I.5.2 QL → OWL and DL-Lite$_R$ → $\mathcal{SROIQ}(D)$

QL → OWL is the sublanguage inclusion obtained by the syntactic restriction according to the definition of QL, see **NR6**. Since by definition, DL-Lite$_R$ is a syntactic restriction of $\mathcal{SROIQ}(D)$, DL-Lite$_R$ → $\mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

I.5.3 RL → OWL and RL → $\mathcal{SROIQ}(D)$

RL → OWL is the sublanguage inclusion obtained by the syntactic restriction according to the definition of RL, see **NR6**. Since by definition, RL is a syntactic restriction of $\mathcal{SROIQ}(D)$, RL → $\mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

I.5.4 SimpleRDF → RDF

SimpleRDF → RDF is an obvious inclusion, except that SimpleRDF resources need to be renamed if they happen to have a predefined meaning in RDF. The model translation needs to forget the fixed parts of RDF models. Since this part can always reconstructed in a unique way, the result is an isomorphic model translation.

I.5.5 RDF → RDFS

This is entirely analogous to SimpleRDF → RDF.

I.5.6 SimpleRDF → $\mathcal{SROIQ}(D)$

A SimpleRDF signature is translated to $\mathcal{SROIQ}(D)$ by providing a class P and three roles sub, $pred$ and obj (these reify the extension relation), and one individual per SimpleRDF resource. A SimpleRDF triple (s, p, o) is translated to the $\mathcal{SROIQ}(D)$ sentence

$$\top \sqsubseteq \exists U.(\exists sub.\{s\} \sqcap \exists pred.\{p\} \sqcap \exists obj.\{o\}).$$

From an $\mathcal{SROIQ}(D)$ model $\mathcal{I}$, obtain a SimpleRDF model by inheriting the universe and the interpretation of individuals (then turned into resources). The interpretation $P^{\mathcal{I}}$ of P gives P_m, and EXT_m is obtained by de-reifying, i.e.

$$EXT_m(x) := \{(y, z) \mid \exists u.(u, x) \in pred^{\mathcal{I}}, (u, y) \in sub^{\mathcal{I}}, (u, z,) \in obj^{\mathcal{I}}\}.$$

RDF → $\mathcal{SROIQ}(D)$ is defined similarly. The theory of RDF built-ins is (after translation to $\mathcal{SROIQ}(D)$) added to any signature translation. This ensures that the model translation can add the built-ins.

I.5.7 OWL → FOL

I.5.7.1 Translation of signatures

$\Phi((\mathbf{C}, \mathbf{R}, \mathbf{I})) = (F, P)$ with

- function symbols: $F = \{a^{(1)} | a \in \mathbf{I}\}$
- predicate symbols $P = \{A^{(1)} | A \in \mathbf{C}\} \cup \{R^{(2)} | R \in \mathbf{R}\}$

I.5.7.2 Translation of sentences

Concepts are translated as follows:

- $\alpha_x(A) = A(x)$
- $\alpha_x(\top) = true$
- $\alpha_x(\bot) = false$
- $\alpha_x(\neg C) = \neg\alpha_x(C)$
- $\alpha_x(C \sqcap D) = \alpha_x(C) \wedge \alpha_x(D)$
- $\alpha_x(C \sqcup D) = \alpha_x(C) \vee \alpha_x(D)$
- $\alpha_x(\exists R.C) = \exists y.(R(x,y) \wedge \alpha_y(C))$
- $\alpha_x(\exists U.C) = \exists y.\alpha_y(C)$
- $\alpha_x(\forall R.C) = \forall y.(R(x,y) \rightarrow \alpha_y(C))$
- $\alpha_x(\forall U.C) = \forall y.\alpha_y(C)$
- $\alpha_x(\exists R.\mathrm{Self}) = R(x,x)$
- $\alpha_x(\leq nR.C) = \forall y_1, \ldots, y_{n+1}. \bigwedge_{i=1,\ldots,n+1}(R(x,y_i) \wedge \alpha_{y_i}(C)) \rightarrow \bigvee_{1 \leq i < j \leq n+1} y_i = y_j$
- $\alpha_x(\geq nR.C) = \exists y_1, \ldots, y_n. \bigwedge_{i=1,\ldots,n}(R(x,y_i) \wedge \alpha_{y_i}(C)) \wedge \bigwedge_{1 \leq i < j \leq n} y_i \neq y_j$
- $\alpha_x(\{a_1, \ldots a_n\}) = (x = a_1 \vee \ldots \vee x = a_n)$

For inverse roles R^-, $R^-(x,y)$ has to be replaced by $R(y,x)$, e.g.

$$\alpha_x(\exists R^-.C) = \exists y.(R(y,x) \wedge \alpha_y(C))$$

This rule also applies below.

Sentences are translated as follows:

- $\alpha_\Sigma(C \sqsubseteq D) = \forall x.\,(\alpha_x(C) \rightarrow \alpha_x(D))$
- $\alpha_\Sigma(a : C) = \alpha_x(C)[x \mapsto a]$[46)]
- $\alpha_\Sigma(R(a,b)) = R(a,b)$
- $\alpha_\Sigma(R \sqsubseteq S) = \forall x,y.R(x,y) \rightarrow S(x,y)$
- $\alpha_\Sigma(R_1; \ldots; R_n \sqsubseteq R) =$
  $\forall x,y.(\exists z_1, \ldots, z_{n-1}.R_1(x,z_1) \wedge R_2(z_1,z_2) \wedge \ldots \wedge R_n(z_{n-1},y)) \rightarrow R(x,y)$
- $\alpha_\Sigma(\mathrm{Dis}(R_1,R_2)) = \neg\exists x,y.R_1(x,y) \wedge R_2(x,y)$
- $\alpha_\Sigma(\mathrm{Ref}(R)) = \forall x.R(x,x)$
- $\alpha_\Sigma(\mathrm{Irr}(R)) = \forall x.\neg R(x,x)$
- $\alpha_\Sigma(\mathrm{Asy}(R)) = \forall x,y.R(x,y) \rightarrow \neg R(y,x)$
- $\alpha_\Sigma(\mathrm{Tra}(R)) = \forall x,y,z.R(x,y) \wedge R(y,z) \rightarrow R(x,z)$

I.5.7.3 Translation of models

- For $M' \in \mathrm{Mod}^{FOL}(\Phi\Sigma)$ define ~~$\beta_\Sigma(M') := (\Delta, \cdot^I)$~~ $\mathcal{I} = \beta_\Sigma(M') := (\Delta, \cdot^\mathcal{I})$ with $\Delta = |M'|$ and ~~$A^I = M'_A, a^I = M'_a, R^I = M'_R$~~ $A^\mathcal{I} = M'_A, a^\mathcal{I} = M'_a, R^\mathcal{I} = M'_R$.

Proposition 24 ~~$C^\mathcal{I} = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \alpha_x(C)\}$~~ $C^\mathcal{I} = \{m \in \Delta | M' + \{x \mapsto m\} \models \alpha_x(C)\}$

Proof. By ~~Induction~~ induction over the structure of C.

- ~~$A^\mathcal{I} = M'_A = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models A(x)\}$~~ $A^\mathcal{I} = M'_A = \{m \in \Delta | M' + \{x \mapsto m\} \models A(x)\}$
- ~~$(\neg C)^\mathcal{I} = \Delta \setminus C^\mathcal{I} =^{I.H.} \Delta \setminus \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \alpha_x(C)\} = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \neg\alpha_x(C)\}$~~ $(\neg C)^\mathcal{I} = \Delta \setminus C^\mathcal{I} =^{I.H.} \Delta \setminus \{m \in \Delta | M' + \{x \mapsto m\} \models \alpha_x(C)\} = \{m \in \Delta | M' + \{x \mapsto m\} \models \neg\alpha_x(C)\}$

46) $t[x \mapsto a]$ means "in t, replace x by a".

□

The ~~satisfaction condition holds as well~~other cases are similar.

The satisfaction condition now follows easily.

I.5.8 $FOL \to$ CL

This comorphism maps classical first-order logic (FOL) to Common Logic.

A FOL signature is translated to CL.Fol by turning all constants into discourse names, and all other function symbols and all predicate symbols into non-discourse names. A FOL sentence is translated to CL.Fol by a straightforward recursion, the base being translations of predications:

$$\alpha_\Sigma(P(t_1,\ldots,t_n)) = (P\ \alpha_\Sigma(t_1)\ \ldots\ \alpha_\Sigma(t_n))$$

Within terms, function applications are translated similarly:

$$\alpha_\Sigma(f(t_1,\ldots,t_n)) = (f\ \alpha_\Sigma(t_1)\ \ldots\ \alpha_\Sigma(t_n))$$

A CL.Fol model is translated to a FOL model by using the universe of discourse as FOL universe. The interpretation of constants is directly given by the interpretation of the corresponding names in CL.Fol. The interpretation of a predicate symbol P is given by using $rel^M(int^M(P))$ and restricting to the arity of P; similarly for function symbols (using fun^M). Both the satisfaction condition and model-expansiveness of the comorphism are straightforward.

I.5.9 OWL $\to$ CL

This comorphism is the composition of the comorphisms described in the previous two sections.

I.5.10 UML class models $\to$ CL

This translation has been described in annex F. Translation of signatures is detailed in section F.4.3, translation of sentences in section F.4.5. Models are translated identically.

I.5.11 $FOL \to$ Casl

This is an obvious sublogic.

I.5.12 UML class model to OWL

Let $\Sigma = ((C, \leq_C), P, O, A, M)$ be a *class/data type net* representing a UML class model as described in annex F. This net can be translated to OWL2 using the approach described in [76]. The ontology is extended by translating parts of this net and its multiplicity constraints $Mult(\Sigma)$:

— For each class $c \in C$ with superclasses $c_1, c_2, \ldots, c_n \in C$ (i.e. $c \leq_C c_i$ for $i = 1, \ldots, n$):

```
Class: c
        SubClassOf: c1
        ...
        SubClassOf: cn
```

— For each attribute declaration $c.p : c'$ in P