Description Logic vs. Metamodelling

Dániel Varró
Foundations of Model Driven System Development
Based on contributions from
Szilvia Varró-Gyapay and the DECOS European IP

Designing Modeling Languages

• Metamodeling: Design methodology of modeling languages
• Metamodel: model of a modeling language
• Features:
  – Concrete syntax
  – Abstract syntax
  – Well-formedness rules
  – Semantics
  – Mappings to other languages
Description Logic and Reasoning

Contribution of Szilvia Varró-Gyapay

Description logic and reasoning

- Semantic web: knowledge management
  - intelligent information retrieval
  - interpretation of the information
- need for meta information and reasoning
- tools:
  - Ontology:
    hierarchical knowledge representation
  - description logic formal model for analysis
Description logics as knowledge representation

- Description of a knowledge field
  - concepts
  - relations between two concepts: roles
- Knowledge-base = TBox + ABox
  - TBox: conceptual knowledge, terminological box (axioms, concepts)
  - ABox: knowledge about the instances of a domain, data box, assertions
The Language of DL

DL Examples

Mother ⊆ Woman
Man ≡ Person ∩ ¬Woman
Woman ≡ Person ∩ Female
Mother ≡ Woman ∩ ∃hasChild.Person
Father ≡ Man ∩ ∃hasChild.Person
Parent ≡ Father ∪ Mother
MotherWithManyChildren ≡ Mother ∩ ≥3hasChild
MotherWithoutDaughter ≡ Mother ∩ ∀hasChild.¬Woman
ancestor ≡ hasChild+
### Extended SHIQ DL language

<table>
<thead>
<tr>
<th></th>
<th>DL notation</th>
<th>RACER syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>atomic concept</td>
<td>A</td>
<td>:atomic-concepts (A)</td>
</tr>
<tr>
<td>top concept</td>
<td>T</td>
<td><em>top</em></td>
</tr>
<tr>
<td>bottom concept</td>
<td>⊥</td>
<td><em>bottom</em></td>
</tr>
<tr>
<td>negation</td>
<td>¬C</td>
<td>(not C)</td>
</tr>
<tr>
<td>conjunction</td>
<td>(C_1 \cap \ldots \cap C_n)</td>
<td>(and C_1 \ldots C_n)</td>
</tr>
<tr>
<td>disjunction</td>
<td>(C_1 \cup \ldots \cup C_n)</td>
<td>(or C_1 \ldots C_n)</td>
</tr>
<tr>
<td>exists restriction</td>
<td>(\exists R.C)</td>
<td>(some R C)</td>
</tr>
<tr>
<td>value restriction</td>
<td>R.C</td>
<td>(all R C)</td>
</tr>
</tbody>
</table>

### Extended SHIQ DL language

<table>
<thead>
<tr>
<th></th>
<th>DL notation</th>
<th>RACER syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>at-least restriction</td>
<td>(\geq n R)</td>
<td>(at-least n R)</td>
</tr>
<tr>
<td>at-most restriction</td>
<td>(\leq n R)</td>
<td>(at-most n R)</td>
</tr>
<tr>
<td>exactly restriction</td>
<td>(= n R)</td>
<td>(exactly n R)</td>
</tr>
<tr>
<td>qualified at-least restriction</td>
<td>(\geq n R.C)</td>
<td>(at-least n R C)</td>
</tr>
<tr>
<td>qualified at-most restriction</td>
<td>(\leq n R.C)</td>
<td>(at-most n R C)</td>
</tr>
<tr>
<td>role name</td>
<td>R</td>
<td>:roles(R)</td>
</tr>
<tr>
<td>inverse role</td>
<td>R^-</td>
<td>(inv R)</td>
</tr>
<tr>
<td>transitive role</td>
<td>R^+</td>
<td>(R :transitive)</td>
</tr>
</tbody>
</table>
Extended SHIQ DL language

- Concrete domain concepts and attribute expressions
  - cardinal, integer, real, complex, string
  - min/max restriction over integers
  - linear polynomial equations over the reals or cardinals
  - nonlinear polynomial equations over complex numbers
  - equalities and inequalities of strings

Concept axioms and terminology

- Statements:
  - subsumption relation between two concepts
  - equivalence between two concepts
  - pairwise disjointness between several concepts
  - subsumption relation between a concept name and a concept term
  - equality between a concept name and a concept term
Role declarations

• only one declaration for each role
• features (attributes)
• transitive roles
• role hierarchy
• declaration of domain and range restrictions for roles

Reasoning on DL
Reasoning

• Input: knowledge base (TBox, ABox)
• queries
  – TBox queries: concept satisfiability, consistency, concept hierarchy
  – ABox queries: concept instances, instance retrieval, inconsistency check
• DL: decidable

Assumptions

• Open World Assumption
  – what cannot be proven to be true is not believed to be false
• Unique Name Assumption
  – all individuals used in an Abox are assumed to be mapped to different elements of the universe
RACER: A Reasoning Tool

• Renamed ABox and Concept Expression Reasoner
  – extension of the SHIQ logic
    (concrete domain concepts, attribute expressions)
  – highly optimized tableau calculus
  – built-in reasoner for several ontology management tools
  – support of general terminological axioms
  – multiple definition and cyclic definition of concepts is allowed

RACER - Application areas

• Semantic web
• E-business
• Medicine/Bioinformatics
• Natural language processing
• Knowledge engineering
• Software engineering
Most important queries
(on type level)

- concept consistency w.r.t. a Tbox
- concept subsumption w.r.t. a TBox
- determining concept hierarchy
- concept equivalence
- concept disjointness

Most important queries
(on instance level)

- consistency check of an Abox w.r.t. a TBox
- instance testing w.r.t. an Abox and a Tbox
- instance retrieval w.r.t. an Abox and a Tbox
- computation of the direct type of an instance w.r.t. an Abox and a Tbox
- concrete domain constraints check