Description Logic Programs: Combining Logic Programs with Description Logic

Presentation for WWW-2003 Conference
May 21, 2003, Budapest Hungary

Presentation by

Benjamin Grosof

MIT Sloan School of Management

bgrosof@mit.edu http://www.mit.edu/~bgrosof/

Joint work with Ian Horrocks, Raphael Volz, and Stefan Decker

horrocks@cs.man.ac.uk volz@fzi.de stefan@isi.edu

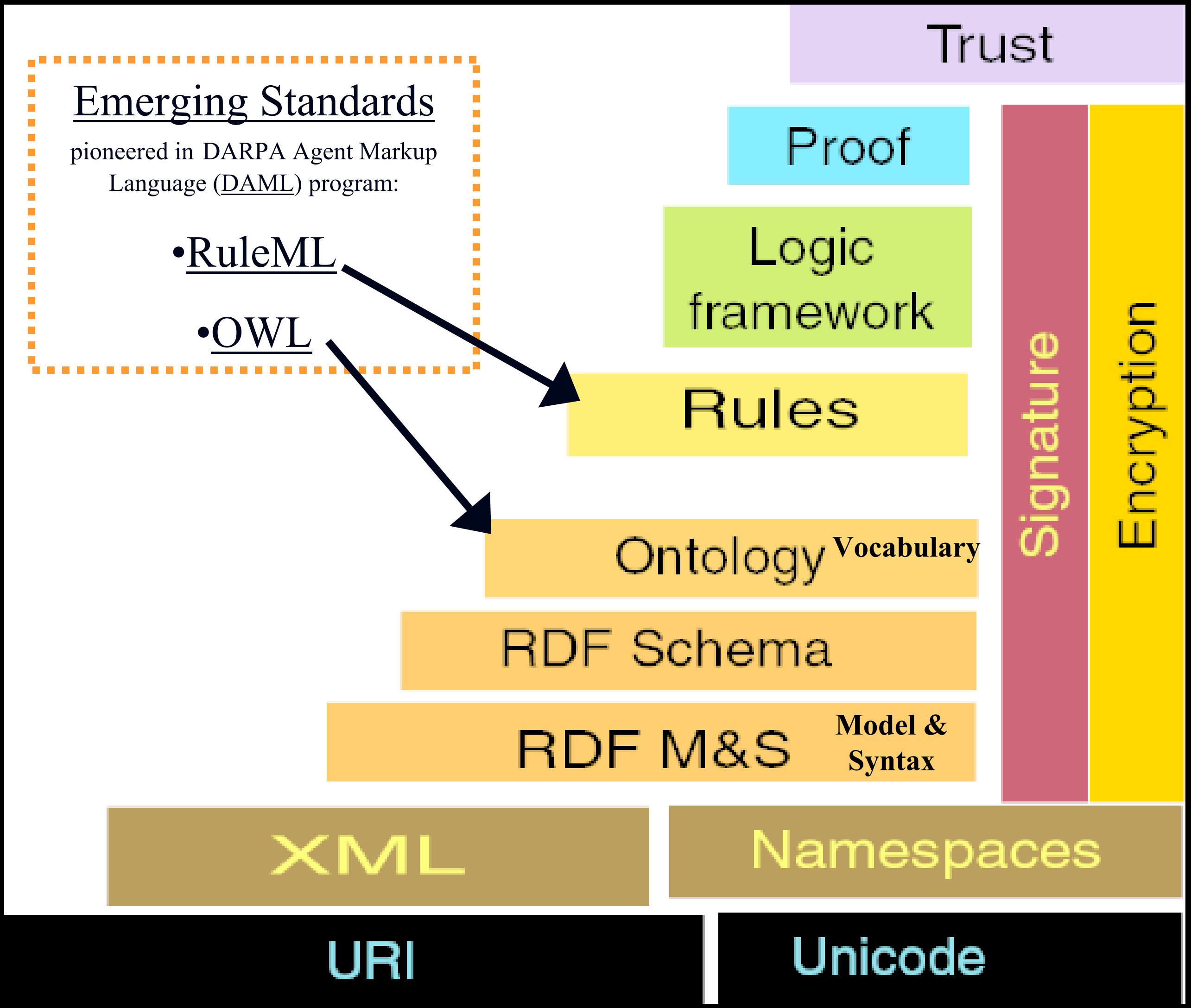
Outline/Overview

- Intro and Motivations
  - Semantic Web rules "on top of" ontologies, for Semantic Web Services
  - Need for unified semantics with completeness, consistency $\Rightarrow$ new KR Theory
- A New KR Expressive Class; Mapping between KR's
  - Define DLP $\subseteq$ LP $\cap$ DL $\Rightarrow\Rightarrow$ Enable LP $\cup$ DL
  - Detailed Mapping from DL to LP ; via Horn FOL ; invertible
  - DLP Fragment of DL is an "ontology sub-language" of LP
  - Expressive features completely captured: RDF-Schema plus much more
- Technical Capabilities and Task Scenarios Enabled
  - Primary and secondary Goals achieved for large expressive class
  - Bi-directionality enables efficiency & options in inferencing & authoring
- More Details on the mapping; Examples
- Conclusions, Related Work, Current/Future Directions

Semantic Web: concept, approach, pieces

- Shared semantics when interchange data ∴ knowledge
- Knowledge Representation (cf. AI, DB) as approach to semantics
  - Standardize KR syntax, with KR theory/techniques as backing
- Web-exposed Databases: SQL; XQuery (XML-data DB's)
  - Challenge: share DB schemas via meta-data
- RDF: "Resource Description Framework" W3C proposed standard
  - Meta-data lower-level mechanics: unordered directed graphs (vs. ordered trees)
  - RDF-Schema extension: simple class/property hierarchy, domains/ranges
- Ontology = formally defined vocabulary & class hierarchy
  - OWL: "Ontologies Working Language" W3C proposed standard
    - Subsumes RDF-Schema and Entity-Relationship models
    - Based on Description Logic (DL) KR ~subset of First-Order Logic (FOL))
- Rules = if-then logical implications, facts ~subsumes SQL DB's
  - RuleML: "Rule Markup Language" emerging standard
    - Based on Logic Programs (LP) KR ~extension of Horn FOL

W3C Semantic Web "Stack": Standardization Steps



[Diagram http://www.w3.org/DesignIssues/diagrams/sw-stack-2002.png is courtesy Tim Berners-Lee]

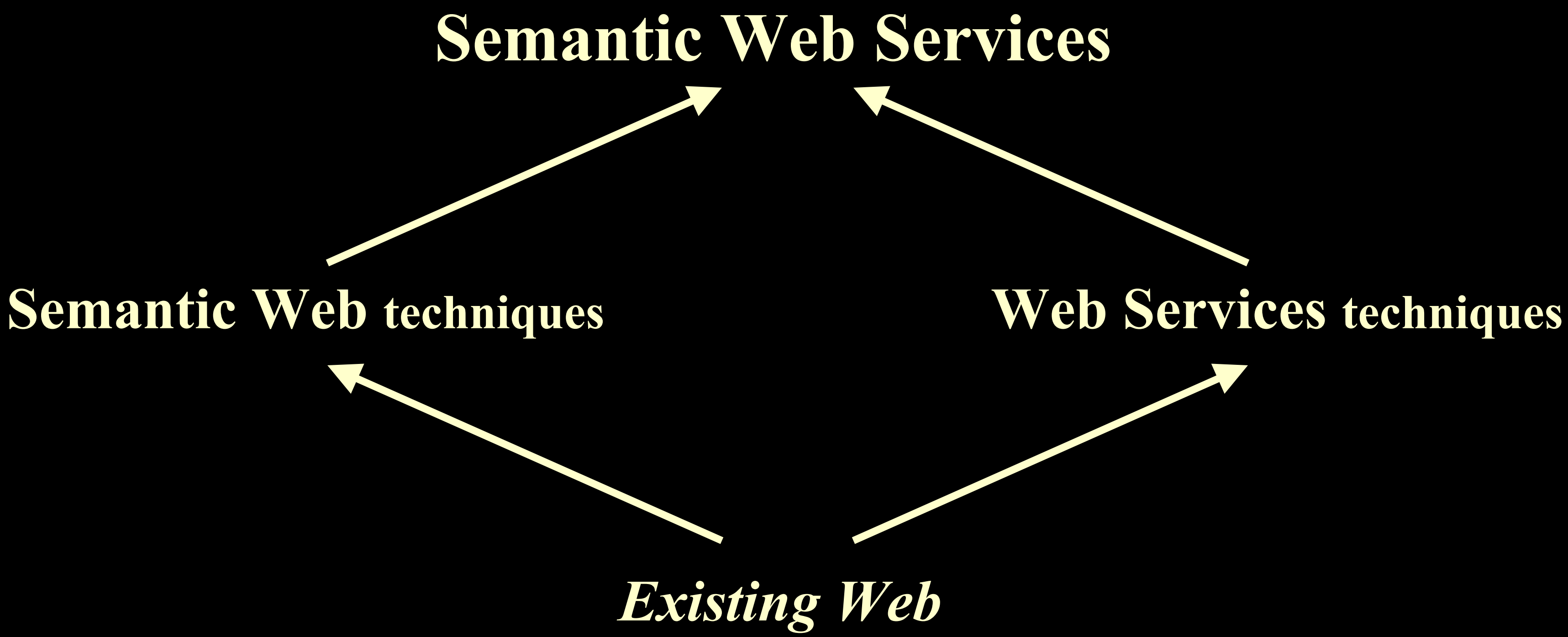
Goal: Hybridize KR's for Rules & Ontologies

- Goal: hybridize two important knowledge representations (KR's):
- 1. Description Logic (DL) ontologies cf. OWL
- 2. Logic Program (LP) rules cf. RuleML

- Primary Task Requirement identified in Semantic Web generally, e.g., by RuleML, DAML, W3C efforts:
  - LP rules use DL ontologies: rules "on top of" ontologies
    - Rules mention predicates defined in the DL ontology KB
    - Rules mention individuals that are DL ontology instances

- Secondary task objective identified in DAML:
  - Extend DL with extra LP expressiveness, to define ontologies

Next Generation Web



Application Scenarios for Rule-based Semantic Web Services

- SweetDeal [Grosof & Poon WWW-2003] configurable reusable e-contracts:
  - LP rules about agent contracts with exception handling
  - … on top of DL ontologies about business processes;
  - *a scenario motivating DLP*

- Other:
  - Trust management / authorization (Delegation Logic) [Li, Grosof, & Feigenbaum 2000]
  - Financial knowledge integration (ECOIN) [Firat, Madnick, & Grosof 2002]
  - Privacy policies (P3P APPEL)
  - Business policies, more generally

Challenges in combining LP rules with DL ontologies for SW

- What Logical KR for combining LP with DL? , with:
  - Power in inferencing? Completeness?
  - Consistency? (needs Completeness/Power)
  - Scaleability in inferencing? Tractability?
  - … Tools? Familiarity by developers for specification?
- Requirement: rules on top of ontologies
- Objective: specify ontologies via rules
- Requirement: scaleability wrt |rules|, |ontologies|

Candidate: First Order Logic

- FOL has practical and expressive drawbacks for union of DL and Rules:
  - Undecidable/Intractable
  - Lacks non-monotonicity and procedural attachments
  - Unfamiliar to mainstream software engineers

Outline/Overview

- Intro and Motivations
  - Semantic Web rules "on top of" ontologies, for Semantic Web Services
  - Need for unified semantics with completeness, consistency $\Rightarrow$ new KR Theory
- A New KR Expressive Class; Mapping between KR's
  - Define DLP $\subseteq$ LP $\cap$ DL $\Rightarrow\Rightarrow$ Enable LP $\cup$ DL
  - Detailed Mapping from DL to LP ; via Horn FOL ; invertible
  - DLP Fragment of DL is an "ontology sub-language" of LP
  - Expressive features completely captured: RDF-Schema plus much more
- Technical Capabilities and Task Scenarios Enabled
  - Primary and secondary Goals achieved for large expressive class
  - Bi-directionality enables efficiency & options in inferencing & authoring
- More Details on the mapping; Examples
- Conclusions, Related Work, Current/Future Directions

Enter… Description Logic Programs (DLP)

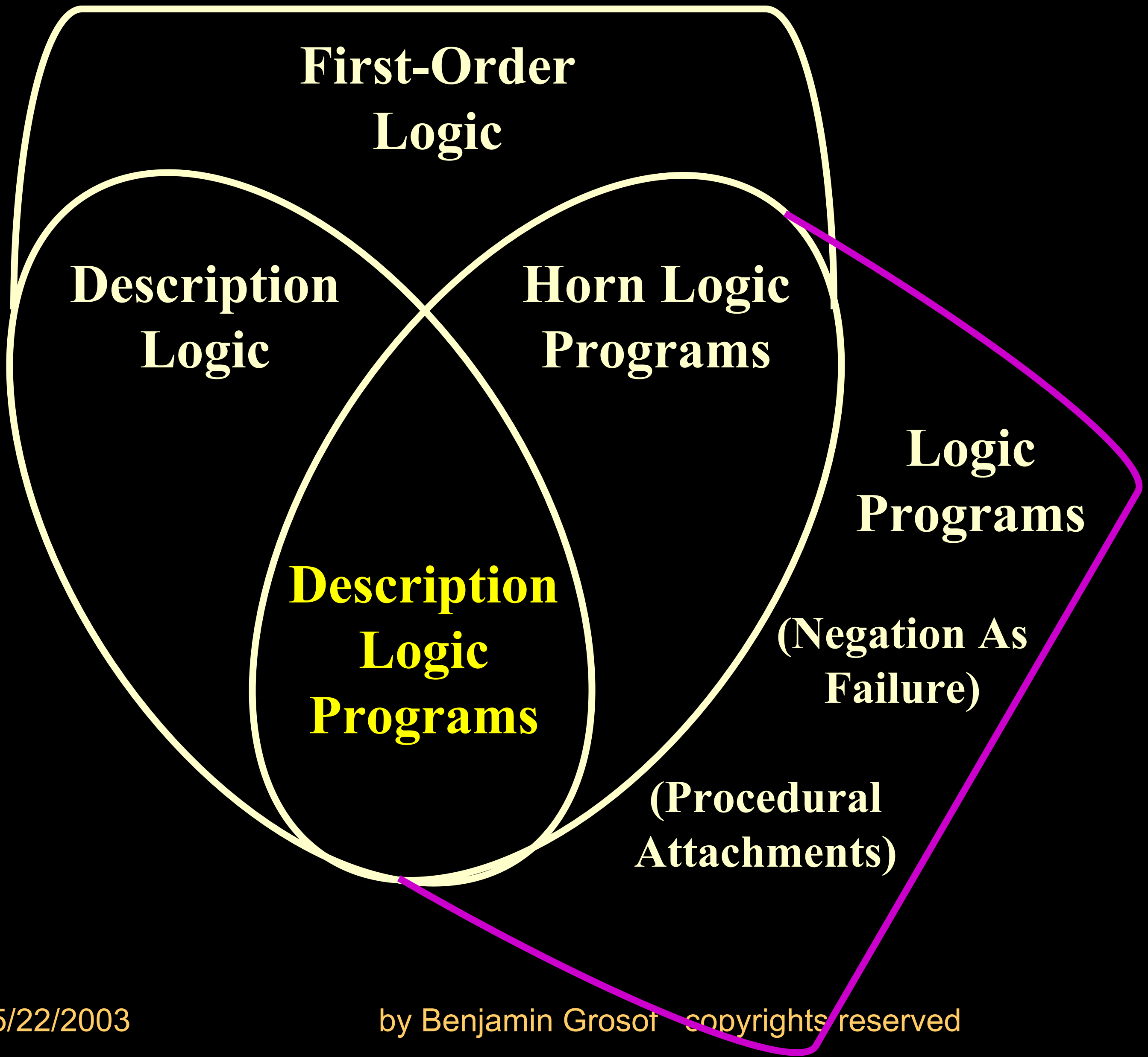
Goal: understand relationship between DL and LP/HornFOL as KR's

Insight: the expressive *intersection* is also
a key to the expressive *combination/union*

Analyze this intersection: define DLP

Enable "DLP-Fusion" as approach:
use DLP as bridge to combine knowledge from DL with knowledge from LP

Venn Diagram: Expressive Overlaps among KR's



 by Benjamin Grosof

LP as a superset of DLP

- “Full” LP, including with non-monotonicity and procedural attachments, can thus be viewed as including an “ontology sub-language”, namely the DLP subset of DL.

Overview of DLP KR Features

- DLP captures completely a subset of DL, comprising RDFS & more
- RDFS subset of DL permits the following statements:
  - Subclass, Domain, Range, Subproperty (also SameClass, SameProperty)
  - instance of class, instance of property

- DLP also completely captures more DL statements beyond RDFS:
  - Using Intersection connective (conjunction) in class descriptions
  - Stating that a property (or inverse) is Transitive or Symmetric
  - Using Disjunction or Existential in a *subclass* expression
  - Using Universal in a *superclass* expression

  - ∴ "OWL Feather" – subset of OWL Lite

Overview of DLP KR Features, Continued

- DLP can *largely but partially* capture: most other DL features:
  - Cardinality, functionality of property (or inverse), existential in superclass, universal in subclass.
  - But NOT: (general) negation, disjunction in superclass

- Map also to Relational DBMS (SQL) – which is LP-based.

- *Current Work:* Extend mapping (and inferencing power) via explicit equality, skolemization, integrity constraints.
  - Explicit equality for: cardinality, functionality
  - Skolemization for: existential in superclass, universal in subclass, cardinality
  - Integrity constraints for: negation

More about the Mapping between DL and LP

- Translation simpler to define from DL ⇒ LP than DL ⇐ LP.

- Translation is actually via Description Horn Logic (DHL), a subset of Datalog Horn FOL (and of DL) (Datalog = no logical functions of arity > 0)
  - Horn LP is a "*f*-weakening" of Horn FOL wrt power in inferencing
    - Conclude only ground facts (– or what's reducible to that).
  - DLP (subset of Horn LP) similarly is f-weakening of DHL
  - Then show formally that DLP is adequate for various DL / LP inferencing tasks that are of most common practical interest
    - (just as Horn LP is adequate wrt most practical inferencing tasks in Horn FOL)
    - Via expressive reduction of various inferencing tasks to other inferencing tasks
  - Additional restriction: equality-free (relaxed in Current Work)

Technical Capabilities Enabled by DLP

- LP rules "on top of" DL ontologies.
  - E.g., LP imports DLP ontologies, with completeness & consistency
  - Consistency via completeness and use of Courteous LP
- Translation of LP rules to/from DL ontologies.
  - E.g., develop ontologies in LP (or rules in DL)
- Use of efficient LP rule/DBMS engines for DL fragment.
  - E.g., run larger-scale ontologies
  - $\Rightarrow$ Exploit: Scaleability of LP/DB engines >> DL engines , as |instances| $\uparrow$.

- Translation of LP conclusions to DL.
- Translation of DL conclusions to LP.

- Facilitate rule-based mapping between ontologies / "contexts"

Outline/Overview

- Intro and Motivations
  - Semantic Web rules "on top of" ontologies, for Semantic Web Services
  - Need for unified semantics with completeness, consistency $\Rightarrow$ new KR Theory
- A New KR Expressive Class; Mapping between KR's
  - Define DLP $\subseteq$ LP $\cap$ DL $\Rightarrow\Rightarrow$ Enable LP $\cup$ DL
  - Detailed Mapping from DL to LP ; via Horn FOL ; invertible
  - DLP Fragment of DL is an "ontology sub-language" of LP
  - Expressive features completely captured: RDF-Schema plus much more
- Technical Capabilities and Task Scenarios Enabled
  - Primary and secondary Goals achieved for large expressive class
  - Bi-directionality enables efficiency & options in inferencing & authoring
- More Details on the mapping; Examples
- Conclusions, Related Work, Current/Future Directions

Simple Examples of the Mapping from DL to LP

- Simple: (are in RDF-Schema subset):
  - dog *is a subclass of* animal:
    - DL: dog $\subseteq$ animal $\Leftrightarrow$ LP: animal(?x) $\leftarrow$ dog(?x)
  - *Domain of* hasBitten *is* animal:
    - DL: Top $\subseteq$ hasBitten.animal
    - $\Leftrightarrow$ LP: animal(?x) $\leftarrow$ hasBitten(?x,?y)

More Complex Example of the Mapping from DL to LP

- More complex: (beyond RDF-Schema subset):
  - DL: $(pet \cap ((dog \cap \exists hasBitten.person) \cup (feline \cap large)))$
    $\subseteq ((dangerous \cap animal) \cap (\forall keeper.careful))$
  - $\Leftrightarrow$ LP: $dangerous(?x) \wedge animal(?x)$
    $\leftarrow pet(?x) \wedge$
    $(\quad (dog(?x) \wedge hasBitten(?x,?y) \wedge person(?y))$
    $\vee (feline(?x) \wedge large(?x)) \quad)$;
  - $careful(?z)$
    $\leftarrow pet(?x) \wedge keeper(?x,?z) \wedge$
    $(\quad (dog(?x) \wedge hasBitten(?x,?y) \wedge person(?y))$
    $\vee (feline(?x) \wedge large(?x)) \quad)$

Outline/Overview

- Intro and Motivations
  - Semantic Web rules "on top of" ontologies, for Semantic Web Services
  - Need for unified semantics with completeness, consistency $\Rightarrow$ new KR Theory
- A New KR Expressive Class; Mapping between KR's
  - Define DLP $\subseteq$ LP $\cap$ DL $\Rightarrow\Rightarrow$ Enable LP $\cup$ DL
  - Detailed Mapping from DL to LP ; via Horn FOL ; invertible
  - DLP Fragment of DL is an "ontology sub-language" of LP
  - Expressive features completely captured: RDF-Schema plus much more
- Technical Capabilities and Task Scenarios Enabled
  - Primary and secondary Goals achieved for large expressive class
  - Bi-directionality enables efficiency & options in inferencing & authoring
- More Details on the mapping; Examples
- Conclusions, Related Work, Current/Future Directions

Related Work to DLP

- CARIN [Halevy & Rousset 1998] on extending DL with some aspects of LP. Focus is on querying DL style KBs.
- [Antoniou 2002] on Defeasible Logic rules + Description Logic (variant) ontologies

Current Work / Future Directions

- Implementation: prototype is running, soon to be public
  - SweetOnto (formerly "Bubo") [Motik, Volz, Grosof, Horrocks, & *et al*]
- Extend mapping (and inferencing power) via: [Grosof, Horrocks, Decker, Volz, Motik, & *et al*]
  - Explicit equality for: cardinality, functionality
  - Skolemization for: existential in superclass, universal in subclass, cardinality
  - Integrity constraints for: negation
- More KR Theory, e.g., Algorithms, Complexity [Grosof, Horrocks,& et al]
- Application scenarios / use cases, e.g., Semantic Web Services [panel 5/23 2pm]
  - E.g., SweetDeal e-contracting [Grosof & Poon, WWW-2003 (5/22 10am)]
  - E.g., running DL via LP/RDBMS engines [Volz, Motik, Horrocks, & Grosof]
- Consider LP with additional features, exploit in LP and in DL: [Grosof & *et al*]
  - Courteous LP for Conflict handling of inconsistencies arising during merging
  - Situated LP for Built-ins: e.g., arithmetic or string operations

OPTIONAL SLIDES FOLLOW

Examples of DL beyond DLP

- DLP is a *strict* subset of DL.
- Examples of DL that is not (completely) representable in DLP:
  - 1. State a subclass of a complex class expression which is a disjunction. E.g.,
    - $(\text{Human} \cap \text{Adult}) \subseteq (\text{Man} \cup \text{Woman})$
  - 2. State a subclass of a complex class expression which is an existential. E.g.,
    - $\text{Radio} \subseteq \exists\ \text{hasPart.Tuner}$
- Why not? Because: LP/Horn, and thus DLP, cannot represent a disjunction or existential in the head.

- (Can partially represent head existential (e.g., (2.)) via skolemizing.)

Examples of LP beyond DLP

- DLP is a *strict* subset of Datalog Horn LP.
- Examples of Datalog Horn LP that are not (completely) representable in DLP:
  - A rule involving (unrestricted appearance of) multiple variables. E.g.,
    - PotentialLoveInterestBetween(?X,?Y)
      ← Man(?X) /\ Woman(?Y).
  - Chaining (besides simple transitivity) to derive values of Properties. E.g.,
    - InvolvedIn(?Company, ?Industry)
      ← Subsidiary(?Company, ?Unit)
      /\ AreaOf(?Unit, ?Industry).
- Why not? Essentially because: DL cannot represent "more than one free variable at a time".