Towards a Stronger Unified Core for AI: Reasoning and Learning

Benjamin Grosof

Kyndi, Inc.

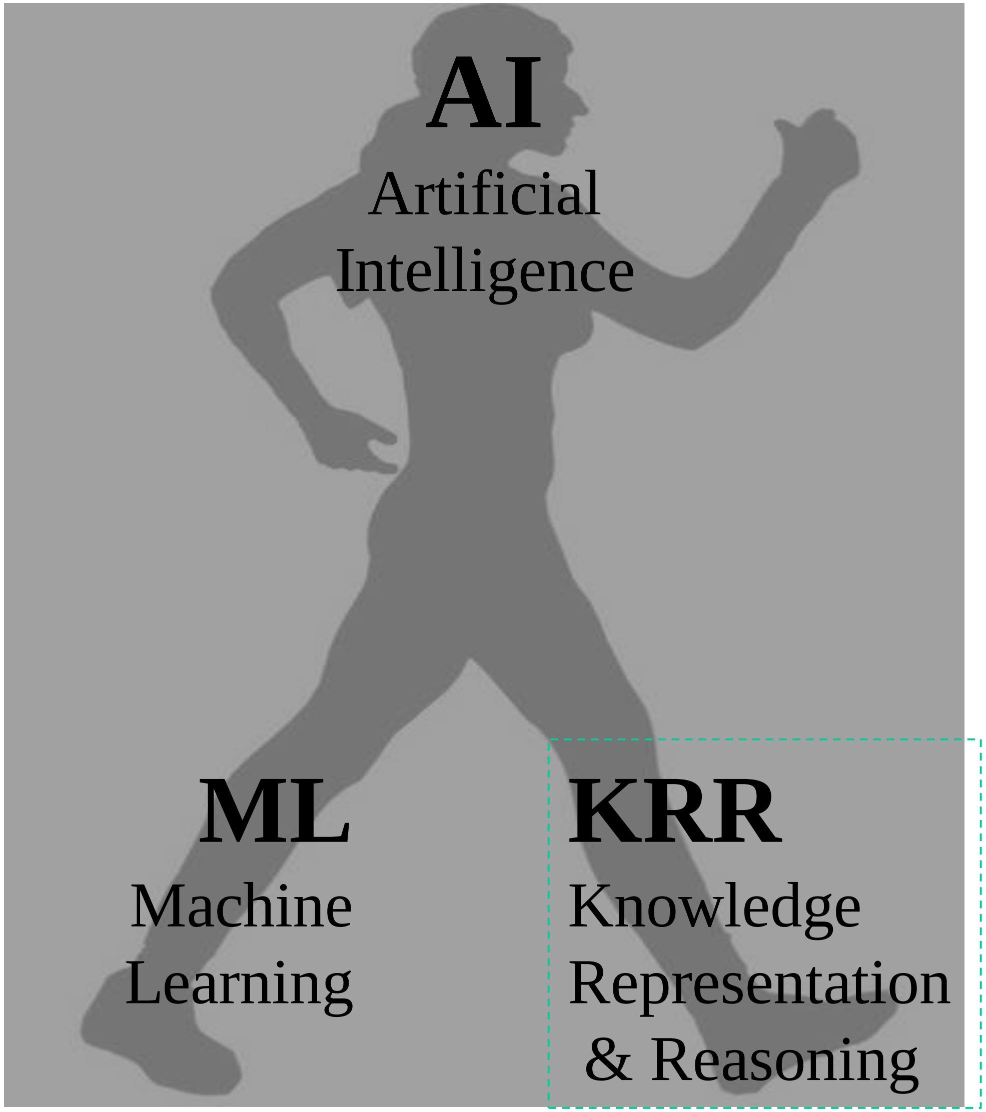
http://benjamingrosof.com

March 6, 2019

This presentation is a work-in-progress, with limited permission for redistribution.

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

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, …

Problem: Analytics for *Complex* Knowledge

Examples: policies, regulations, contracts; terminology mappings; science, causality

Existing *Non-Semantic* Technologies tend to be:

- Shallow
- Siloed
- Costly, and Slow
- Patchily automated
- Opaque
- Inaccurate
- End users not empowered to modify

Based on:
- *Conventional programming languages*
- *Production/ECA rules*
- *Prolog*

Outline of talk

- KRR on trajectory towards tighter combination with ML
  - Mission: strengthen KRR, add to ML, + NLP, for NLU
  - My personal career vector
  - Practical logic. Why add KRR to ML. Desiderata for KRR.
  - Centrality of logic programs (LP) and extensions. LP's spirit.
- Extensions of LP (implemented)
  - Rulelog
  - Textual Rulelog
  - Explanation
  - Probabilistic LP. Additional flavors: PLOW.
- Current Work and promising Future Work directions

Blue indicates areas of design/algorithm/theory contributions by me.

My Bio (I)

- AI researcher, executive, and entrepreneur
- *Currently:*
  - Chief Scientist of Kyndi – AI startup on NL question answering using ML+KRR
  - Co-Founder & Board of Coherent Knowledge – AI startup on KRR
- *Previously:*
  - Principal Director & Research Fellow in AI at Accenture on BPM
  - CTO & CEO of Coherent Knowledge
    - Commercialized Rulelog KRR and initial extensions towards NL
  - Directed advanced AI research at Vulcan: predecessor of Allen Inst. for AI
    - Developed Rulelog KRR theory, algorithms, UI approach
  - MIT Sloan professor and DARPA PI
    - Co-Founder of RuleML, key contributor to W3C KRR standards (esp. OWL-RL, RIF)
  - IBM Research project leader: intelligent agents; biz rules for e-commerce
    - Created IBM Common Rules – 1st successful semantic rules product in industry
  - Stanford AI PhD, combining ML with logical and probabilistic reasoning

My Bio (II)

- *Themes: flexible clean KRR, + ML, + NLP*
  - *ML+KRR at Stanford, IBM, Vulcan, Coherent Knowledge, Accenture, Kyndi*
  - *NLP+KRR at IBM, Vulcan, Coherent Knowledge, Accenture, Kyndi*
  - *ML+NLP+KRR at Vulcan, Accenture, Kyndi*

- Kinds of work:
  - Technical + managerial
  - Core/reusable techniques: invention, design, theory, KB axioms, standards
  - Applications piloting: selection and prototyping, in tandem w/ core
  - Strategy, vision, roadmap, evangelism
  - Steering, cat herding, mentoring
  - Basic research, applied research, advanced development, development
  - Small company, big company, university, non-profit, government contractor

My Bio (III)

- Application domains and tasks:
  - Financial services, accounting, and risk management
  - Compliance with regulations, contracts, and policies
  - Virtual assistants, customer care, helpdesk, chatbots
  - E-commerce, procurement, supply chain, and digital marketing
  - Life science and health care treatment guidance
  - Defense and national intelligence analysis
  - NL question answering, cognitive search, info extraction, associated discovery
  - Privacy/security and trust/sharing management
  - Education and training
  - Workflow and business process management
  - Mobile and personal communications
  - News and publishing
- Application modalities:
  - Decision automation and decision support
  - Query answering. Analytics. Monitoring, including of events. Info flow.

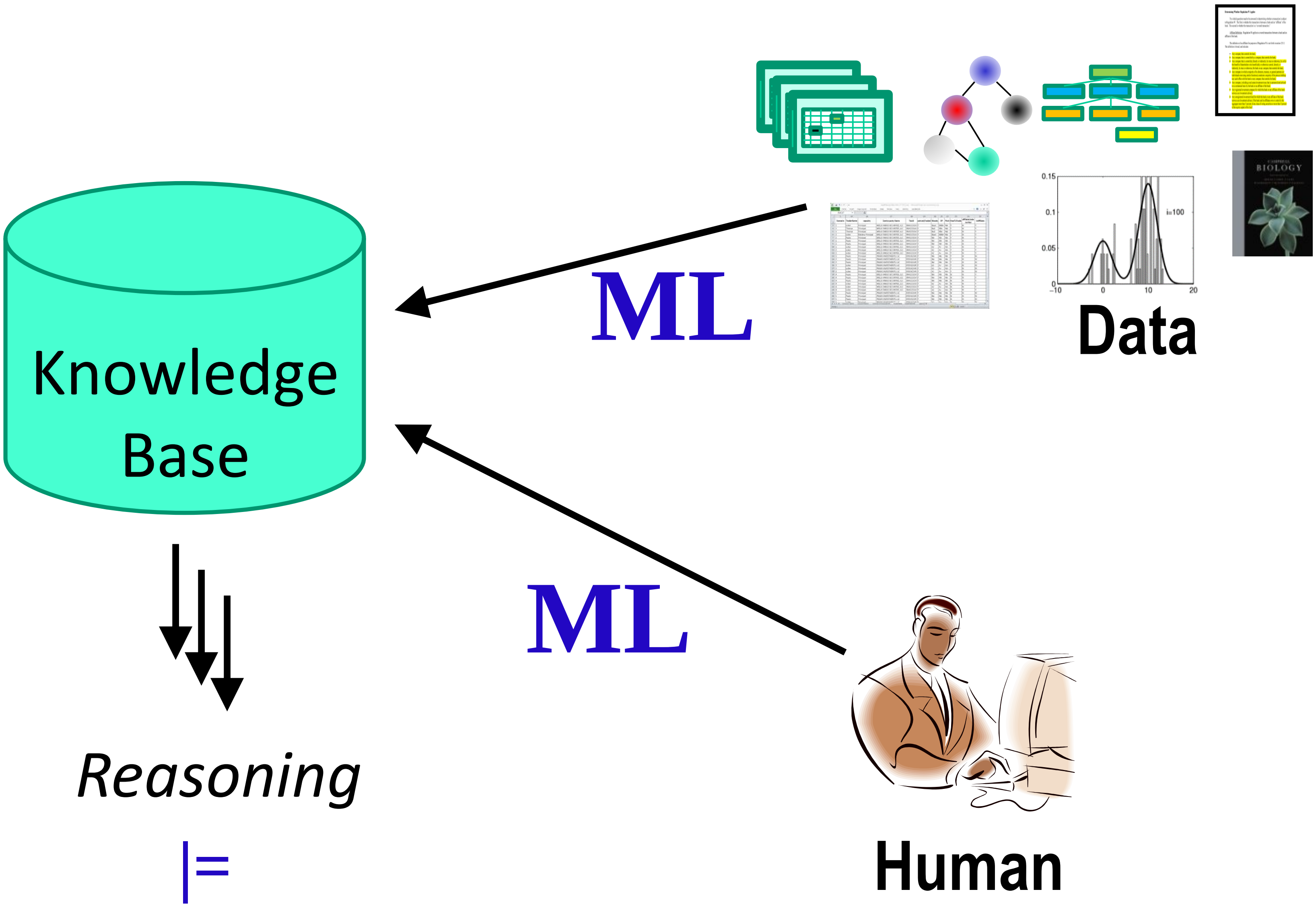
Why Add ML to KRR

2 ways it's useful or even required,
from the viewpoint of KRR, i.e., "for KRR's sake":

1. KB construction: ML is useful to *supply* knowledge

2. *Improve the process of knowledge acquisition*
    - (Can view this as supplying a kind of meta-knowledge)
    - From manual entry of knowledge, e.g., encoding NL into rules
    - From knowledge interchange

Why Add ML to KRR – Summary Diagram



Why Add KRR to ML (I)

10 ways it's useful or even required,
from the viewpoint of ML, i.e., "for ML's sake":

1. The *prediction* step of ML requires reasoning
   - This could be pulled by an ML system via backchaining
   - Why not hook up various external programs such as reasoners, to ML components, e.g., neural networks, to evaluate some nodes/functions?

2. The *target* of ML is a representation

3. Getting business *value* from ML requires reasoning for analysis and decisions

Why Add KRR to ML (II)

4. KRR is required to *combine* results of ML from
   a. Multiple episodes
   b. Multiple sources
   c. Multiple methods

5. KRR is required to *accumulate* knowledge coherently
   - Weakness of ML today
   - Think cultural transmission

Why Add KRR to ML (III)

6. KRR is required to *explain* knowledge understandably to humans
   - Weakness of ML today
   - Needed for humans to trust an automated system
   - Often part of required/desired analysis functionality for own sake

Why Add KRR to ML (IV)

7. Reasoning to *supply derived facts* for ML to chew on as training examples or background info
   - This could be pulled by an ML system via backchaining

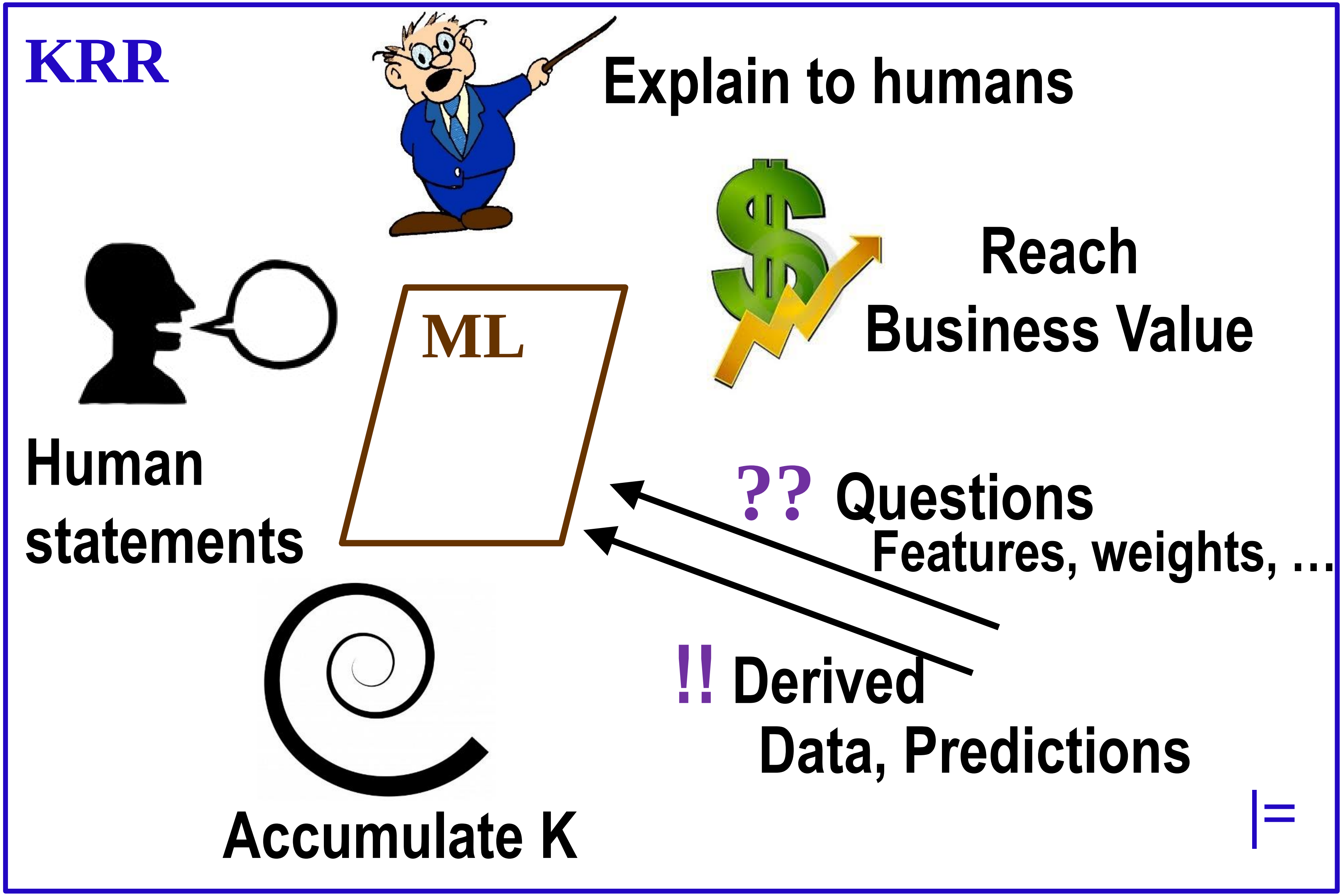
8. *Humans know stuff* beyond what's available via ML training data, and such knowledge is often complex to state / enter
   - KRR methods for entry are often more cost-effective than programming

Why Add KRR to ML (V)

9. Reasoning is desirable to *pose questions* (tasks) to ML
   - as reasoning (sub)goals from KRR

10. Reasoning is desirable to *provide sets of relevant features, (hyper-)parameters, and/or weights* to ML

Why Add KRR to ML – Summary Diagram



KRR <u>Expressive</u> Flexibility Requirements for combination with ML (I)

- Numeric uncertainty (including weighting)
  - Including Bayesian-probabilistic and fuzzy (i.e., triangular norms)
  - Critical for differentiability and neural-network (NN)
- Higher-order syntax and other strong meta
  - Critical for NL, concept mappings, modals (e.g., should)
  - Concept invention in ML. E.g., Inductive LP → Meta Interpretive Learning.
  - Critical for defeasibility, bounded rationality (don't-care), causality
- Defeasibility (exceptions, argumentation, don't-believe)
  - Including prioritization among defaults
  - Treat the evolving character of knowledge and of the world
  - Critical for NL, science, legal/policy, uncertainty
- Quantified formulas (there-exists, for-all, and, or, not, …)
  - Critical for NL

KRR <u>Scalability</u> Requirements for combination with ML (II)

- Scalable computationally
  - To large amounts of asserted and concluded knowledge
  - I.e., to "*volume*" and "*velocity*"
  - Inner loop in many ML techniques/algorithms is deductive
  - Generally, beyond ML: for KB development test-edit cycle

- Scalable "socially" to multiplicity of diverse knowledge
  - To multiplicity of diverse ML/etc. <u>sources, e.g., organizations</u>
  - To multiplicity of diverse ML/etc. <u>algorithmic methods</u>
  - To multiplicity of diverse underlying ML <u>data samples</u>
  - I.e., to "*variety*" including heterogeneity
  - Robust in face of conflict from diverse sources of knowledge

Practical Logic, vs. Classical Logic

- Goal: support IT, vs. mathematics
  - E.g., Databases, Rules
  - Central: declarative logic programs (LP) KR
  - Desirable to have expressiveness enough for
    - Natural language (NL) cognitive abstraction level, structures, semantics
    - Machine learning (ML) results

- Requirements:
  - Scalable computationally
  - Robust in face of human errors and miscommunications
  - →→ "Humble"
    - Avoid general reasoning by cases
    - Avoid general proof by contradiction

What is "reasoning by cases": (background)

Assertions: if A then C. if B then C. A <u>or</u> B.

Conclude: C.

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

Logic Programs Example

- ":-" means "if", i.e., \impliedby
- *Assertions:*
  - human(Socrates). human(Anne). human(Peter).
  - modern(Anne). educated(Socrates).
  - mortal(?x) :- human(?x).
  - fallible(?z) :- mortal(?z).
  - educated(?x) :- human(?x) \and modern(?x).
  - humble(?w) :- fallible(?w) \and educated(?w).
- *Forward chaining:* mortal(Socrates). mortal(Anne). mortal(Peter). fallible(Socrates). fallible(Anne). fallible(Peter). educated(Anne). humble(Socrates). humble(Anne).
- *Query:* ?- humble(?p). **Answer:** {Socrates, Anne}
  - Backward chaining subgoals:
    ?- fallible(?p), ?- educated(?p).
  - ?- mortal(?p). ?p / S
  - ?- human(?p). ?- human(?p), ?- modern(?p).
  - ?p / S, A, P ?p / S, A, P ?p / A

Industry Landscape of Practical KRR

- LP and subsets *(cf. earlier slide)*
- Subsets of Classical Logic:
  - Propositional. E.g., hardware circuit design, satisfiability for planning.
  - First Order Logic (Common Logic). E.g., for program verification.
  - Description Logic (OWL) subset of FOL. For ontologies.
- Emerging: (in roughly descending order of commercial maturity)
  - Rulelog – extension of LP
    - RIF/RuleML Rulelog dialect standard is in draft
  - Bayesian Probabilistic LP
    - Bayesian Networks are a special case
  - Fuzzy Probabilistic LP
    - Probabilistic Soft Logic – is closely related
  - Other probabilistic graphical models (PGM)
    - Markov Logic Networks – closer to classical; thus more difficult to scale
  - Answer Set Programs – closer to classical, less humble
  - (Others are not so commercially/practically prominent)

Some State-of-the-Art 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

Outline of talk

- KRR on trajectory towards tighter combination with ML
- Extensions of LP (implemented)
  - Review of normal LP
  - Rulelog expressiveness, scalability, algorithms
    - Defeasibility. Higher-order. Transformational stack.
    - Restraint bounded rationality. Dependency-aware cacheing.
  - Concept and vision: Textual Rulelog and human-machine logic
  - Mapping text syntax ← → logical syntax, via textual templates
  - Explanations cf. proof
  - Applications for Rulelog
  - Probabilistic LP (PLP) incl. Bayesian, triangular norms, intervals, lattices
- Current Work and promising Future Work directions

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

The "Spirit" of LP

The following summarizes the "spirit" of how LP differs from FOL:

- "Avoid Disjunction"
  - <u>Avoid disjunctions of positive literals</u> as expressions
    - In premises, intermediate conclusions, final conclusions
    - (conclude (A or B))   only if   ((conclude A) or (conclude B))
  - Permitting such disjunctions creates exponential blowup
    - In propositional FOL: 3-SAT is NP-hard
    - In the leading proposed approaches that expressively add disjunction to LP with negation, e.g., propositional Answer Set Programs
  - No "reasoning by cases", therefore
- "Stay Grounded"
  - Avoid (irreducibly) non-ground conclusions

LP, unlike FOL, is straightforwardly extensible, therefore, to:

- Nonmonotonicity – defaults, incl. NAF
- Procedural attachments, esp. external actions

Rulelog - Expressive Extension of LP

- Overall: reduces by efficient transformation to normal LP
- Higher-order, 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 an initial extent, via: higher-order, defeasibility
  - But current implementations not optimized and lack dedicated syntax/structuring
  - 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].

Rulelog ***Scalability***

- Computationally scalable, despite very high expressiveness
  - Keys: database logic (LP) spirit + bounded rationality
  - Reasoning is polynomial time*, as in databases
  - Millions of sentences concluded/asserted on a single processor
  - Up to trillions by orchestrating database etc. systems in distributed settings

- Has capable efficient algorithms AND implementations
  - Dynamic compilation/transformation stack architecture
  - Cacheing of successful & failed inferences, with dependency-awareness, subgoal reordering, analysis of cycles and depths
  - Indexing, tries, other low-level data structures
  - Leverages database and "tabling" techniques

* if, as is typical, one employs the radial restraint feature

HiLog Transformation

- High-level Spirit:
  ?pred(?arg1,?arg2) → → believe(?pred,?arg1,?arg2)

- A simplified version of the transformation, which gives intuition:
  - Rewrite each atom p(a,b) → holds_2(p,a,b)
    - Generic predicate constants holds_1, holds_2, …
  - Treat each term in similar manner
    - f(a,b) → apply_2(f,a,b)
    - Generic function constants apply_1, apply_2, …

Representational Uses of Hidlog

- Hidlog (pronounced "High-Dee-Log") = HiLog + reification + rule id's
- For meta- reasoning, e.g., in knowledge exchange or introspection
  - Meta-data is central to the Web
- Modals, e.g., believe, permit. Multi-agent belief. Deontics.
- Defeasibility: principles of argumentation/debate, i.e., argumentation rules
- Restructuring in mapping of schemas, ontologies, and terminologies
  - E.g., in knowledge integration, federation, KB translation/import
- Conciseness. Simple example: transitive closure of a relation.
- Reasoning control
- Modularization of KB's
- Context, including provenance
- KR macros
- Probabilistic/uncertainty range and confidence about a sentence
  - E.g., import of machine learning (ML) results
- Representing natural language (NL). E.g., compositionality of phrases:
  - Compounding of nouns; salesman(?x); (insurance(salesman))(?x); ((life(insurance))(salesman))(?x)
  - Adverbial modification: quickly(give)(?thing,?recipient)

Argumentation Rules approach to Defaults in LP

- **Combines Courteous + HiLog, and generalizes**
- **New approach to defaults: "argumentation rules"**
  - Meta-rules, in the LP itself, that specify when rules ought to be defeated
  - [Wan, Grosof, Kifer, *et al.* ICLP-2009; RR-2010; SWJ 2015]
- **Extends straightforwardly to combine with other key features**
  - E.g., Frame syntax, external Actions
- **Significant other improvements on previous Courteous**
  - Eliminates a complex transformation
  - Much simpler to implement
    - 20-30 background rules instead of 1000's of lines of code
  - Much faster when updating the premises
  - More flexible control of edge-case behaviors
  - Much simpler to analyze theoretically

LPDA Approach, Continued*

- **More Advantages**
  - 1st way to generalize defeasible LP, notably Courteous, to HiLog higher-order and F-Logic frames
  - Well-developed model theory, reducible to normal LP
  - Reducibility results
  - Well-behavior results, e.g., guarantees of consistency
  - Unifies almost all previous defeasible LP approaches
    - Each reformulated as an argumentation theory
    - E.g., Defeasible Logic (see Wan, Kifer, and Grosof RR-2010 / SWJ 2015 paper)
  - Cleaner, more flexible and extensible semantics
    - Enables smooth and powerful integration of features
    - Applies both to well founded LP (WFS) and to Answer Set Programs (ASP)
  - Leverages most previous LP algorithms & optimizations

- **Implemented** in Ergo, also earlier in Flora-2 and used in SILK

* LP with Defaults and Argumentation Rules

Example – AT for Courteous ($\mathcal{AT}^{\mathcal{GCLP}}$)

```
defeated(?R)      :-  defeats(?S, ?R).
defeats(?R, ?S) :-  refutes(?R, ?S) or $rebuts(?R, ?S).
```

Prioritization (user specified)

```
refutes(?R, ?S) :-  conflict(?R, ?S), \overrides(?R, ?S).
refuted(?R)        :-  refutes(?R2, ?R).
rebuts(?R, ?S)  :-  conflict(?R, ?S),
                            naf refuted(?R), \naf refuted(?S).
```

Default negation (NAF)

Meta predicates (“Reflection”)

```
candidate(?R)    :-  body(?R, ?B), call(?B).
conflict(?R, ?S)  :-  candidate(?R), candidate(?S),
                               \opposes(?R, ?S).
\opposes(?R, ?S)    :-  \opposes(?S, ?R).
\opposes(?L1,?L2) :-  head(?L1, ?H), head(?L2, \neg ?H).
```

Exclusion (user specified)

Explicit negation

Modified from slide courtesy also of Hui Wan

Ecology Ex. of Causal Process Reasoning

```
/*   Toxic discharge into a river causes fish die-off.   */
/* Init. facts, and an "exclusion" constraint that fish count has a unique value */
 occupies(trout,Squamish).
 fishCount(0,Squamish,trout,400).  / * 1st argument of fishCount is an integer time */
 \opposes(fishCount(?s,?r,?f,?C1), fishCount(?s,?r,?f,?C2)) :- ?C1 != ?C2.
/* Action/event description that specifies causal change, i.e., effect on next state */
@{tdf1} fishCount(?s+1,?r,?f,0) :- occurs(?s,discharge,?r) \and occupies(?f,?r).
/* Persistence ("frame") axiom */
 @{pefc1}  fishCount(?s+1,?r,?f,?p) :- fishCount(?s,?r,?f,?p).
/* Action effect axiom has higher priority than persistence axiom */
 \overrides(tdf1,pefc1).
/* An action instance occurs */
 @{UhOh}  occurs(1,toxicDischarge,Squamish).
```

As desired: |= **fishCount(1,Squamish,trout,400),**
fishCount(2,Squamish,trout,0)

Notes: @… declares a rule tag. ? prefixes a variable. :- means if. != means ≠. \opposes indicates an exclusion constraint between two literals, which means "it's a conflict if".

Concept of Humagic Knowledge

- Humagic = <u>hu</u>man-<u>ma</u>chine lo<u>gic</u>
- A humagic KB consists of a set of linked sentences
  - Assertions, queries, conclusions (answers & explanations)
- NL-syntax sentence may have 1 or more logic-syntax sentences associated with it
  - E.g., that encode it, or give its provenance, or represent its text interpretation
- Logic-syntax sentence may have 1 or more NL-syntax sentences associated with it
  - E.g., results of text generation on it
  - E.g., source sentence in text interpretation, that produced it
- Other sentences can be in a mix of NL-syntax and logic-syntax
  - Using textual templates, for text interpretation and text generation

Textual Rulelog (I)

- *Textual* Rulelog extends Rulelog with natural language processing (NLP)
  - Rulelog itself is utilized to map Rulelog logic syntax ← → NL syntax
  - I.e., to do text interpretation and text generation
  - Textual templates aid knowledge entry and explanation generation
- Examples of hybrid-syntax sentences that are executable in ErgoAI:
  - \(The individual affiliate threshold for transaction under Regulation W by ?Bank with ?Counterparty is ?Amount\) :-

    \(?Counterparty is deemed an affiliate of ?Bank under Regulation W\) \and
    \(?Bank has capital stock and surplus ?Capital\) \and
    \(the threshold percentage for an individual affiliate is ?Percentage\) \and
    ?Amount \is ?Capital * ?Percentage/100.

  - @{'each large company has some talented CEO'}
    - forall(?x)^((?x \isa \(large company\)) ==>
      exists(?y)^((\?x has ?y\) \and
      (?y \isa \(talented CEO\)))).

Textual Rulelog (II)

- Textual interpretation and text generation mapping is much simpler and closer than with other KR's
  - Rulelog's high expressiveness is much closer to NL's conceptual abstraction level
  - More often doable and useful:

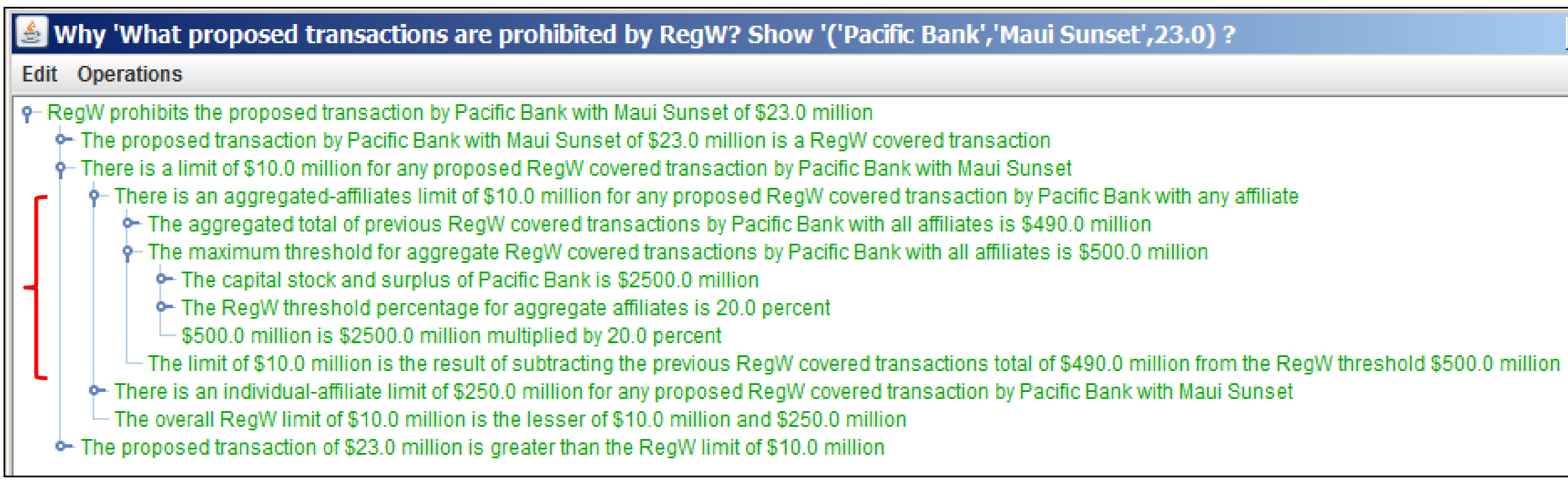
    1 English sentence ↔ 1 Rulelog sentence (rule)

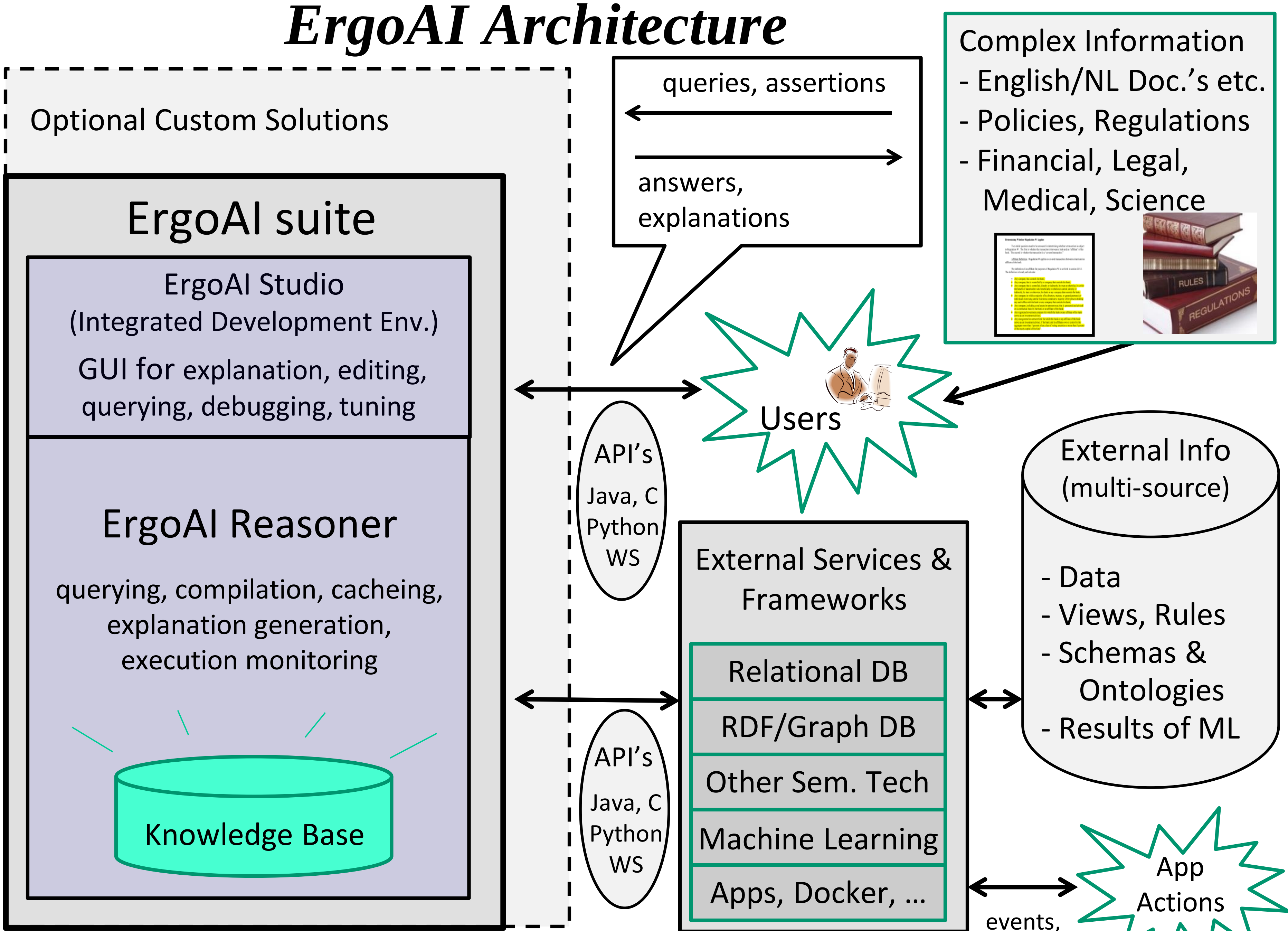
- In principle, almost any NL sentence can be represented with deep semantics as a logical sentence in Rulelog
  - Leverage the general quantified formulas expressive feature of Rulelog

Example of Explanation in Rulelog

Why is the proposed transaction prohibited by Regulation W?

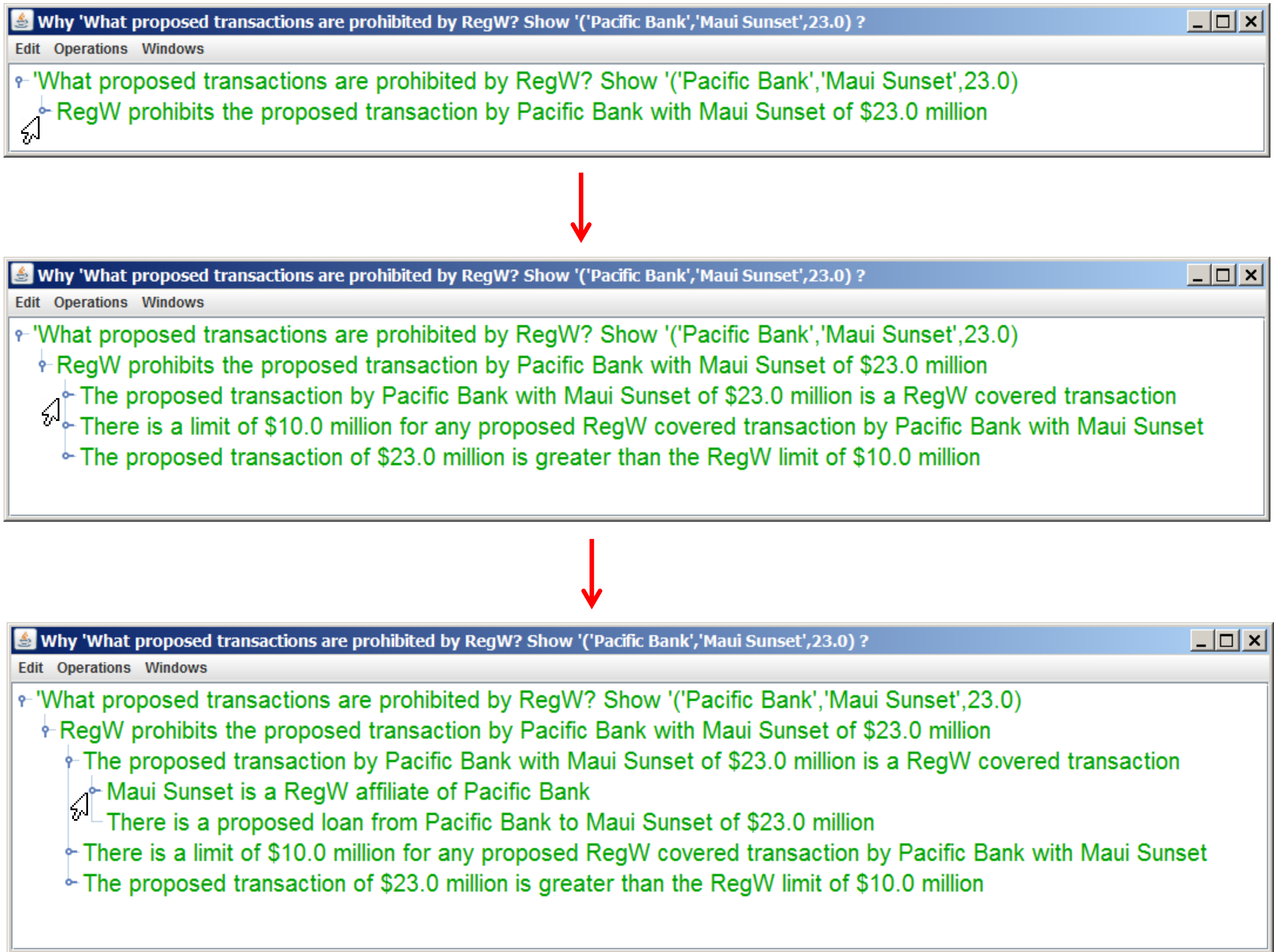
With drill down on: Why is the aggregate-affiliates limit $10 million?





WS = Web Services. Sem. = Semantic. ML = Machine Learning

User Clicks the handles to expand the Explanations



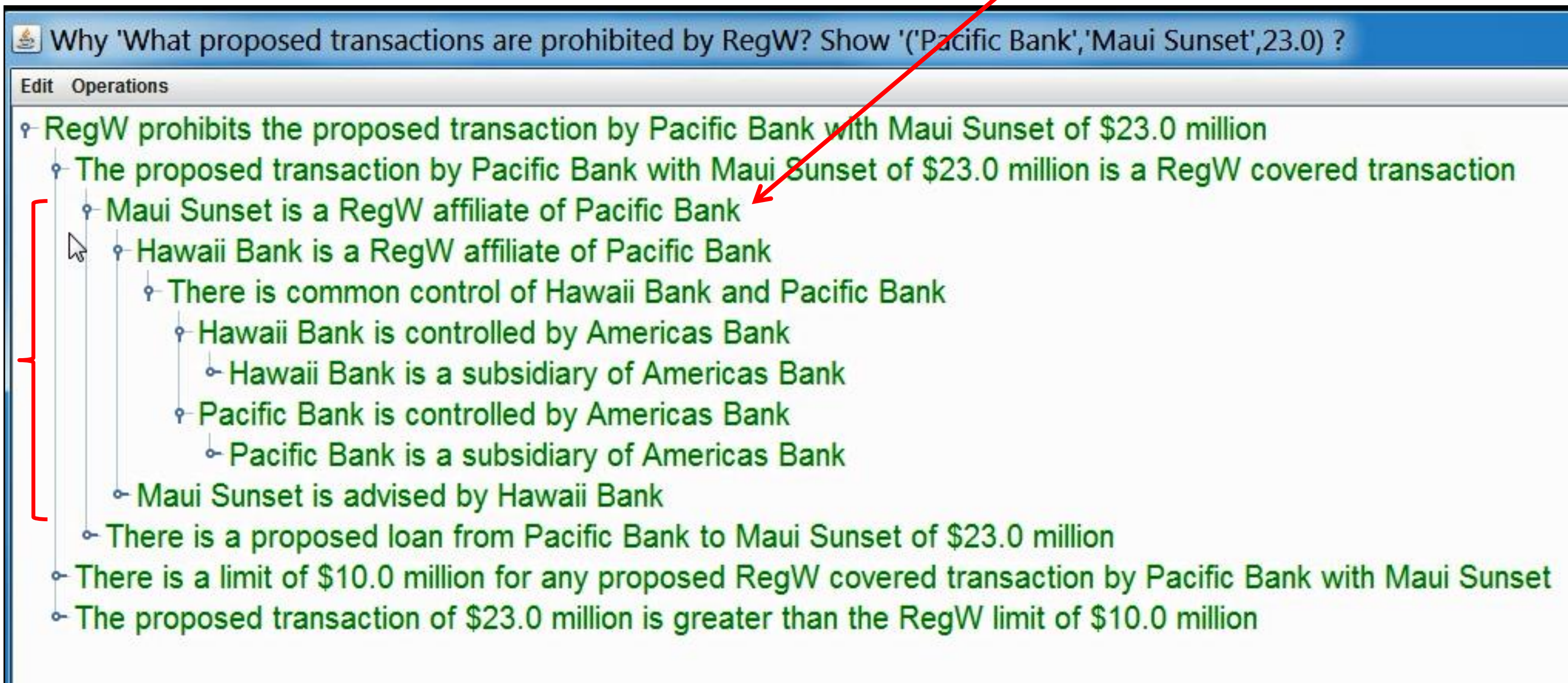
Slide courtesy of: Coherent Knowledge

42

Why is the proposed transaction prohibited by Regulation W?

1. *Is the transaction's counterparty an "<u>affiliate</u>" of the bank?*

YES.



And here's why …

Slide courtesy of: Coherent Knowledge

43

Executable Assertions: non-fact **Rules**

```
/* A company is controlled by another company when the first company
    is a subsidiary of a subsidiary of the second company. */
@!{rule103b} /* declares rule id */
@@{defeasible} /* indicates the rule can have exceptions */
 controlled(by)(?x1,?x2)
 :- /* if */
   subsidiary(of)(?x1,?x3) \and
   subsidiary(of)(?x3,?x2).

/*A case of an affiliate is: Any company that is advised on a contractual basis by
   the bank or an affiliate of the bank. */
@!{rule102b} @@{defeasible}
 affiliate(of)(?x1,?x2) :-
  ( advised(by)(?x1,?x2)
     \or
    (affiliate(of)(?x3,?x2) \and advised(by)(?x1,?x3))).
```

Executable Assertions: **Exception** Rule

```
@!{rule104e}
@{‘ready market exemption case for covered transaction'} /* tag  for prioritizing */
\neg covered(transaction)(by(?x1))(with(?x2))
         (of(amount(?x3)))(having(id(?Id))) :-
  affiliate(of)(?x2,?x1) \and
  asset(purchase)(by(?x1))(of(asset(?x6)))(from(?x2))(of(amount(?x3)))
     (having(id(?Id))) \and
  asset(?x6)(has(ready(market))).

/* prioritization info, specified as one tag being higher than another */
\overrides(‘ready market exemption case for covered transaction',
                'general case of covered transaction').

/* If a company is listed on the New York Stock Exchange (NYSE), then the
   common stock of that company has a ready market. */
@!{rule201} @@{defeasible}
asset(common(stock)(of(?Company)))(has(ready(market))) :-
      exchange(listed(company))(?Company)(on('NYSE')).
```

Reg W Example Sentence using Templates

- Example of hybrid-syntax sentence – <u>executable</u> in ErgoAI:

```
\(The individual affiliate threshold for transaction under Regulation W
by ?Bank with ?Counterparty is ?Amount\)  :-
        \(?Counterparty is deemed an affiliate of ?Bank under Regulation W\) \and
        \(?Bank has capital stock and surplus ?Capital\) \and
        \(the threshold percentage for an individual affiliate is ?Percentage\) \and
        ?Amount  \is  ?Capital * ?Percentage/100.
```


46

Reg W example Template definition in ErgoAI

```
template(headbody,
         \(The proposed transaction ?Id by ?Bank with ?Affiliate of $?Amount
           is a RegW covered transaction\),

         covered(proposed(transaction))(by(?Bank))(with(?Affiliate))
           (of(amount(?Amount)))(having(id(?Id)))
         ).
```

- The templates are self-documenting

47

Terminology Mapping in Textual Rulelog

- Rulelog rules map between phrasings and ontologies – in NL or logic
  - “moving a bomb” implies “transporting weaponized material”
  - isBomb(?x) implies rdftriple(?x,rdftype,bomb)

48

Some Techniques for Textual Rulelog

1. Word as functor (WAF): treat a NL word as a logical functor
2. Phrasal terms (phrasts): treat a NL phrase as a logical term
3. Phrasal mapping from paraphrase knowledge (PMK): e.g., synonyms, hypernyms, hyponyms, equivalent named entities; other implications b/ phrases
4. Textual templates (TET): hybrid of text syntax and logic syntax
5. Quantification of NL determiners (QUD): e.g., treat “every” and “some” as relativized universal and existential logical quantifiers
6. Deep extended NL parsing (DEP): logically represent dependency parse tree extended with coreference analysis and named entity recognition

- *These can be combined*
  - *Many interesting directions & open areas for research! E.g., DEP and QUD.*

Example Application Areas for Rulelog

- Confidentiality policies
- Financial/business reporting
- Contracts
- E-commerce pricing/promotion policies
- E-commerce product catalog integration, supply chain
- Financial regulatory/policy compliance
- Health treatment guidance, insurance
- Defense intelligence analysis
- Education/e-learning: personalized tutoring in sciences
- Info/system integration, e.g., in financial, defense

- Potentially many more: natural language interaction, business intelligence, games, workflow, social media,…

Outline of talk

- KRR on trajectory towards tighter combination with ML
- Extensions of LP (implemented)
  - Review of normal LP
  - Rulelog expressiveness, scalability, algorithms
    - Defeasibility. Higher-order. Transformational stack.
    - Restraint bounded rationality. Dependency-aware cacheing.
  - Concept and vision: Textual Rulelog and human-machine logic
  - Mapping text syntax ← → logical syntax, via textual templates
  - Explanations cf. proof
  - Applications for Rulelog
  - Probabilistic LP (PLP) incl. Bayesian, triangular norms, intervals, lattices
- Current Work and promising Future Work directions

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

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 <u>flavors</u> 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. <u>Generalized</u> “<u>triangular norms</u>”, 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 <u>intervals</u>, where the interval is <u>cautious</u> in regard to the potential <u>correlation</u> 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 .

PLOW system for Probabilistic LP

- The first to implement the generalized t-norm flavor
- Bayesian flavor, 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!

T-norms

- Full Bayesian reasoning is powerful but (computationally) expensive.
- Epistemically, probabilities may not be a good way to represent similarity measures. 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": generalizes the measure to an interval [Lukasiewicz, Godel] expressing an interval of truth, cautious in regard to how much correlation of A and B.

  - Implementation is not complete, as Lukasiewicz and Product are not idempotent for e.g., conjunction.

PLOW Uses – More

- Vague properties – e.g., a certain person may be more or less ”tall”
- Similarity relations – e.g., two documents may be more or less related
- 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

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.
```

Explicit (Strong) Negation in PLOW

- Default negation may be denoted via naf, and explicit negation via neg. Modifying the previous 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: 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

Outline of talk

- KRR on trajectory towards tighter combination with ML
  - Mission: strengthen KRR, add to ML, + NLP, for NLU
  - My personal career vector
  - Practical logic. Why add KRR to ML. Desiderata for KRR.
  - Centrality of logic programs (LP) and extensions. LP's spirit.
- Extensions of LP (implemented)
  - Rulelog. Defeasibility, higher-order syntax, restraint bounded rationality, …
  - Textual Rulelog
  - Explanation
  - Probabilistic LP. Generalized triangular norms: in PLOW.
- Current Work and promising Future Work directions

Blue indicates areas of design/algorithm/theory contributions by me.

Current and promising Future Directions

- Mature PLOW esp. generalized t-norms
- Vision: network of learning and reasoning components feeding each other
- Neural Nets – LP hybrid ; leverage PLOW generalized t-norms as part
- PLP based learning, including with generalized t-norms
- NLU via extended dependency parsing based on KRR and ML
- … Including to deepen and mature Textual Rulelog
- Explanation beyond proof, covering ML: e.g., focus and influence
- Explanation filtering and customization
- Trustworthiness via adding KRR to ML, e.g., ethical/policy/legal guidance
- Application areas for combining KRR with ML, often including explanation

Example Application Areas for Combining KRR with ML, often including Explanation

- Knowledge graphs with ML-based query scoring, ML-based info extraction, and ontologies from both human authoring and ML, e.g., for national intelligence analysis, cf. Kyndi.
- Health treatment guidance
- Collaborative argumentation in: science; national intelligence analysis
- Helpdesk and personal assistants – esp. conversational
- Policy/legal: compliance with suspicion/fraud/launder; contract obligation/violation
- Accounting, tax
- Education/training: with path guidance, explanation-centric conversation

Quantification of Determiners

- Determiners in NL often indicate quantification
  - E.g., treat "*each*" via a logical relativized *universal* quantifier
    - forall(?var)^((?var \isa *fooNounPhrase*)) ==> *barVerbPhraseAbout?Var*)
  - E.g., treat "*some*" via a logical relativized *existential* quantifier
    - exists(?var)^((?var \isa *fooNounPhrase*)) \and *barVerbPhraseAbout?Var*)

- Ex.: "each large company has some talented CEO"

```
exists(?y)^( (?y \isa \(talented CEO\)) \and
                 (\?x has ?y\) )
  :- ?x \isa \(large company\) .

/* above has implicit outermost universal quantification of ?x */
```

Deep Extended NL Parsing (DEP) in Textual Rulelog

- Common useful form of NLP output is "parse++" in following form:
  - NL dependency parsing,
  - extended with coreference analysis (Coref) and named entity recognition (NER)
- DEP: Represent the results of parse++ as a set of logical facts specifying the parse++ tree in full detail
  - Tree structure: dependency edges, left vs. right sequencing, edge labels
  - Node labels: word token, part of speech (PoS), NER, Coref, other word sense
  - Provides grist for deep semantic text interpretation & representation in Rulelog

- ErgoNLP is a recently released open source tool implementing DEP
  - It uses the popular Stanford CoreNLP toolset, which is open source (GPL)
  - Inputs a passage of 1 or more English sentences
  - Outputs ErgoAI facts that represent the parse++ of each sentence
    - Actually uses only the Ergo Lite subset of ErgoAI's syntax & expressiveness
  - https://bitbucket.org/coherentknowledge/ergonlp
  - Implemented in Java. Original authors Coherent Knowledge. Apache 2.0 license.

 65

Example of DEP in ErgoNLP

// Input Sentence 1: Each large company has some talented CEO.

```
ph(104)[ ws(wd)->'has',
  root -> \true,
  ws(PoS)->'VBZ',
  ldp(1)->dp(nsubj,ph(103)),
  rdp(1)->dp(dobj,ph(107))].

ph(103)[ ws(wd)->'company',
  ws(PoS)->'NN',
  ldp(1)->dp(det,ph(101)),
  ldp(2)->dp(amod,ph(102))].

ph(107)[ ws(wd)->'CEO',
  ws(PoS)->'NN',
  ldp(1)->dp(det,ph(105)),
  ldp(2)->dp(amod,ph(106)),
  rdp(1)->dp(rcmod,ph(109))].

ph(101)[ ws(wd)->'Each',
  ws(PoS)->'DT'].

ph(102)[ ws(wd)->'large',
  ws(PoS)->'JJ'].

ph(105)[ ws(wd)->'some',
  ws(PoS)->'DT'].

ph(106)[ ws(wd)->'talented',
  ws(PoS)->'JJ'].
```

// Input Sentence 2: IBM is a huge company.

```
ph(205)[ ws(wd)->'company',
  root -> \true,
  ws(PoS)->'NN',
  ldp(1)->dp(nsubj,ph(201)),
  ldp(2)->dp(cop,ph(202)),
  ldp(3)->dp(det,ph(203)),
  ldp(4)->dp(amod,ph(204)),
  ws(coref)->ph(201)].

ph(201)[ ws(wd)->'IBM',
  ws(PoS)->'NNP',
  ws(ne)->ORGANIZATION].

ph(202)[ ws(wd)->'is',
  ws(PoS)->'VBZ'].

ph(203)[ ws(wd)->'a',
  ws(PoS)->'DT'].

ph(204)[ ws(wd)->'huge',
  ws(PoS)->'JJ'].
```

The example's input text passage is the 2 English sentences.

There's one frame set of facts for each word.

There's one set of frame sets for each English sentence.

In Ergo frame syntax:

  subject[property->value]

is a fact triple that is similar to

  property(subject,value)

in predicate calculus syntax; and

  subj[prop1->val1, prop2->val2, prop3->val3]

is logically equivalent to the 3 fact triples

  subj[prop1->val1] \and subj[prop2->val2] \and subj[prop3->val3].

ph(104) stands for phrasal term number 104. Each such phrast represents a node in the parse++ tree, and also corresponds to a parse++ subtree rooted at that node.

ws stands for word sense info.

ws(wd) stands for the word token.

ws(PoS) stands for part of speech.

ldp stands for left dependency list.

rdp stands for right dependency list.

ldp(1) stands for first left dependency, as one goes left-to-right; ldp(2) stands for the second, etc.

rdp(1) stands for first right dependency, as one goes left-to-right; rdp(2) stands for the second, etc.

dp(*dependency_label,* ph(*number*)) stands for a dependency edge, with a particular edge label, to a particular node.

The dependency edge labels, and PoS node labels, are the usual Penn TreeBank ones. E.g., nsubj for subject NP, JJ for adjective.

66

Outline of talk

- KRR on trajectory towards tighter combination with ML
  - Mission: strengthen KRR, add to ML, + NLP, for NLU
  - My personal career vector
  - Practical logic. Why add KRR to ML. Desiderata for KRR.
  - Centrality of logic programs (LP) and extensions. LP's spirit.
- Extensions of LP (implemented)
  - Rulelog. Defeasibility, higher-order syntax, restraint bounded rationality, …
  - Textual Rulelog
  - Explanation
  - Probabilistic LP. Generalized triangular norms: in PLOW.
- Current Work and promising Future Work directions

Blue indicates areas of design/algorithm/theory contributions by me.

OPTIONAL SLIDES FOLLOW

Executable Assertions: **Import of OWL**

```
:- iriprefix fibof =  /* declares an abbreviation */
    "http://www.omg.org/spec/FIBO/FIBO-Foundation/20120501/ontology/".

/* Imported OWL knowledge:  from Financial Business Industry Ontology (FIBO) */
rdfs#subClassOf(fibob#BankingAffiliate, fibob#BodyCorporate).
rdfs#range(fibob#whollyOwnedAndControlledBy, fibob#FormalOrganization).
owl#disjointWith(edmc#Broad_Based_Index_Credit_Default_Swap_Contract,
                 edmc#Narrow_Based_Index_Credit_Default_Swap_Contract).

/* Ontology Mappings between textual terminology and FIBO OWL vocabulary */
company(?co) :- fibob#BodyCorporate(?co).
fibob#whollyOwnedAndControlledBy(?sub,?parent) :- subsidiary(of)(?sub,?parent).

/* Semantics of OWL - specified as general Rulelog axioms */
?r(?y) :- rdfs#range(?p,?r), ?p(?x,?y).
?p(?x,?y) :- owl#subPropertyOf(?q,?p), ?q(?x,?y).
```

Physics Ex. of Contextual Assumptions

/* "P8: Joe drops a glove from the top of a 100m cliff.
How long does the fall take in seconds?" */

// Initial problem-specific facts

```
AP_problem(P8);  fall_event(P8);  P8[height->100].
// Action description that specifies causal implications on the continuous process
?e[time->((2 * ?h / ?n)^0.5)] :- fall_event(?e) \and ?e[height->?h,  net_accel->?n].
?e[net_accel->(?g - ?a)] :- fall_event(?e) and
                            ?e[gravity_accel->?g,  air_resistance_accel->?a].
// Other facts
?e[gravity_accel->9.8] :- loc(?e, Earth).
?e[gravity_accel->3.7] :- loc(?e, Mars).
// Contextual assumptions for answering Advanced Placement exam (AP) problems
@{implicit_assumption} loc(?e, Earth) :- AP_problem(?e).
\opposes(loc(?e, Earth), loc(?e, Mars)).
@{implicit_assumption} ?e[air_resistance_accel->0] :- AP_problem(?e).
\overrides(explicitly_stated, implicit_assumption).
```

As desired: **|= P8[net_accel->9.8, time->4.52]   // 4.52 = (2*100/9.8)^0.5**

Physics Ex. of Contextual Assumptions (in Ergo)

/* "P8: Joe drops a glove from the top of a 100m cliff on Mars.
How long does the fall take in seconds?" */

/* Initial problem-specific facts*/

AP_problem(P8). fall_event(P8). P8[height->100].

@{explicitly_stated} loc(P8,Mars).

…

As desired: **|= P8[net_accel->3.7, time->7.35] // 7.35 = (2*100/3.7)^0.5**

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

Thank You

Disclaimer: All brands, logos and products are trademarks or registered trademarks of their respective companies.