subprograms, would be the same under all known semantics for ELPs.

Assume now that, instead of $f \mid h$, the program contains the bare fact h. Then, the world view of the bottom becomes $W_3 = \{\{h, e\}\}$. This world view implies $\mathbf{K}e$, so it can be combined with a world view $\{\emptyset\}$ of the middle layer, and since it also implies $not\,\mathbf{K}in$, the further combination is with world view $W_{12} = \{\emptyset\}$ of the top. So, $W_3 = \{\{h, e\}\}$ is in this case the unique world view of the overall program.

7 Main result

It is at this point interesting to try to assess formally which semantics (if any) satisfy the TDESP.

We examine now the case of the semantics introduced in Kahl *et al.* (2015), that we call for short K15. The reason for choosing K15 is that in Cabalar *et al.* (2021) it is noticed that K15 slightly generalizes the semantics proposed in Gelfond (2011) (called G11 for short) and can be seen as a basis for the semantics proposed in Shen and Eiter (2016) (called S16 for short). In particular, S16 (which considers instead of $\mathbf{K}$ the operator **not** A which means $not\,\mathbf{K}A$) treats K15 world views as candidate solutions, to be pruned in a second step, where some unwanted world views are removed by maximizing what is not known. Thus, should K15 satisfy the TDESP, S16 would do as well, and so would G11, the latter however only for the (wide) class of programs where its world views coincide with those of K15.

Definition 7.1 (K15-world views)
The K15-reduct of Π with respect to a nonempty set of interpretations W is obtained by:

(i) replacing by $\bot$ every subjective literal $L \in Body_{subj}(r)$ such that $W \not\models L$, and
(ii) replacing all other occurrences of subjective literals of the form $\mathbf{K}L$ by L.

A nonempty set of interpretations W is a K15-world view of Π iff W is the set of all stable models of the K15-reduct of Π with respect to W.

We are able to prove the following:

Theorem 7.1 (K15 TDESP)
The K15 semantics satisfies the TDESP. That is, given an ELP Π, and set of sets W, where each set is composed of atoms occurring in Π, W is a K15 world view for Π if and only if it is a candidate world view for Π according to Definition 6.10.

Proof
Assume an Epistemic Splitting of given program Π into two layers, top $T_U(\Pi)$ and bottom $B_U(\Pi)$ (where the reasoning below can, however, be iterated over a subdivision into an arbitrary number of levels). Notice that, given a K15 world view W, since each atom A that occurs in the sets composing W is derived in the part of the program including rules with head A, then W can be divided into two parts, W_T and W_B which are world views of $T_U(\Pi)$ and $B_U(\Pi)$, resp., each one composed of stable models of the K15-reduct of that part of the program.

If part. Given a K15 world view W, let Sl^T be the subjective literals occurring in $T_U(\Pi)$ which are entailed by the bottom, that is, either of the form $\mathbf{K}A$, for which $W_B \models A$, or of the form $not\,\mathbf{K}A$, for which $W_B \not\models A$. Let such a set of literals form the set $ES_{T_U(\Pi)}(W_T)$. (As mentioned, the subset of Sl^T that consists of literals involved in constraints in $T_U(\Pi)$ will form set $EC_{T_U(\Pi)}(W_T)$, and the remaining ones will form set $RQ_{T_U(\Pi)}(W_T)$.) Therefore, we can conclude that W, which is a K15 world view, is indeed a candidate world view according to Definition 6.10.

Only if part. Consider a candidate world view W w.r.t. the K15 semantics, obtained by combining a subset W_B of a K15 world view of $B_U(\Pi)$ with a K15 world view W_T of $T_U(\Pi)$ after top-down influence. According to Definition 6.10, the combination is possible only if for each epistemic literal $\mathbf{K}A \in ES_{T_U(\Pi)}(W_T)$, $W_B \models A$, and for each epistemic literal $not\,\mathbf{K}A \in ES_{T_U(\Pi)}(W_T)$, $W_B \not\models A$. If any such literal belongs to $EC_{T_U(\Pi)}(W_T)$, if this is not the case then there would be a constraint violation in $T_U(\Pi)$, so there would be no world views for $T_U(\Pi)$ and for the overall program Π. Considering a subjective literal in $RQ_{T_U(\Pi)}(W_T)$, if it were not the case that W_B entails such literal, then by definition of K15 it would have been substituted by $\bot$, so W_T would have been a different set. The top-down influence step can be disregarded since it performs in advance on elements of $ES_{T_U(\Pi)}(W_T)$, that are required to be entailed by W_B anyway, the same transformation performed by K15, step (ii). Then, a candidate world view W obtained according to Definition 6.10 is indeed a K15 world view. □

In Cabalar *et al.* (2021, Th. 2) it is proved that, for any semantics obeying epistemic splitting, an epistemically stratified program has a unique world view. Actually, it can be seen that epistemically stratified programs admit one (and the same) world view under any existing semantics, and in particular under those considered here: as it is well-known (see, e.g., Gelfond (1994), Shen and Eiter (2016), and Costantini (2019)), multiple world views arise in consequence of negative cycles involving epistemic literals, clearly not present in such programs. So, the unique world view of an epistemically stratified program is, in particular, a K15 world view. Thus, we have the following.

Corollary 7.1
Epistemically Stratified Programs satisfy both the Top-down and Bottom-up Epistemic Splitting Properties under any semantics.

8 Conclusions

In this paper, we have provided a way of exploiting the splitting of ELPs in a top-down fashion, adequate for those situations where the top part of a program is well established as it represents a problem formulation, where the bottom part (representing a problem instance) may vary and is in general not known in advance.

We defined formal conditions for the combination of world views of the top with world views of the bottom into world views of the overall program. In addition, potential world views of the top can be pre-computed, thus simplifying the combination with the world views of each problem instance. We provide a version that is the top-down declination of the well-established approach by Cabalar et al., and a more general version that is applicable to a wider range of semantic approaches.

A question that may arise concerns the efficiency of the top-down approach, even though in many cases it will be an almost inevitable choice. If the subjective literals "connecting" adjacent layers are in small numbers (as it seems reasonable), then efficiency might not be a concern.

It remains to be seen in more depth for which kinds of applications the different approaches to splitting (top-down and bottom-up) might be most profitably exploited. As an example, we can go back to the suggestion proposed in Kahl *et al.* (2015) to encode the problem of finding a conformant plan as the task of obtaining a world view. As emphasized in Cabalar *et al.* (2021), splitting allows one to separate the planner definition (the "top") from the generation of alternative plans (an intermediate layer, we might say "the top of the bottom") from, in turn, the domain description (the "bottom"). The top-down perspective would allow one to analyze the top part independently from the other layers, so as to identify in advance the prerequisites it poses to them.

An investigation of which other semantics might satisfy the TDESP is also a subject of future work.

References

Cabalar, P., Fandinno, J. and del Cerro, L. F. 2021. Splitting epistemic logic programs. *Theory and Practice of Logic Programming 21,* 3, 296–316.

Cabalar, P., Fandinno, J. and Fariñas del Cerro, L. 2019. Splitting epistemic logic programs. In *Proceedings of LPNMR'19*, M. Balduccini, Y. Lierler, and S. Woltran, Eds. Lecture Notes in Computer Science, vol. 11481. Springer, 120–133.

Costantini, S. 2002. Meta-reasoning: A survey. In *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski, Part II*, A. C. Kakas and F. Sadri, Eds. Lecture Notes in Computer Science, vol. 2408. Springer, 253–288.

Costantini, S. 2006. On the existence of stable models of non-stratified logic programs. *Theory and Practice of Logic Programming 6,* 1–2, 169–212.

Costantini, S. 2019. About epistemic negation and world views in epistemic logic programs. *Theory and Practice of Logic Programming 19,* 5–6, 790–807.

Costantini, S. and Formisano, A. 2015. Negation as a resource: a novel view on answer set semantics. *Fundamenta Informaticae 140,* 3–4, 279–305.

Costantini, S. and Formisano, A. 2021. Adding metalogic features to knowledge representation languages. *Fundamenta Informaticae 181,* 1, 71–98.

Costantini, S. and Formisano, A. 2022. Epistemic logic programs: a novel perspective and some extensions. In *Proceedings of the International Conference on Logic Programming 2022 Workshops co-located with the 38th International Conference on Logic Programming (ICLP) 2022, Haifa, Israel, July 31st – August 1st, 2022*, J. Arias, R. Calegari, L. Dickens, W. Faber, J. Fandinno, G. Gupta, M. Hecher, D. Inclezan, E. LeBlanc, M. Morak, E. Salazar, and J. Zangari, Eds. CEUR Workshop Proceedings, vol. 3193. CEUR-WS.org.

Dix, J. 1995. A classification theory of semantics of normal logic programs I-II. *Fundamenta Informaticae 22,* 3, 227–255 and 257–288.

Fandinno, J., Faber, W. and Gelfond, M. 2022. Thirty years of epistemic specifications. *Theory and Practice of Logic Programming 22,* 6, 1043–1083.

Fariñas del Cerro, L., Herzig, A. and Su, E. I. 2015. Epistemic equilibrium logic. In *Proceedings of IJCAI 2015*, Q. Yang and M. Wooldridge, Eds. AAAI Press, 2964–2970.

Gelfond, M. 1994. Logic programming and reasoning with incomplete information. *Annals of Mathematics and Artificial Intelligence 12,* 1–2, 89–116.

Gelfond, M. 2011. New semantics for epistemic specifications. In *Proceedings of LPNMR'11*, J. P. Delgrande and W. Faber, Eds. LNCS, vol. 6645. Springer, 260–265.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proc. of the 5th International Conference and Sympoxium on Logic Programming*, R. Kowalski and K. Bowen, Eds. MIT Press, 1070–1080.

Gelfond, M. and Przymusinska, H. 1991. Definitions in epistemic specifications. In *Proceedings of the 1st International Workshop on Logic Programming and Non-monotonic Reasoning*, A. Nerode, V. W. Marek, and V. S. Subrahmanian, Eds. The MIT Press, 245–259.

Kahl, P., Watson, R., Balai, E., Gelfond, M. and Zhang, Y. 2015. The language of epistemic specifications (refined) including a prototype solver. *Journal of Logic and Computation 30,* 4, 953–989.

Kahl, P. T. and Leclerc, A. P. 2018. Epistemic logic programs with world view constraints. In *Technical Communications of the ICLP 2018*, A. D. Palù, P. Tarau, N. Saeedloei, and P. Fodor, Eds. OASIcs, vol. 64. Schloss Dagstuhl, 1:1–1:17.

Leone, N., Rullo, P. and Scarcello, F. 1997. Disjunctive stable models: Unfounded sets, fixpoint semantics, and computation. *Information & Computation 135,* 2, 69–112.

Lifschitz, V. 2010. Thirteen definitions of a stable model. In *Fields of Logic and Computation*, A. Blass, N. Dershowitz, and W. Reisig, Eds. LNCS, vol. 6300. Springer, 488–503.

Lifschitz, V. and Turner, H. 1994. Splitting a logic program. In *Proceedings of ICLP'94*. MIT Press, 23–37.

Lloyd, J. W. 1987. *Foundations of Logic Programming*. Springer-Verlag.

Shen, Y. and Eiter, T. 2016. Evaluating epistemic negation in answer set programming. *Artificial Intelligence 237*, 115–135.

Shen, Y. and Eiter, T. 2020. Constraint monotonicity, epistemic splitting and foundedness are too strong in answer set programming. *CoRR abs/2010.00191*, 1–6.

Su, E. I. 2019. Epistemic answer set programming. In *Proceedings of JELIA'19*, F. Calimeri, N. Leone, and M. Manna, Eds. LNCS, vol. 11468. Springer, 608–626.

Su, E. I. 2021. Refining the semantics of epistemic specifications. In *Proceedings 37th International Conference on Logic Programming (Technical Communications), ICLP Technical Communications 2021, Porto (virtual event), 20-27th September 2021*, A. Formisano, Y. A. Liu, B. Bogaerts, A. Brik, V. Dahl, C. Dodaro, P. Fodor, G. L. Pozzato, J. Vennekens, and N. Zhou, Eds. Electronic Proceedings in Theoretical Computer Science, vol. 345. EPTCS, 113–126.

Truszczynski, M. 2011. Revisiting epistemic specifications. In *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, M. Balduccini and T. C. Son, Eds. LNCS, vol. 6565. Springer, 315–333.