Summary of Semantics for Hybrid Probabilistic Logic Programs with Function Symbols

Damiano Azzolini
University of Ferrara
Dipartimento di Ingegneria

Fabrizio Riguzzi
University of Ferrara
Dipartimento di Matematica e Informatica

Evelina Lamma
University of Ferrara
Dipartimento di Ingegneria
{damiano.azzolini,fabrizio.riguzzi,evelina.lamma}@unife.it

The aim of Probabilistic Logic Programming is to extend the expressiveness of Logic Programming by adding the possibility to manage uncertainty over the data with the introduction of probabilistic facts representing discrete random variables. Some years ago, to manage also continuous random variables, hybrid programs were proposed, but the semantics was imprecise. In this paper, we present a formal definition of a semantics that assigns a probability value to all queries for programs with a two-valued Well-Founded model. This paper is a summary of [1].

1 Contribution

Probabilistic Logic Programs (PLPs) usually support only discrete probabilistic facts (i.e., facts that can be true or false with a certain probability) representing Bernoulli random variables. However, many models of real world tasks also require managing continuous values. Hybrid Probabilistic Logic Programs [1, 6] extend traditional PLPs with the introduction of continuous random variables (rvs). The resulting probability space, defined as the product of the probability spaces of discrete and continuous rvs, is usually infinite. Furthermore, even in programs with only discrete probabilistic facts, the number of variables may be infinite (for example, when there is at least one function symbol and one constant). When function symbols, continuous random variables, and constraints are combined, the definition of a well-defined semantics is crucial to assign a probability value to queries.

We consider the probability space induced by the program as the product of the spaces of discrete and continuous random variables. Then, for every sample taken from the sample space for the whole program, a ground normal program is defined, called a *world*. If each world has a two-valued Well-Founded model, we call the probabilistic program *sound*. We prove that every sound program is well-defined (each query has an assigned probability). If the program fails to have all worlds with a two-valued Well-Founded model, other semantics must be used [2].

During the years, several semantics for programs with both discrete and continuous random variables have been proposed, none of them managing function symbols. Hybrid ProbLog [3] was one of the first languages that allowed the definition of a finite set of continuous probabilistic facts. Similarly, continuous distributions are supported by Distributional Clauses (DC) [4] but negation in the body of rules is not allowed: this limitation is no more present in HAL-ProbLog [8], one of its extension. The same happens with Extended PRISM [5]. Finally, the semantics of Probabilistic Constraint Logic Programs [7] constrain the sample space of the whole program to be countable, a situation that may be violated when there is at least one constant and a function symbol, as explained before.

A. Formisano, Y.A Liu, et al. (Eds.): International Conference on Logic Programming (Technical Communications) 2021 (ICLP 2021)
EPTCS ??, 2021, pp. i–ii, doi:10.4204/EPTCS.??.??

For example, consider a game where a player needs to pick a card from a deck and spin a wheel. The card can be either blue or red and it is reinserted in the deck after every round. The game continues until the player does not pick a red card and the axis of the wheel is between 0 and π. In this simple example, there are both discrete (color of card) and continuous (position of the axis of the wheel) random variables. We may be interested in computing the probability that the player draws at least one time, or that he/she never draws a red card: to compute the probability of these two queries, we need to consider an infinite number of trials. The following program models the aforementioned scenario:

```
1/2 :: blue(X).
angle(_,X) : uniform_dens(X,0,6.28).
pick(0,blue) :- blue(0), angle(0,V), V > 3.14.
pick(0,red) :- \+ blue(0), angle(0,V), V > 3.14.
pick(s(X),blue):- \+ pick(X,red), angle(X,V), V > 3.14,
    blue(s(X)), angle(s(X),V), V > 3.14.
pick(s(X),red):- \+ pick(X,red), angle(X,V), V > 3.14,
    \+ blue(s(X)), angle(s(X),V), V > 3.14.

at_least_once_red :- pick(_,red).
never_red :- \+ at_least_once_red.
```

Both queries `at_least_once_red` and `never_red` have and infinite number of groundings, since the anonymous variable `_` is unified iteratively with `0`, `s(0)`, `s(s(0))`, ... To compute its probability we need to consider, for example, mutually disjoint covering sets of worlds. See [1] for further details.

References

[1] Damiano Azzolini, Fabrizio Riguzzi & Evelina Lamma (2021): *A Semantics for Hybrid Probabilistic Logic Programs with Function Symbols*. *Artificial Intelligence* 294, p. 103452, doi:10.1016/j.artint.2021.103452.

[2] Fabio Gagliardi Cozman & Denis Deratani Mauá (2020): *The joy of Probabilistic Answer Set Programming: Semantics, complexity, expressivity, inference*. *International Journal of Approximate Reasoning* 125, pp. 218–239, doi:10.1016/j.ijar.2020.07.004.

[3] Bernd Gutmann, Manfred Jaeger & Luc De Raedt (2011): *Extending ProbLog with Continuous Distributions*. In Paolo Frasconi & Francesca A. Lisi, editors: *20th International Conference on Inductive Logic Programming (ILP 2010)*, *LNCS* 6489, Springer, pp. 76–91, doi:10.1007/978-3-642-21295-6_12.

[4] Bernd Gutmann, Ingo Thon, Angelika Kimmig, Maurice Bruynooghe & Luc De Raedt (2011): *The magic of logical inference in probabilistic programming*. *Theory and Practice of Logic Programming* 11(4-5), pp. 663–680, doi:10.1017/S1471068411000238.

[5] Muhammad Asiful Islam, CR Ramakrishnan & IV Ramakrishnan (2012): *Inference in probabilistic logic programs with continuous random variables*. *Theory and Practice of Logic Programming* 12, pp. 505–523, doi:10.1017/S1471068412000154.

[6] S Michels (2016): *Hybrid Probabilistic Logics: Theoretical Aspects, Algorithms and Experiments*. Ph.D. thesis, Radboud University Nijmegen.

[7] Steffen Michels, Arjen Hommersom, Peter J. F. Lucas & Marina Velikova (2015): *A new probabilistic constraint logic programming language based on a generalised distribution semantics*. *Artificial Intelligence* 228, pp. 1–44, doi:10.1016/j.artint.2015.06.008.

[8] Pedro Zuidberg Dos Martires, Anton Dries & Luc De Raedt (2018): *Knowledge Compilation with Continuous Random Variables and its Application in Hybrid Probabilistic Logic Programming*. *CoRR* abs/1807.00614.