PLOW: Probabilstic Logic Over the Well-Founded semantics

Benjamin Grosof and Theresa Swift
Kyndi, Inc.
San Mateo, California, USA
http://benjamingrosof.com
http://cs.stonybrook.edu/~tswift

Motivations: A central challenge for AI is how to combine most effectively two core areas: knowledge representation & reasoning (KRR) and machine learning (ML). ML can provide capabilities for acquiring knowledge from data at relatively low cost in human labor. Adding KRR to ML can provide the ability to combine heterogeneous forms and/or multiple sources of knowledge (both human crafted and machine learned), and then use this combination to recommend or make decisions. When the form of KRR is highly explainable, it can provide significantly more trustability to AI systems, helping to meet the rapidly rising demands of users. KRR also offers further advantages in interpreting context-specific ML results. Significantly, KRR can be essential for complying with organizational policies, ethics, and pertinent legal regulations; and for applying human social and experiential knowledge.

Problem Addressed: KRR systems of various kinds have advanced rapidly in the past 5–10 years. In particular, a class of recent logic programming systems, called *Rulelog* systems, semantically extend database logic and provide high expressiveness together with strong explainability (Grosof 2013; Andersen et al. 2013). These systems support logical functions and higher-order syntax, defeasibility (a.k.a. exceptions, argumentation, and defaults), constraint-based reasoning, and reactivity. Further, they provide a variety of semantically meaningful forms of bounded rationality to enforce query termination. Unlike logic programming systems based on answer set programming (ASP), these Rulelog systems are based on the three-valued well-founded semantics, which offers much better scalability than ASP. Notable Rulelog systems include ErgoAI, Flora-2, and XSB.[1] However, until recently Rulelog systems such as these have lacked the sort of quantitative uncertainty reasoning that is needed to reason productively and efficiently using results from a wide variety of ML approaches.

Copyright held by the author(s). In A. Martin, K. Hinkelmann, A. Gerber, D. Lenat, F. van Harmelen, P. Clark (Eds.), Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019). Stanford University, Palo Alto, California, USA, March 25-27, 2019.

[1]Available through http://coherentknowledge.com (ErgoAI); http://flora.sourceforge.net (Flora-2); http://xsb.sourceforge.net (XSB). ErgoAI was formerly known as Ergo Suite.

Approach and its Features: The PLOW system seeks to address this lack by offering several flavors of uncertainty, each with a solid semantic and proof-theoretic basis. These encompass both Bayesian and non-Bayesian kinds of probability. Specifically, PLOW includes the following three flavors, all within a single system:

- *Bayesian Probability-based Semantics* are based on the distribution semantics (Sato 1995) for probabilistic logic programs, using the syntax of Logic Programs with Annotated Disjunctions (LPADs) (Vennekens, Verbaeten, and Bruynooghe 2004). The probabilistic semantics supported include:
  - *The full distribution semantics*, which extends logic programs with full Bayesian reasoning. (In fact logic programs under the full distribution semantics are strictly more powerful than Bayesian nets.) However, like probabilistic reasoning in general, probabilistic logic programming can have a high computational cost (cf. (Riguzzi and Swift 2018) for an overview).
  - *The restricted distribution semantics*. For applications that don't need the generality of the full distribution semantics, PLOW also offers an implementation of the restricted distribution semantics, which makes the assumption of the independence of probabilistic choices made within a derivation, along with an assumption of the exclusivity of the probabilistic choices of multiple derivations of a given subgoal (Sato, Kameya, and Zhou 2005).
- *T-norm based Semantics*. A variety of fuzzy logics are based on different T-norms (cf. (Klement, Mesiar, and Pap 2004) for an overview). T-norms originally were formulated when studying probabilistic metric spaces. Logics based on T-norms have proven to be well-suited for reasoning with vague concepts (such as whether a given person is tall), and with reasoning over knowledge that is based on similarity or relevancy measures. Reasoning about such statements often requires the use of a quantitative strength that is not probabilistic in the Bayesian sense.
- *Lattice-based Semantics*. Lattice-based semantics have formed the basis of various multi-valued logics as well as possibilistic logics (Dubois, Lang, and Prade 1994). Such

logics have been used to capture both contradictory evidence and qualitative measures of belief (e.g., *unlikely*, *possible*, *likely* and so on).

Each of these three approaches can be brought to bear when combining ML with reasoning. The case for probabilistic semantics is perhaps the most obvious. Results of some statistical learning techniques, such as linear/quadratic discrimination analysis, have formal probabilistic properties. Results of other types of (machine) learning, such as neural networks with a softmax output, are often also interpreted probabilistically. Each of these results can be propagated through (deductive) reasoning under a probabilistic semantics. However, there are situations in which probabilistic reasoning is not suitable, either because of its high computational complexity, or when ML results are quantitative but not clearly probabilistic (such as similarity scores obtained by word embeddings e.g., (Mikolov et al. 2013)). For such cases, reasoning with T-norms may be more appropriate. And finally, there may be applications where ML results need to be transformed into qualitative measures to be used or interpreted: in such casees lattice-based semantics can be helpful.

PLOW has been implemented as a package of the XSB system[2], and supports a general logic programming framework for KRR that includes:

- Fully recursive rules that include logical functions along with both default and explicit negation (also known as negation-as-failure and strong negation, respectively).
- The ability to specify the strength of quantitative rules via a variety of functions of the strength of their bodies, including strength boosting, strength decay, sigmoid, and rectified linear functions.
- Evaluation of these rules according to the three-valued Paraconsistent Well-Founded Semantics (Damásio and Pereira 1998), a kind of (unprioritized) defeasibility.

Discussion: An important limitation of PLOW is that it does not support general reasoning-by-cases (i.e., inferring disjunctions), since it is based on the well-founded semantics. By contrast, some other approaches to probabilistic logical KRR do support general reasoning-by-cases, via extending first-order classical logic (FOL) or using ASP.

On the other hand, PLOW has several advantages compared to other approaches to probabilistic logical KRR. First, PLOW's support for the well–founded semantics, strong negation and other features provides a basis for the full support of much, perhaps even all, of Rulelog's approach to KRR. Second, PLOW offers multiple flavors of probability under one roof; it has more flexible expressiveness in that regard than any previous logic programming system of which we are aware. This is useful not only for application system builders but also to facilitate study of these probability flavors at the level of KRR algorithm design and theory. Third, PLOW offers several kinds of relatively *scalable* probabilistic reasoning: not only restricted distribution semantics, but also T-norm based and lattice based. Scalability is a major challenge in probabilistic logical KRR; many, if not most, of other approaches that provide medium or high expressiveness in the non-probabilistic logical aspect suffer from intractable computational complexity. Finally, in terms of pragmatic consideration of interoperability: Since PLOW is implemented using XSB, it inherits XSB's database, web, Java, C, Python and other interfaces.

Acknowledgements The work described here was done in part while previously at Coherent Knowledge, Inc.

References

Andersen, C.; Benyo, B.; Calejo, M.; Dean, M.; Fodor, P.; Grosof, B. N.; Kifer, M.; Liang, S.; and Swift, T. 2013. Advanced knowledge base debugging for Rulelog. In *Joint Proceedings of the 7th International Rule Challenge, the Special Track on Human Language Technology and the 3rd RuleML Doctoral Consortium, Seattle, USA, July 11 -13, 2013.*

Damásio, C. V., and Pereira, L. M. 1998. A survey on paraconsistent semantics for extended logic programas. In Gabbay, D., and Smets, P., eds., *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 2. Kluwer Academic Publishers. 241–320.

Dubois, D.; Lang, J.; and Prade, H. 1994. Possibilistic logic. In Gabbay, D. M.; Hogger, C. J.; and Robinson, J. A., eds., *Handbook of logic in artificial intelligence and logic programming,vol. 3*. Oxford University Press. 439–514.

Grosof, B. 2013. Rapid Text-based Authoring of Defeasible Higher-Order Logic Formulas, via Textual Logic and Rulelog (Summary of Invited Talk). In *Proc. RuleML-2013, the 7th Intl. Web Rule Symposium.*

Klement, E.-P.; Mesiar, R.; and Pap, E. 2004. Triangular norms. position paper i: basic analytical and algebraic properties. *Fuzzy Sets and Systems* 143:5–26.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Neural Information Processing Systems*, 3111–3119.

Riguzzi, F., and Swift, T. 2018. A survey of probabilistic logic programming. In *Declarative Logic Programming: Theory, Systems, and Applications*, volume 20 of *ACM Books*, 185–233.

Sato, T.; Kameya, Y.; and Zhou, N.-F. 2005. Generative modeling with failure in PRISM. In *International Joint Conference on Artificial Intelligence*, 847–852.

Sato, T. 1995. A statistical learning method for logic programs with distribution semantics. In *International Conference on Logic Programming*, 715–729. MIT Press.

Vennekens, J.; Verbaeten, S.; and Bruynooghe, M. 2004. Logic programs with annotated disjunctions. In *International Conference on Logic Programming*, 195–209.

[2]A development version of PLOW is available at `https://github.com/theresasturn/plow`.