OWL, DL and rules

Based on slides from Grigoris Antoniou, Frank van Harmele and Vassilis Papataxiarhis
OWL and Rules

- Rule based systems are an important and useful way to represent and reason with knowledge.
- Adding rules to OWL has proved to be fraught with problems.
- We’ll look at the underlying issues and two approaches:
  - SWRL: a failed standard that is none the less widely used.
  - RIF: a successful standard that’s not yet widely used.
The Semantic Web is grounded in logic

But what logic?

- OWL Full = Classical first order logic (FOL)
- OWL-DL = Description logic
- N3 rules \(\sim=\) logic programming (LP) rules
- SWRL \(\sim=\) DL + LP
- Other choices are possible, e.g., default logic, fuzzy logic, probabilistic logics, ...

How do these fit together and what are the consequences
We need both structure and rules

- **OWL’s ontologies** based on DL (and thus in FOL)
  - The Web is an open environment
  - Reusability / interoperability
  - An ontology is a model easy to understand

- Many **rule systems** based on logic programming
  - To achieve decidability, ontology languages don’t offer the expressiveness we want. Rules do it well
  - Efficient reasoning support already exists
  - Rules are well-known and often more intuitive
A common approach

- **Rules Layer**
  - SWRL
  - High Expressiveness

- **Ontology Layer**
  - OWL-DL
  - Conceptualization of the domain

- **Signature**
  - URI/IRI
  - Unicode
  - Namespaces
  - XML Schema
  - XML Query
  - RDF Model & Syntax
  - Ontology
  - Rules/Query
  - Logic
  - Proof
  - Trusted SW
Description Logics vs. Horn Logic

- Neither is a subset of the other
- Impossible to say that people who study & live in same city are local students in OWL, but can be done using a rule
  \[ \text{studies}(X,Y), \text{lives}(X,Z), \text{loc}(Y,U), \text{loc}(Z,U) \rightarrow \text{localStud}(X) \]
- Rules cannot assert the information that every person is either a man or a woman
  - This is easily expressed in OWL using disjoint union
Basic Difficulties

Classical logic and logic programming differ along several key issues

- **Monotonic vs. non-monotonic** features
  - Open-world vs. Closed-world assumption
  - Negation-as-failure vs. classical negation

- Non-ground entailment

- **Strong negation** vs. classical negation

- Equality

- Decidability
What’s Horn clause logic

- Prolog and most ‘logic’-oriented rule languages use horn clause logic
  - Defined by UCLA mathematician Alfred Horn

- Horn clauses are a subset of FOL where every sentence is a disjunction of literals (atoms) where at most one is positive
  \[ \sim P \lor \sim Q \lor \sim R \lor S \]
  \[ \sim P \lor \sim Q \lor \sim R \]

- Atoms are propositional variables (isRaining) or predicates (married(alice, ?x))
An alternate formulation

- Horn clauses can be re-written using the implication operator
  \[ \sim P \lor Q = P \rightarrow Q \]
  \[ \sim P \lor \sim Q \lor R = P \land Q \rightarrow R \]
  \[ \sim P \lor \sim Q = P \land Q \rightarrow \]

- What we end up with is \sim “pure prolog”
  - Single positive atom as the rule conclusion
  - Conjunction of positive atoms as the rule antecedents (conditions)
  - No not operator
  - Atoms can be predicates (e.g., mother(X,Y))
Where are the quantifiers?

- Quantifiers (forall, exists) are implicit
  - Variables in *rule head* (i.e., conclusion or consequent) are universally quantified
  - Variables only *in rule body* (i.e., condition or antecedent) are existentially quantified

Example:
- isParent(X) ← hasChild(X,Y)
- forall X: isParent(X) if Exisits Y: hasChild(X,Y)
We can relax this a bit

- Head can contain a conjunction of atoms
  - $P \land Q \leftarrow R$ is equivalent to $P \leftarrow R$ and $Q \leftarrow R$

- Body can have disjunctions
  - $P \leftarrow R \lor Q$ is equivalent to $P \leftarrow R$ and $P \leftarrow Q$

- But something are just not allowed:
  - No disjunction in head
  - No negation operator, i.e. NOT
Facts & rule conclusions are definite

- Definite means *not a disjunction*
- Facts are rule with the trivial true condition
- Consider these true facts:
  - \( P \lor Q \) # either \( P \) or \( Q \) (or both) are true
  - \( P \rightarrow R \) # if \( P \) is true, then \( R \) is true
  - \( Q \rightarrow R \) # if \( Q \) is true, then \( R \) is true
- What can you conclude?
- Can this be expressed in horn logic?
Facts & rule conclusions are definite

- Consider these true facts where *not* is classical negation rather than “negation as failure”
  
  \[
  \begin{align*}
  \text{not}(P) & \Rightarrow Q, \text{not}(Q) \Rightarrow P \quad \# \text{i.e. } P \lor Q \\
  P & \Rightarrow R, Q \Rightarrow R
  \end{align*}
  \]

- A horn clause reasoner can’t prove that either P or Q is necessarily true or false so can’t show that R must be true

- Treating *not* as negation as failure yields a loop
Open- vs. closed-world assumption

- **Logic Programming – CWA**
  - If $KB \not\models a$, then $KB = KB \cup \neg a$

- **Classical Logic – OWA**
  - It keeps the world open.
  - $KB$:
    
    $\text{Man} \sqsubseteq \text{Person}, \text{Woman} \sqsubseteq \text{Person}$
    
    $\text{Bob} \in \text{Man}, \text{Mary} \in \text{Woman}$

Query: “find all individuals that are not women”
Non-ground entailment

- The LP-semantics is defined in terms of minimal Herbrand model, i.e., sets of ground facts.
- Because of this, Horn clause reasoners can not derive rules, so that can not do general subsumption reasoning.
Decidability

- The largest obstacle!
  Tradeoff between expressiveness and decidability

- Facing decidability issues from
  - In LP: Finiteness of the domain
  - In classical logic (and thus in DL): combination of constructs

- Problem:
  Combination of “simple” DLs and Horn Logic are undecidable. (Levy & Rousset, 1998)
Rules + Ontologies

- Still a challenging task!
- A number of different approaches exists:
  - SWRL, DLP (Grosof), dl-programs (Eiter), DL-safe rules, Conceptual Logic Programs (CLP), AL-Log, DL+log
- Two main strategies:
  - Tight Semantic Integration (Homogeneous Approaches)
  - Strict Semantic Separation (Hybrid Approaches)
Homogeneous Approach

• Interaction with tight semantic integration
• Both ontologies and rules are embedding in a common logical language
• No distinction between rule predicates and ontology predicates
• Rules may be used for defining classes and properties of the ontology
• Example: SWRL, DLP
Hybrid Approach

• Integration with strict semantic separation between the two layers
• Ontology used to conceptualize the domain
• Rules can’t define ontology classes & properties, but some application-specific relations
• Communication via a “safe interface”
• Example: answer set programming (ASP)
Semantic Web Rule Language

- SWRL is the union of DL and horn logic + many built-in functions (e.g., math)
- Submitted to the W3C in 2004, but failed to become a recommendation
  - W3C pursued a more general solution: RIF
- Problem: full SWRL specification leads to undecidability in reasoning
- SWRL is well specified and subsets are widely supported (e.g., in Pellet, HermiT)
SWRL

- OWL classes are unary predicates, properties are binary ones
  
  \[ \text{Person}(?p) \land \text{sibling}(?p, ?s) \land \text{Man}(?s) \rightarrow \text{brother}(?p, ?s) \]

- As in Prolog, built-ins can be booleans or do a computation and unify the result to a variable
  - `swrlb:greaterThan(?age2, ?age1)`  # age2>age1
  - `swrlb:subtract(?n1, ?n2, ?diff)`  # diff=n1-n2

- SWRL predicates for OWL axioms and data tests
  - `differentFrom(?x, ?y), \text{sameAs}(?x, ?y), \text{xsd:int}(?x), [3, 4, 5](?x), ...`
Protégé 4.x has minimal support for SWRL. Can add/edit rules, some reasoners (Pellet, HermiT) use them.

Protégé 3.x has Jess, an internal rules engine. Jess is a production rule system with a long ancestry.

And good tools for editing, managing and using rules.

See the SWRL tab.

---

**SWRL architecture for Protégé 3.x**
The Essence of SWRL

- Combines OWL DL (and thus OWL Lite) with function-free Horn logic
- Thus it allows Horn-like rules to be combined with OWL DL ontologies
Rules in SWRL

B1, . . . , Bn → A1, . . . , Am

A1, . . . , Am, B1, . . . , Bn have one of the forms:

- C(x)
- P(x, y)
- sameAs(x, y) differentFrom(x, y)

where C is an OWL description, P is an OWL property, and x, y are variables, OWL individuals or OWL data values.
Drawbacks of SWRL

- Main *source of complexity*: arbitrary OWL expressions, such as restrictions, can appear in the head or body of a rule
- Adds significant expressive power to OWL, but causes *undecidability*
  - there is no inference engine that draws exactly the same conclusions as the SWRL semantics
SWRL Sublanguages

- SWRL adds the expressivity of DLs and function-free rules
- One challenge: identify sublanguages of SWRL with right balance between expressivity and computational viability
- A candidate OWL DL + \textit{DL-safe rules}
  - every variable must appear in a non-description logic atom in the rule body
(all?) reasoners support only DL-safe rules
  - Rule variables bind only to known individuals

Example (mixing syntaxes):

:Vehicle(?v) ^ :Motor(?m) ^ :hasMotor(?v,?m) -> :MotorVehicle(?v)
:Car = :Vehicle and some hasMotor Motor
:x a :Car

The reasoner will not bind ?m to a motor since it is not a known individual
Protégé 3.x had SWRL-Tab

Good support for creating and editing SWRL rules
Protégé 4.3 bare-bones SWRL support

Simple interface to add and edit SWLR rules
SWRL limitations

SWRL rules do not support many useful features of some rule-based systems

- Default reasoning
- Rule priorities
- Negation as failure (e.g., for closed-world semantics)
- Data structures
- ...

The limitations gave rise to RIF
Horn logic is a subset of predicate logic that allows efficient reasoning, orthogonal to description logics.

Horn logic is the basis of monotonic rules.

DLP and SWRL are two important ways of combining OWL with Horn rules.
  - DLP is essentially the intersection of OWL and Horn logic.
  - SWRL is a much richer language.
Nonmonotonic rules are useful in situations where the available information is incomplete. They are rules that may be overridden by contrary evidence. Priorities are sometimes used to resolve some conflicts between rules. Representation XML-like languages is straightforward.
LP and classical logic overlap

FOL:  (All except (6)),  (2)+(3)+(4): DLs
(4): Description Logic Programs (DLP),  (3): Classical Negation
(4)+(5): Horn Logic Programs,  (4)+(5)+(6): LP
(6): Non-monotonic features (like NAF, etc.)  (7): ^head and, ∨ body