Knowledge Representation for the Semantic Web

Part II: Rules for OWL

Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph

KI 2009 Paderborn

Most recent versions of all slides available at http://semantic-web-book.org/page/KI2009

Why Rules?

OWL may not suffice for all applications

- There are statements that cannot be expressed in OWL
- Modeling constructs of OWL not always adequate or most desirable
- First-order logic in general may be insufficient (e.g. if non-monotonic negation is desired)

→ "Rules" as an alternative paradigm for knowledge modeling

What is a Rule?

- Logical Rules (predicate logic implications)
  - "F → G" (equivalent to "¬F v G")
  - Logical extension of a knowledge base (**static**)
  - **Open World**
  - Declarative (descriptive)
- Procedural Rules (e.g. production rules)
  - "If X then Y else Z"
  - Executable machine directive (**dynamic**)
  - **Operational** (meaning = effect on execution)
- Logic Programming (e.g. Prolog, F-Logic)
  - "`man(X) :- person(X), not woman(X)`"
  - Approximating logical semantics with procedural aspects, built-ins possible
  - Typically **Closed World**
  - "**semi-declarative**"
- Deduction rules of a calculus
  - Rules not part of the knowledge base, "meta-rules"

Which Rule Language?

Rule languages are hardly compatible with each other → important to chose adequate rule language

Possible criteria:

- Clear specification of syntax and semantics?
- Support by software tools?
- Which expressive features are needed?
- Complexity of implementation? Performance?
- Compatibility with other formats, e.g. OWL?
- Declarative (describing) or operational (programming)?
- …

Which Rule Language?

- **Logical Rules** (predicate logic implications)
  - Clearly defined, extensively researched, well understood
  - Very well compatible with OWL and RDF
  - Not decidable if unrestricted

- **Procedural Rules** (e.g. production rules)
  - Many independent approaches, vague definition
  - Used like programming languages, relation to RDF and OWL not clear
  - Efficient processing possible
- **Logic Programming** (e.g. Prolog, F-Logic)
  - Clearly defined, but many independent approaches
  - Partly compatible with OWL and RDF
  - Decidability/complexity depends very much on the chosen approach

→ In this tutorial: predicate logic rules
(which are also the basis for logic programming)

Predicate Logic as a Rule Language

- Rules as first-order logic implications (Horn clauses):

  $A1 \wedge A2 \wedge \ldots \wedge An \rightarrow H$ ("Body → Head")

  Example: "Man(x) ∧ happilyMarriedWith(x,y) → HappyHusband(x)"

- Constants, variables, function symbols can be used; but no negation
- Quantifiers are omitted: free variables considered universally quantified
- **Datalog:** rules without function symbols
  - Originally developed for deductive databases
  - Knowledge bases ("datalog programs") are sets of function-free Horn clauses
  - Decidable
  - efficiently implementable for large datasets (PTime data complextiy, overall complexity still ExpTime)

Semantics of Datalog?

- Semantics follows from first-order logic:
  - Generally well-known
  - Compatible with other first-order fragments, e.g. description logics (→ OWL)

SWRL

How can datalog and OWL be combined?

SWRL – Semantic Web Rule Language [swirl]

- Proposal for a rule extension for OWL (W3C member submission)
- Idea: datalog rules referring to an OWL ontology
  → Symbols in rules can be OWL identifiers or new symbols
- Various further features and syntactic forms (not relevant here)

Semantics of SWRL

Combined semantics OWL DL + datalog?
→ use first-order mapping of OWL (see Part I)

In effect:

- OWL individuals are datalog constants
- OWL classes are unary datalog predicates
- OWL properties are binary datalog predicates

→ A first-order interpretation can at the same time be a model for an OWL ontology and a datalog program
→ Entailment over OWL+datalog (SWRL) well-defined

Example: SWRL knowledge base

For readability, our datalog syntax does not follow a formal specification (SWRL XML is very verbose, based on RuleML)

$$
\begin{aligned}
&(1)\quad \mathit{Vegetarian}(x) \wedge \mathit{Fishproduct}(y) \rightarrow \mathit{dislikes}(x,y)\\
&(2)\quad \mathit{ordered}(x,y) \wedge \mathit{dislikes}(x,y) \rightarrow \mathit{Unhappy}(x)\\
&(3)\quad \mathit{ordered}(x,y) \rightarrow \mathit{Dish}(y)\\
&(4)\quad \mathit{dislikes}(x,z) \wedge \mathit{Dish}(y) \wedge \mathit{contains}(y,z) \rightarrow \mathit{dislikes}(x,y)\\
&(5)\quad \rightarrow \mathit{Vegetarian}(\mathit{markus})\\
&(6)\quad \mathit{Happy}(x) \wedge \mathit{Unhappy}(x) \rightarrow\\
&\\
&(7)\quad \exists \mathit{ordered}.\mathit{ThaiCurry}(\mathit{markus})\\
&(8)\quad \mathit{ThaiCurry} \sqsubseteq \exists \mathit{contains}.\mathit{Fishproduct}
\end{aligned}
$$

Example: SWRL knowledge base

OWL DL can be translated to first-order logic:

$$
\begin{array}{ll}
(1) & \mathtt{Vegetarian(x)} \wedge \mathtt{Fishproduct(y)} \rightarrow \mathtt{dislikes(x,y)} \\
(2) & \mathtt{ordered(x,y)} \wedge \mathtt{dislikes(x,y)} \rightarrow \mathtt{Unhappy(x)} \\
(3) & \mathtt{ordered(x,y)} \rightarrow \mathtt{Dish(y)} \\
(4) & \mathtt{dislikes(x,z)} \wedge \mathtt{Dish(y)} \wedge \mathtt{contains(y,z)} \rightarrow \mathtt{dislikes(x,y)} \\
(5) & \rightarrow \mathtt{Vegetarian(markus)} \\
(6) & \mathtt{Happy(x)} \wedge \mathtt{Unhappy(x)} \rightarrow \\
 & \\
(7) & \exists \mathtt{y.ordered(markus,y)} \wedge \mathtt{ThaiCurry(y)} \\
(8) & \forall \mathtt{x.ThaiCurry(x)} \rightarrow (\exists \mathtt{y.contains(x,y)} \wedge \mathtt{FishProduct(y)})
\end{array}
$$

→ Semantics completely defined
→ Expected conclusion: **`Unhappy(markus)`**

Note: empty rule heads correspond to “false” (rule body must never be true)
empty rule bodies correspond to “true” (rule head must always be true)

How Hard is SWRL?

- Deduction for OWL DL is NExpTime-complete
- Deduction for OWL 2 DL is N2ExpTime-complete
- Deduction in datalog is ExpTime-complete

→ How hard is deduction for SWRL?

Deduction for SWRL is undecidable

(for OWL and thus for OWL 2, even for OWL EL)

Undecidability of SWRL

SWRL is undecidable:
There is no algorithm that can draw *all* logical conclusions from *all* SWRL knowledge bases, even with unlimited time and resources.

Practically possible:

- Algorithms that draw *all* conclusions for *some* SWRL knowledge bases
- Algorithms that draw *some* conclusions from *all* SWRL knowledge bases

→ Both trivial if "some" refers to very few things

Decidable Fragments of SWRL

Which classes of SWRL knowledge bases allow for complete inference algorithms?

- All SWRL knowledge bases consisting only of OWL (2) axioms
- All SWRL knowledge bases consisting only of datalog rules
- Every fixed finite class of SWRL knowledge bases

→ Which more interesting decidable fragments exist?

- **Description Logic Rules**
- **DL-safe Rules**

Description Logic Rules

Observation:
Some SWRL-rules can already be expressed in OWL 2.

- Identifying all such Description Logic Rules leads to a decidable fragment
- Goal: Exploit "hidden" expressivity of OWL 2
- Implementation directly by OWL 2 tools

Simple Rules in OWL 2: Examples

- Simple OWL 2 axioms correspond to rules:

$$\begin{aligned} \mathtt{Class1} &\sqsubseteq \mathtt{Class2} \\ \mathtt{Property1} &\sqsubseteq \mathtt{Property2} \end{aligned}$$

correspond to

$$\begin{aligned} \mathtt{Class1}(x) &\rightarrow \mathtt{Class2}(x) \\ \mathtt{Property1}(x,y) &\rightarrow \mathtt{Property2}(x,y) \end{aligned}$$

Simple Rules in OWL 2: Examples

- Some classes can be decomposed in rules:

$$Happy \sqcap Unhappy \sqsubseteq \bot$$

$$\exists livesIn.\exists locatedIn.EUCountry \sqsubseteq EUCitizen$$

correspond to

$$Happy(x) \wedge Unhappy(x) \rightarrow$$
$$livesIn(x,y) \wedge locatedIn(y,z) \wedge EUCountry(z) \rightarrow EUCitizen(x)$$

Simple Rules in OWL 2: Examples

- Property chains provide further rule-like axioms:

$$hasParent \circ hasBrother \sqsubseteq hasUncle$$

correspond to

$$hasParent(x,y) \wedge hasBrother(y,z) \rightarrow hasUncle(x,z)$$

→ In all examples, mapping can also be inverted (expressing rules in OWL 2)
→ Axioms also available in OWL 2 RL (see Part I of this tutorial)

More Rules (1)

What about the following?

$$dislikes(x,z) \wedge Dish(y) \wedge contains(y,z) \rightarrow dislikes(x,y)$$

- Rule head with two variables → not representable by subclass axiom
- Rule body contains class expression → not representable by subproperty axiom
- Not available in OWL 2 RL

Yet, this rule can be encoded using OWL 2!

More Rules (2)

Simpler example:

$$Man(x) \wedge hasChild(x,y) \rightarrow fatherOf(x,y)$$

Idea: replace $Man(x)$ by a property expression to encode rule as property chain

- Self can be used to transform classes to properties:
  - Auxiliary property P_{Man}
  - Auxiliary axiom:

$$Man \equiv \exists P_{Man}.Self$$

  - "Men are exactly those things that have an P_{Man} relation to themselves."

We can now encode the rule as follows:

$$P_{Man} \circ hasChild \sqsubseteq fatherOf$$

More Rules (3)

$$dislikes(x,z) \wedge Dish(y) \wedge contains(y,z) \rightarrow dislikes(x,y)$$

becomes

$$Dish \equiv \exists P_{Dish}.Self$$

$$dislikes \circ contains^{-} \circ P_{Dish} \sqsubseteq dislikes$$

(Note the inverse role $contains^{-}$)

More Rules (4)

Not so simple:

$$Vegetarian(x) \wedge Fishproduct(y) \rightarrow dislikes(x,y)$$

Idea: use universal role $\mathbf{U}$ for linking unconnected rule body parts

- Self can be used to transform classes to properties:
  - Auxiliary properties $\mathbf{P_{Vegetarian}}$ and $\mathbf{P_{Fishproduct}}$
  - Axioms:

$$Vegetarian \equiv \exists P_{Vegetarian}.Self$$
$$Fishproduct \equiv \exists P_{Fishproduct}.Self$$
$$P_{Vegetarian} \circ U \circ P_{Fishproduct} \sqsubseteq dislikes$$

Limits of Description Logic Rules

Not all rules can be encoded like this! Example:

$$\mathit{ordered}(x,y) \wedge \mathit{dislikes}(x,y) \rightarrow \mathit{Unhappy}(x)$$

Overview of possible transformations:

- Inverting properties
- "Rolling-up" side branches, e.g.

$$\mathit{locatedIn}(x,y) \wedge \mathit{EUCountry}(y) \qquad \text{becomes}$$

$$\exists \mathit{locatedIn}.\mathit{EUCountry}(x)$$

- Replacing concepts by properties (using **hasSelf**)
- Turn property conjunctions into chains

Defining Description Logic Rules

- Preparation: **Normalise rule**
  - For each occurrence (!) of an individual name ***n*** in the rule: Use a fresh variable **x**, add $\{n\}(x)$ to rule body, and replace the occurrence of ***n*** by **x**.
  - Replace every atom $P(x,x)$ by $\exists P.Self(x)$.
- **Dependency graph of a rule:** undirected graph where
  - Nodes = variables of a rule
  - Edges = property atoms of a rule (direction is ignored)
- **A SWRL rule is a Description Logic Rule if:**
  - All rule atoms use OWL 2 class and property names only
  - The normalised rule's dependency graph has no cycles

Example

(1) $\text{Vegetarian}(x) \wedge \text{Fishproduct}(y) \rightarrow \text{dislikes}(x,y)$

(2) $\text{ordered}(x,y) \wedge \text{dislikes}(x,y) \rightarrow \text{Unhappy}(x)$

(3) $\text{ordered}(x,y) \rightarrow \text{Dish}(y)$

(4) $\text{dislikes}(x,z) \wedge \text{Dish}(y) \wedge \text{contains}(y,z) \rightarrow \text{dislikes}(x,y)$

(5) $\rightarrow \text{Vegetarian}(\text{markus})$

(6) $\text{Happy}(x) \wedge \text{Unhappy}(x) \rightarrow$

Note: Restrictions like regularity and simplicity must still be satisfied after the translations.

Transforming DL Rules to OWL 2

Input: A Description Logic Rule

(1) Normalise the rule.

(2) For every variable **z** in the head: if **z** is not in body, add $\top$**(z)** to body.

(3) For every pair of variables **x** and **y**:
If **x** is not reachable from **y** in the dependency graph, insert **U(x,y)** into the rule

(4) The rule head must have a form **D(z)** or **S(z,z')**.
For every atom **R(x,y)** in the rule body:
If the path from **z** to **y** is shorter than the path from **z** to **x** in the dependency graph, then replace **R(x,y)** with $\mathbf{R^-(y,x)}$.

(5) While the rule body contains an atom **R(x,y)** such that y does not occur in any other binary predicate of the rule *do*:

- If the body contains n unary atoms **C1(y)**, …, **Cn(y)** then let **E** denote the expression (**C1** $\sqcap$ … $\sqcap$ **Cn**) and remove **C1(y)**, …, **Cn(y)** from the body. Otherwise define **E** to be $\top$.
- Replace **R(x,y)** by: $\exists$**R.E(x)**

Transforming DL Rules to OWL 2

The rule can now be expressed in OWL 2:

- If the rule head is unary, then the rule has the form $C1(x) \wedge \ldots \wedge Cn(x) \rightarrow D(x)$ then replace it by

$$C1 \sqcap \ldots \sqcap Cn \sqsubseteq D$$

- If the rule head is binary
  - For every unary atom $C(z)$ in the rule body:
    Create a new axiom

$$C \equiv \exists R.Self$$

    and replace $C(z)$ by $R(z,z)$, where R is a fresh property.
  - Rule now has the form $R1(x,x2) \wedge \ldots \wedge Rn(xn,y) \rightarrow S(x,y)$. Replace it by:

$$R1 \circ \ldots \circ Rn \sqsubseteq S$$

Notes on the Transformation

- Replacing a SWRL rule in a SWRL knowledge base by the resulting set of OWL 2 axioms does not affect satisfiability (introduced auxiliary symbols do not occur elsewhere, of course)
- The given algorithm is not optimised: it may produce overly complex axioms in some cases

DL-safe Rules

Observation: Datalog is decidable since rules can be applied in only finitely many ways: variables represent only constants.

- Variables in SWRL might represent arbitrarily many infered individuals
- Goal: Make rules "safe" by restricting possible variable assignments
- **DL-safe rules** as another decidable fragment of SWRL

DL-safe Rules: Definition

Rules now may also include non-OWL predicates:

- A **datalog atom** is an atom with a predicate symbol that does not occur as a class or property in any OWL axiom.

A SWRL rule is **DL-safe** if:

- Every variable in the rule head occurs in a datalog atom in the body.

→ Only constant symbols relevant when considering variable assignments in datalog atoms.

A Note on DL-Safety

(Extra slide for offline use)

The previous definition "hides" a recursive condition: in a DL-safe knowledge base, datalog atoms cannot occur in the head of any non-DL-safe rule. Since we do not consider rules that are not DL-safe, we thus do not explicitly require datalog atoms to not occur in the head of unsafe rules. If DL-safe rules were used in combination with other rules, then one either needs this extended condition, or one needs to consider the other rules as part of the OWL knowledge base (this is what we do for Description Logic Rules).

Enforcing DL-Safety

- Example:

$$\mathit{ordered}(x,y) \wedge \mathit{dislikes}(x,y) \rightarrow \mathit{Unhappy}(x)$$

→ not DL-safe if ***ordered*** or ***dislikes*** occur in OWL axioms

- Enforcing DL-safety by restricting rules to **named** individuals:

$$\mathit{ordered}(x,y) \wedge \mathit{dislikes}(x,y) \wedge O(x) \wedge O(y) \rightarrow \mathit{Unhappy}(x)$$

where a fact → $O(a)$ is added for all individuals ***a***.

→ Rule only applicable to **named** OWL individuals

DL-Safe Rules in Practice

- OWL 2 with DL-safe rules is decidable
- Naïve implementation: each rule expressible by finitely many DL rules where all variables are replaced by individual symbols in all possible ways (very inefficient)
- No increase in worst-case complexity
- OWL 2 keys are a special form of DL-safe rules:

$$motherOf(x1,y) \wedge motherOf(x2,y) \wedge O(x1) \wedge O(x2) \wedge O(y) \rightarrow x1=x2$$

Implementations:

- Basic support in some reasoners (KAON2, Pellet)
- Support for keys in OWL 2 tools (e.g. HermiT)

A Combined Example

OWL 2 + Description Logic Rules + DL-safe rules still decidable:

```
(1)  Vegetarian(x) ∧ Fishproduct(y)   → dislikes(x,y)
(2)ordered(x,y) ∧ dislikes(x,y) ∧ O(x) ∧ O(y)  → Unhappy(x)
(3)                      ordered(x,y)  → Dish(y)
(4) dislikes(x,z) ∧ Dish(y) ∧ contains(y,z)   → dislikes(x,y)
(5)                                    → Vegetarian(markus)
(6)            Happy(x) ∧ Unhappy(x)   →
(7) ∃ordered.ThaiCurry(markus)
(8) ThaiCurry  ⊑  ∃contains.Fishproduct
(9) → O(markus)
```

→ We cannot conclude **Unhappy(markus)**

A Combined Example

Explicitly use named individual:

```
(1)  Vegetarian(x) ∧ Fishproduct(y)   → dislikes(x,y)
(2)ordered(x,y) ∧ dislikes(x,y) ∧ O(x) ∧ O(y) → Unhappy(x)
(3)                      ordered(x,y)  → Dish(y)
(4) dislikes(x,z) ∧ Dish(y) ∧ contains(y,z)   → dislikes(x,y)
(5)                                    → Vegetarian(markus)
(6)            Happy(x) ∧ Unhappy(x)   →
(7) ordered(markus,redThaiCurry)
    ThaiCurry(redThaiCurry)
(8) ThaiCurry  ⊑  ∃contains.Fishproduct
(9) → O(markus)          → O(redThaiCurry)
```

→ Now we can conclude **Unhappy(markus)**

Summary: Rules

- SWRL ("OWL+ datalog") is undecidable
- Description Logic Rules:
  - SWRL fragment expressible in OWL 2
  - Supported indirectly by OWL 2 reasoners
  - Definition and translation based on dependency graph
- DL-safe rules:
  - SWRL fragment where variables can only assume concrete values
  - Support by some OWL reasoners
  - DL-safety can be enforced (also done implicitly in some tools)
- Combination OWL 2 + DL Rules + DL-safe rules possible

Rules for the Semantic Web?

- Standards and best-practices for rules still missing
- SWRL syntax (in XML) most widely used in applications
- W3C RIF (Rule Interchange Format) Working Group
  - Standard for various rule languages, also SWRL-like rules
  - Various new features, e.g. syntax from Frame Logic
  - Official specification expected by end 2009
- Many studies on interfacing Logic Programming and OWL
- OWL 2 RL: a profile that can be translated to datalog rules
  (note: inverse direction of Description Logic Rules)
  → enables some interoperability OWL 2 ↔ RIF
- Operational "inference rules" or "production rules" supported by some RDF-stores (e.g. Jena)

Further Reading

- P. Hitzler, S. Rudolph, M. Krötzsch: **Foundations of Semantic Web Technologies.** CRC Press, 2009. (Chapter 6 closely related to this lecture; this also contains more references on types of rules not discussed here)
- I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean. **SWRL: A Semantic Web Rule Language.** W3C Member Submission, 21 May 2004. Available at http://www.w3.org/Submission/SWRL/. (description of SWRL)
- **RIF working group homepage** (containing current status and pointers to documents): http://www.w3.org/2005/rules/wiki/RIF_Working_Group.
- S. Abiteboul, R. Hull, V. Vianu. **Foundations of Databases.** Addison Wesley, 1994. (the "Alice book" is an excellent resource on datalog)

Selected research articles:

- M. Krötzsch, S. Rudolph, P. Hitzler. **Description Logic Rules.** In Proc. 18th European Conf. on Artificial Intelligence (ECAI 2008), IOS Press, 2008. (original reference on Description Logic Rules)
- M. Krötzsch, S. Rudolph, P. Hitzler. **ELP: Tractable rules for OWL 2.** In Proc. 7th Int. Semantic Web Conf. (ISWC 2008), Springer, 2008. (extension of DL Rules to light-weight languages related to OWL EL and OWL RL)
- B. Motik, U. Sattler, R. Studer. **Query answering for OWL DL with rules.** Journal of Web Semantics, 3(1):41?60, 2005. (original reference on DL-safe rules)
- B.N. Grosof, I. Horrocks, R. Volz, S. Decker. **Description logic programs: combining logic programs with description logic.** In Proc. 12th Int. World Wide Web Conference (WWW-03), ACM, 2003. (original paper introducing DLP, a description logic that can be translated to datalog; closely related to OWL 2 RL)