Weighted Answer Set Programs

Francisco, Bruno, Salvador, Dietmar

November 8, 2024

Abstract

Drawing inspiration from HMMs; State of the art of PLP and Probabilistic ASP; Leveraging current Prob ASP systems;

Using a logic program to model and reason over a real world scenario is often hard because of uncertainty underlying the problem being worked on. Classic logic programs represent knowledge in precise and complete terms, which turns out to be problematic when the scenario is characterized by stochastic or observability factors. For example, medical exams illustrate both problems: a system with unreachable parts *e.g.* some parts of a living organism can't be directly observed; a sensor that adds noise to real values *e.g.* limits and imperfections of instrumentation.

In this work we aim to explore how answer set programs plus weight augmented facts can lead to useful characterizations for this class of problems; We assume that knowledge about a *system* includes both a theoretical *model*|[1] and empirical *data* such that:

- The model is an answer set program whose stable models are the the system states.
- The data is a set of observations and each observation is a set of literals from the model.
- The weights in the augmented facts are propagated to the stable models and, in general, to observations such as above.

[1]|Use 'answer set' instead of 'stable model'?

In this setting the empirical distribution from the data can be used in two different tasks:

1. Estimate parameteres required in the propagation process above.
2. Evaluate the model *wrt.* the observations.

To frame this setting we assume that the atoms in the model are associated to sensors of the system's states. More specifically, if a an atom, a state can activate the sensor a or $\neg a$ (strong negation) whereas no activation is represented by $\sim a$ and $\sim\neg a$ (default negation). This redundancy is required to model hidden parts of a system as well as faulty sensors. |[2] For example, in

$$\{a, \neg a, \sim b, \sim\neg b, \sim c, \neg c\}$$

[2]|Look carefully to the relation between missing values in observations and default negation.

both a and $\neg a$ are activated (suggesting a fault in the relevant sensors), b is not observed (hidden?), and $\sim c, \neg c$ reports the (consistent) activation of $\neg c$ and no activation of c. If we omit the default negations and use $\overline{x}$ to denote $\neg x$ this observation can be shortened to the equivalent form $a\overline{a}\overline{c}$.

Example 1 (Coins, Cards, and Dices) *Consider a scenario where a coin is tossed and, if it lands in* heads*,* a*, then a dice is thrown,* b*, or a card is drawn,* c*. If the coin lands in* tails*,* $\neg a$*, no more action is performed. A model of this is, for example, the program*

$$P_{ccd} = \begin{cases} a \vee \neg a, \\ \quad b \vee c \leftarrow a \end{cases} \tag{1}$$

that has atoms $\{a, b, c\}$*, literals* $\{a, \neg a, b, \neg b, c, \neg c\}$ *and, using the shortned form, stable models*

$$ab, ac, \overline{a}. \tag{2}$$

Possible observations include:

- ac*, short of* $\{a, \sim\neg a, \sim b, \sim\neg b, c, \sim\neg c\}$*: a stable model, therefore a system state, observed* heads *and* card.
- a*: not a SM but contained in the two SMs* ab *and* ac*; observed* heads *but we have no information about the* cards *or the* dice.
- $\overline{a}\overline{b}$*: not a SM but contains the SM* $\overline{a}$*; coin observed as* not heads *(*tails*?); also we recorded no card drawn but have no information about the dice.*
- b, c*: not related with any SM.*
- $a\overline{a}$*: an inconsistent observation.*

We use sets of literals, instead of atoms, to represent observations because they make possible the distintion between a *explicit negative observation* $\neg\alpha$ (*e.g.*"I see tails.") and *not observed* $\sim\alpha$ (*e.g.*"The coin in hidden."). This corresponds to assuming a "boolean sensor" for each α and $\neg\alpha$.

So, the program 1 defines the "sensors"

$$a, \neg a, b, \neg b, c, \neg c$$