A Solution for the Man-Man Problem in the Family History Knowledge Base

Dmitry Tsarkov, Uli Sattler, Margaret Stevens, Robert Stevens
School of Computer Science
The University of Manchester
Disclaimer:
the name “Man-Man” for this problem was coined by Michael Schneider in some discussions on public-owl-dev.

The reason for this name will become clear later, but has nothing to do with gender, men, etc.
Robert’s Family History Knowledge Base
Robert’s Family History Knowledge Base

James Alexander Bomaker and Violet Robinson
(with Robert’s grandma Iris Ellen)
Robert’s Family History Knowledge Base

William Robson and Elizabeth Frances Jessop.

Image of a photograph of two individuals, possibly William Robson and Elizabeth Frances Jessop.
Robert’s Family History Knowledge Base

- Contains facts of 400+ members of Robert’s family
  - available at www.cs.man.ac.uk/~stevensr/ontology/family.rdf.owl
- Modest-size TBox in OWL 2
  - ~ 300 axioms that use all SROIQ constructs
  - rich property hierarchy
- Large ABox with “sparse” assertions:
  - *Minimal* number of property assertions
  - No class assertions
- Goal:
  - no duplication information
  - little effort during creation
  - easy to maintain consistency KB consistent
  - rich RBox => many inferences
The TBox

• *Every brother is a man*
  
  \[ \text{Range}(\text{hasBrother}) = \text{Man} \]

• *A parent’s brother is an uncle*
  
  \[ \text{hasParent} \circ \text{hasBrother} \sqsubseteq \text{hasUncle} \]

• *A sibling-in-law is a sibling of a spouse or a spouse of a sibling*
  
  \[ \text{hasSibling} \circ \text{hasSpouse} \sqsubseteq \text{hasSiblingInLaw}, \]
  \[ \text{hasSpouse} \circ \text{hasSibling} \sqsubseteq \text{hasSiblingInLaw} \]

• *A person’s male sibling is their brother*
  
  ???
A person’s male sibling is their brother

- A class-level approach:
  
  \[ \text{hasSibling some Man} \sqsubseteq \text{hasBrother some Person} \]

- But...
  
  - a person having a male sibling is a person having a brother
    is different from
    
    a person’s male sibling is \textbf{their} brother

  - ...so this doesn’t really work
A person’s male sibling is their brother

- A rule-based approach:
  \[\text{hasSibling}(x,y), \text{Man}(y) \implies \text{hasBrother}(x,y)\]

- But...
  - General rules make OWL 2 undecidable
  - As a **DL-safe** rule:
    - does the trick on individuals:
      \[(\text{Jane, Peter}):\text{hasSibling} \text{ and Man(Peter)} \text{ entail} (\text{Jane, Peter}):\text{hasBrother}\]
    - But does not affect TBox entailments:
      e.g., O does not entail
      \[\text{isSiblingOf} \text{ some father} \sqsubseteq \text{hasBrother} \text{ some Person}\]
    - ...so this doesn’t really work
The ManMan approach [SchneiderPOD, Gasse+2008, Kroetzsch+2008]:

- Create a new property, ManMan, and add an axiom
  \[ \text{Man} \sqsubseteq \text{ManMan} \sqsubseteq \text{some Self} \]

- further add
  \[ \text{hasSibling} \sqsubseteq \text{ManMan} \sqsubseteq \text{hasBrother} \]

- works both for individuals and classes

But, to stay in OWL 2, one has to discard

- \( \text{hasBrother} \sqsubseteq \text{hasSibling} \)
  - ...so this doesn’t really work
A person’s male sibling is their brother

- Follow ManMan approach + exploit **extended regularity condition** [Kazakov, 2009]:

\[
\text{hasSibling} \circ \text{ManMan} \sqsubseteq \text{hasBrother}
\]

\[
\text{hasBrother} \sqsubseteq \text{hasSibling}
\]

- such cycles not necessarily cause undecidability
  FHKB is such an ontology

- but:
  - cycle check and
  - implementation
    require significant modification to the existing reasoners
  - No known implementation in OWL 2 reasoners yet
Our approach

- Extend OWL 2 with new property axioms
  - SpecFrom(PropA, Class) \sqsubseteq PropB
    
    ![Diagram showing SpecFrom(PropA, Class) \sqsubseteq PropB]
    
    with semantics: \( \forall x,y.\text{PropA}(x,y) \land \text{Class}(x) \Rightarrow \text{PropB}(x,y) \)
  
  - SpecInto(PropA, Class) \sqsubseteq PropB
    
    ![Diagram showing SpecInto(PropA, Class) \sqsubseteq PropB]
    
    with semantics: \( \forall x,y.\text{PropA}(x,y) \land \text{Class}(y) \Rightarrow \text{PropB}(x,y) \)
  
  - observe:
    
    SpecInto(PropA, Class) \sqsubseteq PropB is equivalent to
  
    SpecFrom(Inv(PropA), Class) \sqsubseteq Inv(PropB)
**Our approach**

- Tableau reasoner is easily extended:
  Dmitry only needed to add
  - 2 new rules to Fact++,
  - similar to existing rules

  **Spec-apply-rule:**
  - if SpecFrom(R, C) ⊑ S, x is not blocked,
    \[ R \in L(x, y), C \in L(x), S \notin L(x, y) \]
  - then add S to L(x, y)

  **Spec-choose-rule:**
  - if SpecFrom(R, C) ⊑ S, x is not blocked,
    \[ R \in L(x, y), \{C, \neg C\} \cap L(x) = \emptyset \]
  - then set \( L(x) = L(x) \cup \{D\} \), where \( D \in \{C, \neg C\} \)
Properties of our approach

• Tableaux algorithms correctly captures semantics
• Solves the problem on both individual and class level
• Pay-as-you-go:
  
  no rule specialisation axioms ➞ no overhead
• No change to existing tableaux reasoner architecture
Evaluation

- Extended OWL 2 syntax
  - with 2 new constructors
    SpecFrom(Prop,Class), SpecInto(Prop,Class)

- Implemented 2 new rules in FaCT++

- Added four new axioms to FHKB:
  - SpecFrom(isSiblingOf,Man) ⊑ isBrotherOf
  - SpecFrom(isSiblingOf,Woman) ⊑ isSisterOf
  - SpecFrom(hasParent,Man) ⊑ isSonOf
  - SpecFrom(hasParent,Woman) ⊑ isDaughterOf

- Ran some tests on FHKB+
Brother/Sister tests

- Brother1 = Man and (isSiblingOf some Person)
- Brother2 = isBrotherOf some Person
- Sister1 = Woman and (isSiblingOf some Person)
- Sister2 = isSisterOf some Person

<table>
<thead>
<tr>
<th>Class</th>
<th>Instances in FHKB</th>
<th>Instances in FHKB+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brother1</td>
<td>160</td>
<td>160</td>
</tr>
<tr>
<td>Brother2</td>
<td>152</td>
<td>160</td>
</tr>
<tr>
<td>Sister1</td>
<td>163</td>
<td>163</td>
</tr>
<tr>
<td>Sister2</td>
<td>153</td>
<td>163</td>
</tr>
</tbody>
</table>
Uncle/Aunts tests

Uncle = isBrotherOf some (hasChild some Person)
GreatUncle = isBrotherOf some (hasChild some (Person and hasChild some Person))
Aunt = isSisterOf some (hasChild some Person)
GreatAunt = isSisterOf some (hasChild some (Person and hasChild some Person))

<table>
<thead>
<tr>
<th>Class</th>
<th>Instances in FHKB</th>
<th>Instances in FHKB+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncle</td>
<td>55</td>
<td>76</td>
</tr>
<tr>
<td>GreatUncle</td>
<td>49</td>
<td>67</td>
</tr>
<tr>
<td>Aunt</td>
<td>58</td>
<td>77</td>
</tr>
<tr>
<td>GreatAunt</td>
<td>55</td>
<td>71</td>
</tr>
</tbody>
</table>
Son/Daughter tests

Son = isSonOf some Person
Daughter = isDaughterOf some Person

<table>
<thead>
<tr>
<th>Class</th>
<th>Instances in FHKB</th>
<th>Instances in FHKB+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Son</td>
<td>7</td>
<td>202</td>
</tr>
<tr>
<td>Daughter</td>
<td>6</td>
<td>208</td>
</tr>
</tbody>
</table>

• Note that
  • FHKB+ entails Son = Man
  • FHKB didn’t
BrotherInLaw tests

- BIL1 = Man and
  
  (isSiblingOf some (isSpouseOf some Person)) or
  
  (isSpouseOf some (isSiblingOf some Person))

- BIL2 = isBrotherInLawOf some Person
  
  isBrotherOf o isSpouseOf ⊑ isBrotherInLawOf
  
  isHusbandOf o isSiblingOf ⊑ isBrotherInLawOf

<table>
<thead>
<tr>
<th>Class</th>
<th>Instances in FHKB</th>
<th>Instances in FHKB+</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIL1</td>
<td>39</td>
<td>39</td>
</tr>
<tr>
<td>BIL2</td>
<td>25</td>
<td>39</td>
</tr>
<tr>
<td>SIL1</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>SIL2</td>
<td>12</td>
<td>30</td>
</tr>
</tbody>
</table>
Discussion

• FHKB is interesting test case for OWL 2
  – not the best kind of knowledge to represent in OWL,
  – but it works mostly fine
  – stress-tests reasoners: expansion is future work

• New constructors seem
  – nice:
    • help with modeling our family relations
    • we have seen various similar request & questions
  – easy-going:
    • require only simple modification of tableau reasoners
    • other reasoners should be easily extensible as well
  – not as general as extended regularity condition [Kazakov09]

• ...worth considering for OWL 3?