PLOW: Probabilistic Logic Over the Well-founded Semantics

<u>Benjamin Grosof</u> and Theresa Swift

Kyndi, Inc.

http://benjamingrosof.com
http://www3.cs.stonybrook.edu/~tswift
https://kyndi.com

Poster and Demo presentation (20 minutes) at the AAAI 2019 Spring Symposium on Combining Machine Learning and Knowledge Engineering (AAAI-MAKE)

March 26, 2019

Stanford, California, USA

Acknowledgements and Thanks: This presentation includes, in part, work done by the authors while at Coherent Knowledge and at Kyndi, Inc. It includes, in part, material originally co-authored by: Michael Kifer, Janine Bloomfield, and Paul Fodor.

Overview of Talk

- Goal: Provide an approach for uncertainty in KRR, to be used in combining logical KRR with ML, that has a better balance of expressiveness and computational scalability.

- Contribution: theory and implementation, as extended form of logic programming
  - Fuzzy (t-norms), in addition to Bayesian

- Simple examples, and brief demo

- How influenced by others' work at the Symposium:
  - Applications / use cases
  - Design patterns for adding KRR to ML

Motivation and Background

- Probabilistic logic KRR is a fundamental bridge between ML and KE
- Declarative logic programs (LP) is the central KR of IT
  - DBs: Relational DBs (SQL). Knowledge graphs, a.k.a. graph DBs (SPARQL).
  - Ontologies: OWL-RL, RDF-S.
  - Rules: Prolog; RIF; Production rules, Event-Condition-Action rules.
- LP's non-classical logic – invented by/for computer science not math
  - Humble spirit: avoid reasoning-by-cases/disjunction; avoid proof-by-contradiction; stay grounded
  - Well-founded semantics: 3 truth values, benefits for scalability & robustness
- Rulelog – extended LP with high expressiveness + scalability
  - Defeasibility, higher-order syntax, object-oriented (frame) syntax, quantified classical-like formulas, restraint bounded rationality, provenance; poly-time!
  - But lacks kind of quantitative uncertainty needed to reason productively and efficiently using results from a wide variety of ML approaches
- Distribution semantics – extended LP with Bayesian-flavor probability
  - But lacks good scalability, due to reintroducing head disjunction

Why Need Scalability of the Uncertain KRR for Combining ML and KE

- Inner loop of ML
- KB dev edit-test cycle
- Large KGs/KBs

Presenters' Background

- Kyndi: AI startup combining ML+KRR+NLP; venture-backed
  - Specialized search & question-answering, via advanced knowledge graphs
  - Customers in national intelligence, pharma, other domains

- Benjamin Grosof – Chief Scientist at Kyndi. Previously:
  - Founding CTO/CEO of Coherent Knowledge, AI startup on Rulelog KRR engine
  - Led advanced research portion of Allen Institute for AI's predecessor (Vulcan)
  - MIT Sloan IT professor, DARPA PI, IBM Research projects lead, Accenture exec
  - Co-invented many advances in LP/Rulelog

- Theresa Swift – scientist at Kyndi
  - Also researcher/engineer at US Customs & Border Patrol
  - Lead implementer of XSB
  - Co-founder of Coherent Knowledge
  - Co-invented many advances in LP/Rulelog

Probabilistic LP – Expressive Extension of LP

- Numerical truth values for atoms (and rules) range on real interval [0..1]
- *head* formula can be: \or of disjoint atoms/literals whose weights add to 1
    - friendly(?x)~0.8 \or unfriendly(?x)~0.2 :- student(?x).
- Two major flavors of numerical uncertainty
    1. Bayesian flavor cf. "distribution semantics" [Sato]
        - Superset of Bayesian Networks, expressively
        - General case is computationally intractable, even for function-free
    2. Generalized "triangular norms" (t-norms), a.k.a. fuzzy flavor.
        - Parametrized by choice of the t-norm function F.
        - pr(A \and B) = F(pr(A),pr(B)). I.e., "truth-functional" – key to scalability.
        - E.g., F = *min*. Co-norm for \or: e.g., *max* . Same F is applied to every A,B.
        - Polynomial time for function-free
        - Generalization: F=MinMax, a function on intervals, where the interval is cautious in regard to the potential correlation of A and B.

Bayesian PLP Reasoning: Example

heads(Coin)~0.5 \or tails(Coin)~0.5   :- toss(Coin) \and fair(Coin).

heads(Coin)~0.6 \or tails(Coin)~0.4   :- toss(Coin) \and biased(Coin).

fair(Coin)~0.9 \or biased(Coin)~0.1.

toss(coin).

- Conclude: heads(Coin)~0.51 .

T-Norms

- Full Bayesian reasoning is powerful but (computationally) expensive.

- Epistemically, Bayesian probabilities may not be a good way to represent similarity and relevancy distances. We say, more generally: "<u>measures</u>".

- Hence, T-Norms (Triangular Norms, a generalization of Fuzzy Logic)
  - Godel (i.e., "Min" for conjunction): the measure of A *op* B expresses perfect correlation (+1) of A and B
  - Lukasiewicz: the measure of A *op* B expresses negative correlation (-1) of A and B
  - Product: the measure of A *op* B expresses independence (correlation 0) of A and B
  - "MinMax" (new!): generalizes the measure to an interval [Lukasiewicz, Godel] expressing an interval of truth, cautious in regard to how much correlation of A and B.

PLOW System for Probabilistic LP

- The first to implement the generalized t-norm flavor
- Bayesian flavor (a.k.a. distribution semantics), too
- Lattice flavor qualitative uncertainty, too
- Supports \neg (strong negation)
- Utilizes *undefined* truth value, as do normal LP and Rulelog
- A way to combine deductive reasoning with ML facts and rules
    - E.g., in knowledge graphs
- Implementation extends XSB
    - The PLPs are transformed into normal LP
    - BDDs (Binary Decision Diagrams) are used to collate information from different deduction paths
- In-progress: Aim to integrate tightly with as many Rulelog features as possible. Starting with defeasibility and restraint. Already reusing some of Rulelog's algorithms, theory, implementation!

PLOW Uses

- Similarity relations – e.g., two documents may be more or less related
- Vague properties – e.g., a certain person may be more or less ”tall”
- Relevancy relations – e.g., a document may be more or less relevant to a query
- Confidence measures – e.g., a document may come from a more or less trusted source
- Lower complexity probability measures – such as “evidential” probabilities

Strong Negation in PLOW

- Notation:
  naf(q) denotes default negation of q. (“not believe” q)
  neg(q) denotes strong (a.k.a. explicit) negation of q. (“believe opposite” of q)
- Simple example:

p~0.4.

p~0.5.

p :- undefined.

neg(p)~0.2.

In this case, p~M is

t if M <= 0.5

u if 0.5 < M < 0.8

f if 0.8 < M <= 1

One can view there as being 3 zones (or bands) of measures having the 3 truth values: a zone for (or where) t, a zone for u, a zone for f.

PLOW Paraconsistent/Defeasibility Semantics

- Semantics is an extension of Well-Founded Semantics with Explicit Negation to include quantitative values
  - Uses the coherence principle: strong (i.e., explicit) negation implies default negation.
- Paraconsistent values are mapped to u. This is a kind of defeasible conflict handling.
- Thus, given the assertions:
  - p~0.6
  - neg(p)~0.6
- Then conclude that:

p~M is:

- t for M < 0.4
- u for 0.4 <= M <= 0.6
- f for 0.6 < M <= 1

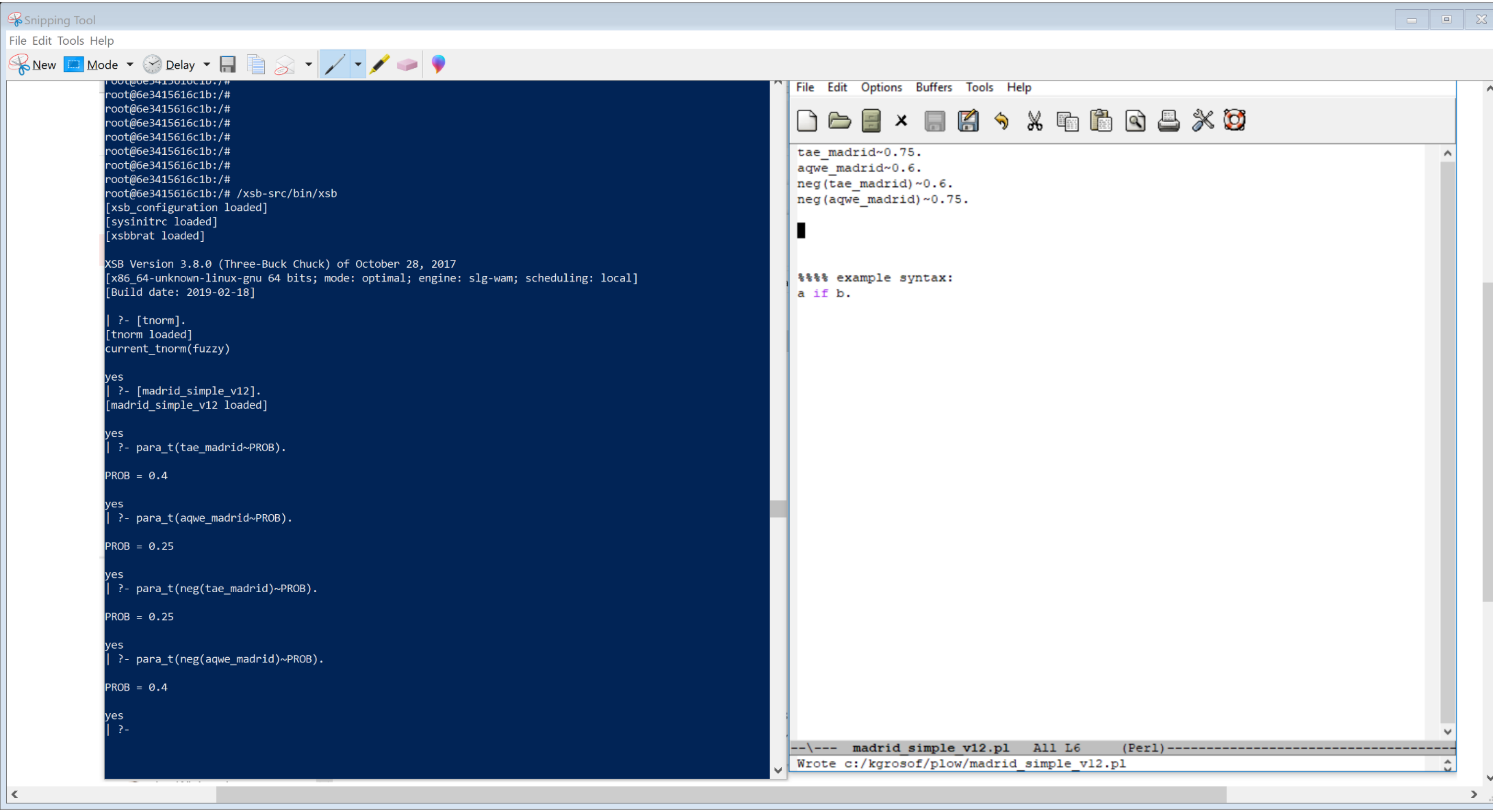
neg(p)~M is:

t for M < 0.4
u for 0.4 <= M <= 0.5
f for 0.6 < M <= 1

BRIEF DEMO GOES HERE

The next few slides are screenshots

Overall Demo – XSB/PLOW command line; and KB editor (in Emacs)



Example KB (zoomed)

```
tae_madrid~0.75.
aqwe_madrid~0.6.
neg(tae_madrid)~0.6.
neg(aqwe_madrid)~0.75.
```

Start XSB, PLOW; load example KB

```
root@6e3415616c1b:/# /xsb-src/bin/xsb
[xsb_configuration loaded]
[sysinitrc loaded]
[xsbbrat loaded]

XSB Version 3.8.0 (Three-Buck Chuck) of October 28, 2017
[x86_64-unknown-linux-gnu 64 bits; mode: optimal; engine: slg-wam; scheduling: local]
[Build date: 2019-02-18]

| ?- [tnorm].
[tnorm loaded]
current_tnorm(fuzzy)

yes
| ?- [madrid_simple_v12].
[madrid_simple_v12 loaded]

yes
```

Query the example KB, in PLOW

```
| ?- para_t(aqwe_madrid~PROB).

PROB = 0.25

yes
| ?- para_t(neg(tae_madrid)~PROB).

PROB = 0.25

yes
| ?- para_t(neg(aqwe_madrid)~PROB).

PROB = 0.4

yes
| ?-
```

Conclusions: Contributions

- Multiple flavors of uncertainty for logic programs, all under one roof
  - Bayesian, i.e., distribution semantics. Both general and restricted.
  - Fuzzy, i.e., t-norms. Highly scalable.
  - Lattice, i.e., qualitative
  - Implementation as extension (package) of XSB, inheriting many good features
- Interval t-norm: MinMax
  - With interpretation of bounds on correlation
- Leverages undefined truth value, and supports unstratified NAF
- Supports strong negation (\neg), with basic defeasibility
- Supports well: logical functions, in combination with uncertainty
  - Well-defined: Finite number of finite models, unlike other probabilistic LP approaches. Ensured by restraint + call subsumption (features of XSB).
  - Positioned well to combine with the higher-order syntax (HiLog) feature of Rulelog, useful to represent advanced defeasibility, causality, natural language

Current and Future Directions

- KRR end:
  - Relate MinMax t-norm to approximation of distribution semantics
  - More on defeasibility including prioritization, argumentation meta-rules
  - Explore and roadmap integration with more/rest of Rulelog features
  - Address idempotence issues for product and Lukasiewicz t-norms. Ideas:
    - Path independence cf. IND. Compilation cf. BDDs/circuits. Human-authored control.
  - Converge syntax with LPAD cf. PITA

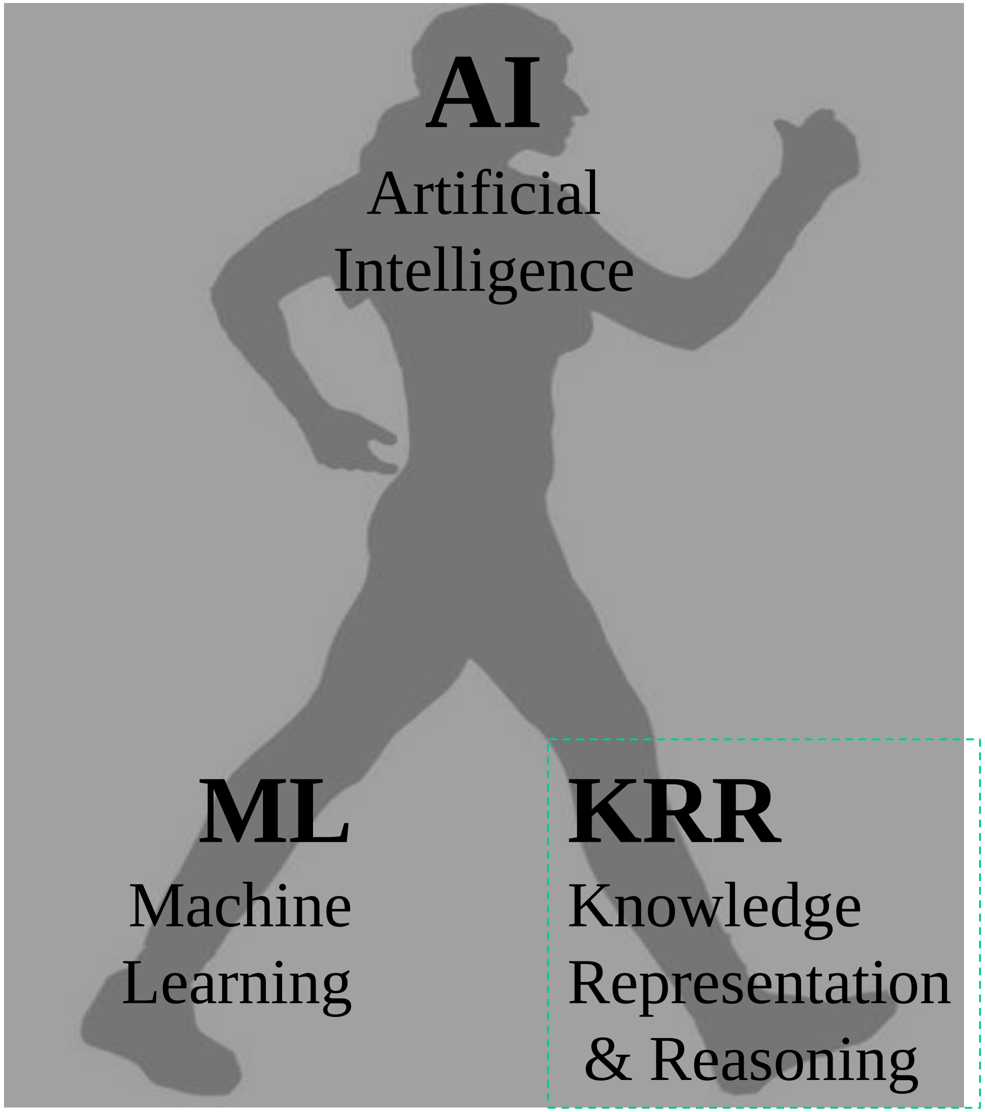
- ML end:
  - Pursue relationships to important specific ML techniques. Including for:
    - Distribution semantics. E.g., cplint, Problog, PRISM.
    - Neural network deep learning. E.g., via t-norms.
  - Apply to constructing knowledge graphs from NL + structured info
    - As at Kyndi. E.g., in entity tagging.

For More Info

- Rulelog detailed tutorial (3 hours) at KR-2018 conference (Oct. 2018):
  - At: http://benjamingrosof.com/misc-publications/#KR2018RulelogTutorial
  - It links to:
  - http://benjamingrosof.com/wp-content/uploads/2018/11/talk-kr2018-rulelog-tutorial-slides-2.pdf
- Invited talk on: why and how to add KRR to ML (July 2018)
  - At: http://benjamingrosof.com/misc-publications

OPTIONAL SLIDES FOLLOW

The Core of AI



It takes two legs to walk

Logic-based KRR's Roles in AI

- Complements ML … in sense of induction from data
  … to enable ML in broader sense

- The power of cultural transmission
  - "Evolution's lesson" (Wolfgang Bibel)

- Accumulate knowledge coherently

- Communicate with humans: expertise, questions

- "Inject" ML results into predictable software

A Mission for AI Reasoning

- Goal to automate:
  - Powerful reasoning and knowledge that <u>combines tightly</u>:
    1. Multiple features within KRR: uncertainty, defeasibility, …
    2. KRR with ML. I.e., deductive and inductive inference.
    3. KRR+ML with NLP. For NL understanding and dialogue.
  - Flexible, deep. Scalable. Composable. Evolvable. Explainable.

- *Thereby enable:*
  - Harness humanity's storehouse of textual knowledge
  - Author at low cost: subject matter expertise
  - Star Trek computer interface
  - Unlock $Trillions of social value in education, healthcare, HCI, customer care, finance & accounting, legal, e-commerce & shopping, science, analytics, workflow & BPM, …

Rulelog - Expressive Extension of LP

- Cleanly semantic. Overall: reduces by efficient transformation to normal LP.
- Higher-order syntax (HiLog), reification; rule id's, provenance
  - Higher-order relies on (logical) functions. Elegant transformation.
- Defeasibility: prioritized defaults, exceptions, argumentation
  - Flexible behavior. Efficient approach. Elegant higher-order "argumentation rules".
  - Both weak negation (\naf) and strong negation (\neg); \naf is outside of \neg
- Restraint bounded rationality: guarantee polynomial time
  - Specify *undefined*-ness in various circumstances, e.g., when term size > threshold.
- General classical-like formulas:
  - Head quantifiers; \exists treated via skolemization
  - Head \or, treated as "omnidirectional" (weak)
- Object-oriented ("frame") syntax
- External queries; import of most kinds of enterprise info ; thus orchestration
- Probabilistic, to a rudimentary extent, via: higher-order, defeasibility
  - Flexible to "roll your own": can have tuple of parameters for the probability
    - pr(*formula1*)[low->0.91,hi->0.94]. pr(*formula2*)[mu->0.925,sigma->0.008].
- Standardization is in process (thru RuleML to W3C and Oasis)

T-norms (Example of Syntax) in PLOW

```
container_contains_hts(Container,HTS) if
      cargo_description_hts(Container,HTS),
      container_xray(Container,HTS).

cargo_description_hts(c1,7208)~0.2.
cargo_description_hts(c1,8481)~0.5.

container_xray(c1,7208)~0.2.
container_xray(c1,8481)~0.1.
```

Skipping Restraint – Example in Process Persistence

```
occurs(e1,0)~1.
occurs(E,T)~P :- T >= 0, integer(T), horizon(Bound), T =< Bound,
                 T1 is T-1, occurs(E, T1)~P1 P is 0.99*P1.
occurs(E,T)~0.5:-horizon(Bound) , T > Bound,
                  occurs(E, Bound)~_ , T =< 2*Bound.
occurs(E,T)~1 :- horizon(Bound),T > 2*Bound,
                 occurs(E, Bound)~_, undefined.
```

Reasoning with Uncertainty using Rulelog

- Treat probabilistic statements as a special case of general logical statements and approximate the desired type of reasoning by incorporating numerical uncertainty weights into the general Ergo reasoning facilities. I.e., "roll-your-own" uncertain reasoning. Axiomatize the semantics in Rulelog itself.
  - Leverages highly expressive features of Rulelog including higher-order syntax
  - Examples of reasoning with uncertainty that can be incorporated into rules:
    - Lower and upper bounds on probability value.
      - E.g., *prob_range(needs_repair(part32), 0.89, 0.94).*
    - Confidence level. E.g., *prob_range_with_confidence(needs_repair(part32), 0.89, 0.94, 0.001).*
    - Source and provenance info about the statistical or ML method used to derive probability value/range. This can be a basis for numerical uncertainty weights.
      - E.g., source(prob_range(needs_repair(part32), 0.89, 0.93), ML_episode(myFavoriteMLClassifier, 'Michael Kifer', 'Feb 11, 2017', http://mycompany.com/dataset41)).

Modified from slide courtesy also of Janine Bloomfield and Theresa Swift

Additional Ways of Reasoning with Uncertainty in ErgoAI

In development:

- "Evidential reasoning": combines information about probabilities based on different conditions and associated data sets
  - Addresses the probability whether a particular "thing" (i.e, a person, situation, etc.) belongs to a particular class.
  - Examples:
    - The probability that person X has a given disease
    - The probability that a given transaction is risky
    - The probability that a given airplane part will fail given its age, type, and manufacturer
  - This type of reasoning does not require complete or even consistent knowledge about probabilities and is scalable

Modified from slide courtesy also of Janine Bloomfield and Theresa Swift

Logic Programs: technical overview (I)

- Knowledge base (KB) is a set of rules, each of form:
  - *Head_formula* :- *Body_formula*.
  - Intuitively: OK to infer (establish) the head if can infer the body
- Basic normal LP: each rule has form:
  - atom :- literal_1 \and … \and literal_m.
  - Plus: atoms are all first-order
- *atom* has form: (predicate(arg_1,…,arg_k)), where each arg_i is a term
- *literal* has form: (atom) or (\naf atom)
- Weak negation: \naf p – p is not believed (essentially, known to be not provable)
- Strong negation: \neg p – p is believed to be strongly false (i.e., opposite of true)
  - Not permitted in normal LP. But permitted in extensions of normal LP, e.g., in Rulelog.
- Aggregation: setof{?x | *condition*}, where ?x appears in *condition* formula
  - Enabled by \naf. Aggregate operators also include avg, max, min, count.
  - average_salary(?co,?amt) :-
    company(?co) \and avg(?amt | employee(?co,?e) \and salary(?e,?amt)}.

Logic Programs: technical overview (II)

- Horn subset: body literals are restricted to be atoms
- Datalog subset: Horn, and function-free
- Full normal LP permits also:
  - in head: \and
  - in body, freely nested: \or, \forall, \exists, aggregates, \and, \naf
  - Integrity constraints via *violation(…)* as a head atom predicate
  - Reduces via transformation to basic normal LP
- Semantics (well-founded) is based on:
  - An alternating least fixed point construction in 3-valued logic
  - Each instantiated atom is assigned to 1 of 3 truth values {t,f,u}:
    - t = *true*; f = "*false*" (cf. \naf); u = "*undefined*" (don't-care).
    - *undefined* is useful for paradox and restraint bounded rationality
- Function-free case is polynomial time
- Functions (thru recursion) lead to undecidability

LP is the Central Form of Practical Logic

- ***LP is the core KR of structured knowledge management today***
- A non-classical logic invented by computer scientists

- Subsets of LP – important in industry landscape today
  - Relational databases (SQL) *[Datalog subset of LP]*
  - Knowledge graphs, a.k.a. graph databases (SPARQL) *[Datalog]*
    - Also: XML databases, object-oriented databases, other semi-structured databases
  - Production rules, Event-Condition-Action rules. More precisely: their logical subsets.
  - Prolog. More precisely: its "pure" logical subset.
  - Ontologies, incl. for knowledge graphs: e.g., OWL-RL, RDF-Schema *[Datalog]*
  - *Industry standards for semantic rules*
    - Many RuleML & RIF dialects, e.g., RIF-BLD, RIF-Core, SWRL

Some State-of-the-Art Semantic KRR Systems

- LP: XSB (Stonybrook U., Theresa Swift, David Warren, et al)
  - Full programming language that is Prolog+
- Rulelog: ErgoAI (Coherent Knowledge, free for research), and its open-source subset Flora-2 (originally Stonybrook U.)
  - Full programming language that is Prolog++ and XSB++
- Probabilistic LP: Problog (Luc de Raedt et al); PRISM (T. Sato et al); PLOW (T. Swift, F. Riguzzi, B. Grosof)
- Probabilistic Soft Logic: (Lise Getoor et al, UC Santa Cruz)
- Markov Logic Networks: Alchemy (Pedro Domingos et al, U. Washington)
- Probabilistic Graphical Models generally: See STARAI workshops

Computation

- XSB has an engine that computes WFS in a scalable manner
  - Applications with up to O(10^8) facts kept in XSB's space. You can of course scale more using database interfaces (or more RAM!)
- The ErgoAI system uses a highly sophisticated compiler from Rulelog to XSB that supports sophisticated program constructs, including adjoint defeasibility theories, quantifiers, a frame-based syntax and much else
- The XSB engine also supports a lattice-theoretic semantics – all of the quantitative logics mentioned above can be modeled by using different lattices. (Bayesian probabilities requires additional infrastructure such as Binary Decision Diagram libraries.)

END OF OPTIONAL SLIDES

Thank You