Decidability of $SHIQ$ with Complex Role Inclusion Axioms

Ian Horrocks and Ulrike Sattler
Computer Science Department, University of Manchester, UK

Extending $SHIQ$ with expressive means for the propagation of properties along roles, involving surprising (un)decidability results

Acapulco, August, 2003
The Description Logic \textit{SHIQ}

\textit{SHIQ} is a Description Logic which

✓ is underlying ontologies languages \textit{OIL, DAML+OIL, and OWL}
✓ is implemented in the successful DL reasoner \textit{FaCT, who}
✓ behaves well despite reasoning in \textit{SHIQ} being \textit{ExpTime-complete}
✓ extends \textit{ALC} (or multi modal \textit{K}) with

- \textit{general TBoxes} (sets of GCIs of the form $C \sqsubseteq D$)
- \textit{number restrictions} (e.g. $(\geq 4 \text{hasComp.Wheel})$ or $(\leq 4 \text{hasComp.Wheel})$)
- \textit{transitive roles} (e.g. \textit{hasPart, hasAncestor})
- \textit{inverse roles} (e.g. both \textit{hasPart and hasPart})
- \textit{role inclusions} (set of axioms of the form $r \sqsubseteq s$, e.g. \textit{hasDaughter} $\sqsubseteq \text{hasChild}$)
The Description Logic \( SHIQ \)

Example: a \( SHIQ \) TBox

\[
\text{Control-rod} = \text{Device} \sqcap \exists\text{part-of.\text{Reactor-core}}
\]

\[
\text{Reactor-core} = \text{Device} \sqcap \exists\text{has-part.\text{Control-rod}} \sqcap \exists\text{part-of.\text{N-reactor}}
\]

\[
\text{N-reactor} \sqcap \exists\text{has-part.\text{Faulty}} = \text{Dangerous}
\]

and an implied subsumption relationship:

\[
\text{Control-rod} \sqcap \text{Faulty} \text{ is subsumed by } \exists\text{part-of.Dangerous}
\]

Inference problems: satisfiability and subsumption w.r.t. a TBox and an RBox
The Tableau Algorithm for *SHIQ*

Tableau algorithm for *SHIQ* [HorrocksS_Tobies-LPAR99,HorrocksS_ECAI2002]

- decides satisfiability and subsumption of *SHIQ*-concepts w.r.t. TBoxes
- is implemented in the DL reasoner *FaCT* [Horrocks-KR98]
- tries to generate a model of input concept *C* w.r.t. TBox *T* by
  - breaking down *C* and *T* syntactically, thus inferring contraints on such a model
  - uses a special cycle detection mechanism to ensure termination
  - whose careful design is crucial for correctness of algorithm and performance of its implementation
The Tableau Algorithm for \textit{SHIQ}

More precisely: the tableau algorithm works on a tree, whose

\textbf{nodes} correspond to objects

\textbf{edges} edges indicate “direct” role-successorship, where

\textbf{implied} edges (transitivity!) have to be added in the model construction

Example: rules that are applied to the tree include

- if $C \cap D \in \mathcal{L}(x)$, then add $C$ and $D$ to $\mathcal{L}(x)$
- if $\exists r.C \in \mathcal{L}(x)$, then create a new $r$-successor $y$ of $x$ with $\mathcal{L}(y) = \{C\}$
- if $\forall r.C \in \mathcal{L}(x)$ and $y$ is an $r$-neighbour of $x$, then
  - add $C$ to $\mathcal{L}(y)$ and
  - if $r$ is transitive, then add $\forall r.C$ to $\mathcal{L}(y)$ and
  - if $r$ has a transitive sub-role $s$, then add $\forall s.C$ to $\mathcal{L}(y)$
**SHIQ and the Propagation of Properties**

Expressible in **SHIQ**:

**faultiness** propagates from a component to its aggregate

Device ⊑ (Faulty ⇒ ∀hasComp⁻.Faulty)

**colours** propagate from a segment to its aggregate

Thing ⊑ (Green ⇒ ∀hasSegment⁻.Green) □ (Red ⇒ ∀hasSegment⁻.Red) □ ...

Not expressible in **SHIQ** or other implemented DL

⇒ various questionable work-arounds:

**ownership** propagates from an aggregate to its parts, e.g.
the owner of the car is also the owner of the car’s parts

**localisation** propagates from a division to its aggregate, e.g.
a trauma located in a part of a body structure is a trauma of the body structure
Extending *SHIQ* with Complex RIAs

Solution: extend *SHIQ* with role inclusion axioms (RIAs) of the form $r \circ s \sqsubseteq t$, e.g.

- $\text{owns} \circ \text{has-part} \sqsubseteq \text{owns}$,
- $\text{hasLocation} \circ \text{divisionOf} \sqsubseteq \text{hasLocation}$

Result: known from Grammar Logic [Baldoni1998]:

*SHIQ* (or even *ALC*) with such an extension becomes undecidable

Next Solution: investigate motivating examples more closely,

- axioms of the form $r \circ s \sqsubseteq s$ or $s \circ r \sqsubseteq s$ suffice for propagation
- undecidability result from [Baldoni1998] not applicable

Next Result: *SHIQ* with such an extension is still undecidable

proof by reduction of the Domino Problem
An Undecidable Problem: the Domino Problem

Given (an unbounded amount of) instances of finitely many domino types

...
General idea: describe staircases – easy, possible in $SHIQ$
Reducing the Domino Problem to \textit{SHIQ} with Complex RIAs

**General idea:** describe \textit{staircases} – easy, possible in \textit{SHIQ}

use RIAs to \textbf{merge} staircases into \textbf{grid}
General idea: describe staircases – easy, possible in SHIQ
use RIAs to merge staircases into grid
Reducing the Domino Problem to \textit{SHIQ} with Complex RIAs

General idea: describe \textbf{staircases} – easy, possible in \textit{SHIQ}
use RIAs to \textbf{merge} staircases into \textbf{grid}
A Decidable Extension of $SHIQ$ with RIAs

Source of complexity of RIAs: inverse roles and cycles in set of RIAs

Third solution: dissallow cycles, i.e., $RIQ$ is the extensions of $SHIQ$ with finite, cycle-free set of RIAs $\mathcal{R}$, where

$\mathcal{R}$ is cycle-free if uses$^+$ is cycle-free: $s$ uses $r_i$ iff

$\text{expr}(r_1, \ldots, r_n) \subseteq s \in \mathcal{R}$ and $r_i \neq s$

In $RIQ$, each RIA can be of the form

$r_1 \cdots r_\ell s \sqsubseteq s$ or $sr_1 \cdots r_\ell \sqsubseteq s$ or $ss \sqsubseteq s$

Acyclicity is not a serious restriction:

1. motivating examples still expressible
2. some ontology experts thinks that cycles indicate modelling flaws
Tableau Algorithm for $RIQ$

Tableau algorithm for $RIQ$ similar to the one for $SHIQ$, but

1. $R$ is made explicit in a finite automaton $A_r$ for each role $r$
2. concepts $\forall r. C$ in node labels are replaced with $\forall A_r. C$
3. automata $A_r$ are then used to
   1. remember roles paths along which $\forall r. C$ was "pushed":
      
      if $y$ is an $s$-neighbour of $x$ and $\forall A. C \in \mathcal{L}(x)$, then add $\forall A'. C$ to $\mathcal{L}(y)$, where $A'$ is the result of $A$ reading $s$ obtained by switching initial states
   2. decide whether to add $C$ to $\mathcal{L}(y)$
      
      if $\forall A. C \in \mathcal{L}(y)$ and $A$ is in a final state, then add $C$ to $\mathcal{L}(y)$
Construction of $A_r$: working up the cycle-free (!) uses relation, for each role $r$,

1. construct automaton $A_r^1$ for $\text{expr}(s_1, \ldots, s_n) \cup r$
Tableau Algorithm for $\mathcal{RIQ} \ II$

Construction of $A_r$: working up the cycle-free (!) uses relation, for each role $r$,

1. construct automaton $A_r^1$ for $\mathsf{regexp}(s_1, \ldots, s_n) \cup r$
2. add a disjoint copy of $A_s$ for each $\bullet \xrightarrow{s} \bullet$ in $A_r^1$
3. add $\varepsilon$-transition from $\bullet \xrightarrow{s} \bullet$ to $\mathsf{init}(A_s)$
4. add $\varepsilon$-transitions from $\mathsf{final}(A_s)$ to $\bullet \xrightarrow{s} \bullet$

automaton $A_r$ for $r$ — whose size is possibly exponential in $\mathcal{R}$: unfolding
Construction of $A_r$: working up the cycle-free (!) uses relation, for each role $r$,

1. construct automaton $A^1_r$ for regexp($s_1, \ldots, s_n$) $\cup$ $r$
2. add a disjoint copy of $A_s$ for each $s$ in $A^1_r$
3. add $\epsilon$-transition from $s$ to $\text{init}(A_s)$
4. add $\epsilon$-transitions from $\text{final}(A_s)$ to $s$

automaton $A_r$ for $r$ — whose size is possibly exponential in $\mathcal{R}$: unfolding

Lemma:

$\mathcal{I}$ is a model of $\mathcal{R}$

iff

for each $r_1 \cdots r_n \in L(A_r)$, for each $x, y \in \Delta^\mathcal{I}$,

if $\langle x, y \rangle \in r^\mathcal{I}_1 \circ \cdots \circ r_n$, then $\langle x, y \rangle \in r^\mathcal{I}$
Implementing the Tableau Algorithm for RIQ

FaCT was extended to RIQ:

- each $A_r$ is transformed into minimal DFA using AT&T FSM Library\textsuperscript{TM} [MoPR98]
  - “only” pre-processing
  - yields fewer node labels in tableau algorithm $\Rightarrow$ smaller search space

- first tests on Galen medical terminology KB
  (2,740 named concepts, 413 roles, 26 transitive ones) is promising:
  - the (pre-)processing of RIAs takes some time: +100%
  - but satisfiability algorithm shows similar performance: only +3%
  - system can draw useful, additional inferences: e.g., w.r.t. the RIA

\[
\text{hasLocation } \sqsubseteq \text{divisionOf } \sqsubseteq \text{hasLocation, the concept}
\]

Fracture $\sqcap \exists \text{hasLocation}.(\text{Neck } \sqcap \exists \text{isDivisionOf}.\text{Femur})$

is indeed subsumed by

Fracture $\sqcap \exists \text{hasLocation}.\text{Femur}$
Extending successful $SHIQ$ with a sought-after constructor for propagation

**Results:**

1. a **naive** such extension leads to undecidability
2. a **semi-naive** such extension still leads to undecidability
3. a **careful** such extension, $RIQ$, yields a DL with
   - elegant tableau algorithm
   - behaviour similar to the one for $SHIQ$
   - being able to draw useful, additional inferences

**Open questions:**

1. is exponential blow-up **avoidable**?
2. how does implementation of $RIQ$ behave on other knowledge bases?