Let $[\ \ ]^p_q$ be a mapping that maps OWL 2 DL axioms (and OWL expressions that are part of axioms) to (sets of) FOL expressions. It has two (optional) parameters p, q, which we will use to keep track of substitutions that need to be made. When the translation function is called on an axiom, the parameters are empty, i.e. $p = q = \emptyset$. If a parameter is empty, we omit it and, for example, write $[E]^p$ instead of $[E]^p_\emptyset$ The OWL expressions are based on the specification of the OWL 2 functional syntax (OWLFS)[1].

We use A, B as symbols for class expressions, and R for object property expressions (with and without indices). *Ann* are annotations.

Axioms highlighted in green are already implemented and tested successfully. Yellow axioms are implemented, but not tested satisfactorily. Red axioms are not implemented yet.

1 Background Axioms

$\forall x(\mathit{owl{:}Thing}(x) \vee \mathit{rdfs{:}Literal}(x))$	our domain consists of objects and data
$\forall x\, \mathit{owl{:}Thing} \rightarrow \neg \mathit{rdfs{:}Literal}$	Object domain and data domain are disjunct
$\exists x\, \mathit{owl{:}Thing}(x)$	There are things
$\exists x\, \mathit{rdfs{:}Literal}(x)$	The data domain is nonempty, too
$\forall x(C(x) \rightarrow \mathit{owl{:}Thing}(x))$	for root class C (i.e., a class that is not a subclass of anything)
$\forall xy(P(x,y) \rightarrow \mathit{owl{:}Thing}(x))$	for every object property P (redundant if there is a domain axiom for P)
$\forall xy(P(x,y) \rightarrow \mathit{owl{:}Thing}(y))$	for every object property P (redundant if there is a range axiom for P)
$\forall xy(P(x,y) \rightarrow \mathit{owl{:}Thing}(x))$	for every data property P (redundant if there is a domain axiom for P)
$\forall xy(P(x,y) \rightarrow \mathit{rdfs{:}Literal}(y))$	for every data property P (redundant if there is a range axiom for P)
$\forall xy(F(x,y) \rightarrow \mathit{rdfs{:}Literal}(x))$	for every facet F that occurs in the text
$\forall xy(F(x,y) \rightarrow \mathit{rdfs{:}Literal}(y))$	for every facet F that occurs in the text
$\mathit{owl{:}Thing}(a)$	for every individual a (redundant if a occurs in a class assertion / positive object property / positive data property a
$\mathit{rdfs{:}Literal}(l)$	for every literal l (redundant if l occurs in a positive data property axiom or in a datatype restriction)

The following axioms need only to be added, if the corresponding owl keyword is used.

owl:Nothing	$\forall x \neg \mathit{owl{:}Nothing}(x)$
owl:topObjectProperty	$\forall xy(\mathit{owl{:}topObjectProperty}(x,y) \leftrightarrow (\mathit{owl{:}Thing}(x) \wedge \mathit{owl{:}Thing}(y))$
owl:bottomObjectProperty	$\forall xy \neg \mathit{owl{:}bottomObjectProperty}(x,y)$
owl:topDataProperty	$\forall xy(\mathit{owl{:}topObjectProperty}(x,y) \leftrightarrow (\mathit{owl{:}Thing}(x) \wedge \mathit{rdfs{:}Literal}(y))$
owl:bottomDataProperty	$\forall xy \neg \mathit{owl{:}bottomDataProperty}(x,y)$

[1] `https://www.w3.org/TR/owl2-syntax/`

The following axioms need only to be added, if the datatype DT is used.

$$\forall x(DT(x) \rightarrow \mathit{rdfs{:}Literal}(x))$$

The OWL standard allows user-defined datatypes (Datatypes are just IRIs), but the datatypes that are explicitly supported by OWL (and, thus, are commonly used) are: owl:real, owl:rational, xsd:decimal, xsd:integer, xsd:nonNegativeInteger, xsd:nonPositiveInteger, xsd:positiveInteger, xsd:negativeInteger, xsd:long, xsd:int, xsd:short, xsd:byte, xsd:unsignedLong, xsd:unsignedInt, xsd:unsignedShort, xsd:unsignedByte, rdf:PlainLiteral, xsd:string, xsd:normalizedString, xsd:token, xsd:language, xsd:Name, xsd:NCName, xsd:NMTOKEN, xsd:boolean, xsd:hexBinary, xsd:base64Binary, xsd:anyURI, xsd:dateTime, xsd:dateTimeStamp, rdf:XMLLiteral.

There are relationships between these datatypes that may be expressed as first-order axioms (e.g., owl:rational is a subclass of owl:real). We can add these later.

2 Logical Axiom Visitor

2.1 Class Expression Axioms

Corresponds to section 9.1. in OWLFS.

Name	E	$[E]^p_q$	
Subclass Axiom	SubClassOf(Ann, A,B)	$\forall x([A]^x \rightarrow [B]^x)$	new variable x; skip Ann for now
Equivalent Classes	EquivalentClasses(Ann, A, B)	$\forall x([A]^x \leftrightarrow [B]^x)$	new variable x; skip Ann for now
Disjoint Classes	DisjointClasses(Ann,A,B)	$\forall x \neg([A]^x \wedge [B]^x)$	new variable x; skip Ann for now
Disjoint Union	DisjointUnion($Ann, A, B_1, \ldots, B_n$)	removed by OWL API during preprocessing	

For Disjoint Classes, we only need the Binary Case, because the OWL API is able to map DisjointClasses(Ann,A_1, ..., A_n) to a set of pairwise DisjointClasses-Axioms.

2.2 Object Property Axioms

Corresponds to section 9.2. in OWLFS.

Name	$E(OWL \dots Axiom)$	$[E]_q^p$	
Object Subproperty	SubObjectPropertyOf(Ann, R_1, R_2)	$\forall x, y([R_1]_y^x \rightarrow [R_2]_y^x)$	skip Ann for now
	R_1 and R_2 can be Object Properties or...	Property Expression Chains	
Equivalent Object Properties	EquivalentObjectProperties(Ann, R_1, R_2)	$\forall x, y([R_1]_y^x \leftrightarrow [R_2]_y^x)$	skip Ann for now
Disjoint Object Properties	DisjointObjectProperties(Ann, R_1, R_2)	$\forall x, y \neg([R_1]_y^x \wedge [R_2]_y^x)$	skip Ann for now
Inverse Object Properties	InverseObjectProperties(Ann, R_1, R_2)	$[EquivalentProperties(Ann, (R_1),$ $ObjectInverseOf(R_2))]_q^p$	
Object Property Domain	ObjectPropertyDomain(Ann, R, A)	$\forall x, y([R]_y^x \rightarrow [A]^x)$	skip Ann for now
Object Property Range	ObjectPropertyRange(Ann, R, A)	$\forall x, y([R]_y^x \rightarrow [A]^y)$	skip Ann for now
Functional Object Properties	FunctionalObjectProperty(Ann, R)	$\forall x, y, z(([R]_y^x \wedge [R]_z^x) \rightarrow y = z)$	skip Ann for now
Inverse-Functional Object Properties	InverseFunctionalObjectProperty(Ann, R)	$[FunctionalObjectProperty(Ann,$ $ObjectInverseOf(R))]_q^p$	
Reflexive Object Properties	ReflexiveObjectProperty(Ann, R)	$\forall x([R]_x^x)$	skip Ann for now
Irreflexive Object Properties	IrreflexiveObjectProperty(Ann, R)	$\forall x \neg([R]_x^x)$	skip Ann for now
Symmetric Object Properties	SymmetricObjectProperty(Ann, R)	$\forall x, y([R]_y^x \rightarrow [R]_x^y)$	skip Ann for now
Asymmetric Object Properties	AsymmetricObjectProperty(Ann, R)	$\forall x, y \neg([R]_y^x \wedge [R]_x^y)$	skip Ann for now
Transitive Object Properties	TransitiveObjectProperty(Ann, R)	$\forall x, y, z(([R]_y^x \wedge [R]_z^y) \rightarrow [R]_z^x)$	skip Ann for now

Note on Functional Object Properties: We chose not to translate them as functions in FOL, but instead treat them as binary predicates (as with any other object property). This is because mapping all functional roles to functions in FOL would be incorrect, since in standard FOL functions are total, but in OWL functional roles do not need to be total.

If R is a total functional role in some OWL ontology and the user wishes to link it to some function f, which occurs in FOL annotations, this may be achieved by adding the following FOL axiom as annotation:
(forall x (forall y (iff (R x y) (= (f x) y)))))

2.3 Data Property Axioms

Corresponds to section 9.3. in OWLFS.

Name	E	$[E]^p_q$	
Data Subproperties	SubDataPropertyOf(Ann, R_1, R_2)	$\forall x, y([R_1]^x_y \rightarrow [R_2]^x_y)$	skip Ann for now
Equivalent Data Properties	EquivalentDataProperties(Ann, R_1, R_2)	$\forall x, y([R_1]^x_y \leftrightarrow [R_2]^x_y)$	skip Ann for now
Disjoint Data Properties	DisjointDataProperties(Ann, R_1, R_2)	$\forall x, y \neg([R_1]^x_y \wedge [R_2]^x_y)$	skip Ann for now
Data Property Domain	DataPropertyDomain(Ann, R, A)	$\forall x, y([R]^x_y \rightarrow [A]^x)$	skip Ann for now
Data Property Range	DataPropertyRange(Ann, R, DR)	$\forall x, y([R]^x_y \rightarrow [DR]^y)$	skip Ann for now
Functional Data Properties	FunctionalDataProperty(Ann, R)	$\forall x, y, z(([R]^x_y \wedge [R]^x_z) \rightarrow y = z)$	skip Ann for now

2.4 Assertions Axioms

Corresponds to section 9.6. in OWLFS.

Name	E	$[E]^p_q$ oder rekursiv owl	
Individual Equality	SameIndividualAxiom(a,b)	a = b	
Individual Inequality	DifferentIndividualsAxiom(a,b)	a != b	
Class Assertion	ClassAssertionAxiom(Ann,A,b)	$[A]^b$	skip Ann for now
Positive Object Property Assertion	ObjectPropertyAssertionAxiom(Ann,R,a,b)	$[R]^a_b$	skip Ann for now
Negative Object Property Assertion	NegativeObjectPropertyAssertionAxiom(Ann,R,a,b)	$\neg[R]^a_b$	skip Ann for now
Positive Data Property Assertion	DataPropertyAssertionAxiom(Ann,R,a,l)	$[R]^a_l$	skip Ann for now
Negative Data Property Assertion	NegativeDataPropertyAssertionAxiom(Ann,R,a,l)	$\neg[R]^a_l$	skip Ann for now

2.5 Other Axioms

Name	E	$[E]^p_q$	
HasKey	HasKey(A ($B_1 \ldots B_m$) ($D_1 \ldots D_n$))	$\forall x, y, z_1, \ldots, z_m, w_1, \ldots, w_n(([A]^x \wedge [A]^y \wedge [B_1]^x_{z_1} \wedge \ldots \wedge [B_m]^x_{z_m} \wedge [B_1]^y_{z_1} \wedge \ldots \wedge [B_m]^y_{z_m} \wedge [D_1]^x_{w_1} \wedge \ldots \wedge [D_n]^x_{w_n} \wedge [D_1]^y_{w_1} \wedge \ldots \wedge [D_n]^y_{w_n}) \rightarrow x = y)$	
Datatype Definition	DatatypeDefinition(Ann, DT, DR)	$\forall x([DT]^x \leftrightarrow [DR]^x)$	skip for now, section 9.4 in OWLFS.
SWRLRule			skip for now

3 Class Expression Visitor

Corresponds to section 8 in OWLFS.

3.1 Individuals

Name	E	$[E]^p_q$
ObjectIntersectionOf	ObjectIntersectionOf $(A_1, \ldots, A_n)$	$([A_1]^p_q \wedge \ldots \wedge [A_n]^p_q)$
ObjectUnionOf	ObjectUnionOf $(A_1, \ldots, A_n)$	$([A_1]^p_q \vee \ldots \vee [A_n]^p_q)$
ObjectComplementOf	ObjectComplementOf(A)	$\neg[A]^p_q \wedge \mathit{owl{:}Thing}(p)$
ObjectOneOf	ObjectOneOf$(a_1, \ldots, a_n)$	$p = a_1 \vee p = a_2 \vee \ldots \vee p = a_n$

3.2 Object Property Restrictions

Name	E	$[E]^p_q$	
ObjectSomeValuesFrom	ObjectSomeValuesFrom(R,A)	$\exists x([R]^p_x \wedge [A]^x)$	new variable x
ObjectAllValuesFrom	ObjectAllValuesFrom(R,A)	$\forall x(\mathit{owl{:}Thing}(p) \wedge ([R]^p_x \rightarrow [A]^x))$	new variable x
Individual Value Restriction	ObjectHasValue(R,a)	$[R]^p_a$	
Self-Restriction	ObjectHasSelf(R)	$[R]^p_p$	

Remark: *owl:Thing* has been added to ObjectAllValuesFrom in PR #19

3.3 Object Property Cardinality Restrictions

Name	E	$[E]^p_q$ oder rekursiv owl
Minimum Cardinality	ObjectMinCardinality(n,R,A)	$\exists x_1, \ldots, x_n(x_1 \neq x_2 \wedge \ldots \wedge x_1 \neq x_n \wedge x_2 \neq x_3 \wedge \ldots \wedge x_{n-1} \neq x_n$ $\wedge [A]^{x_1} \wedge \ldots \wedge [A]^{x_n} \wedge [R]^p_{x_1} \wedge \ldots \wedge [R]^p_{x_n})$
Maximum Cardinality	ObjectMaxCardinality(n,R,A)	$\forall x_1, \ldots, x_{n+1}(([A]^{x_1} \wedge \ldots \wedge [A]^{x_{n+1}} \wedge [R]^p_{x_1} \wedge \ldots \wedge [R]^p_{x_{n+1}})$ $\rightarrow \neg(x_1 \neq x_2 \wedge \ldots \wedge x_1 \neq x_{n+1} \wedge x_2 \neq x_3 \wedge \ldots \wedge x_n \neq x_{n+1}))$
Exact Cardinality	ObjectExactCardinality(n,R,A)	[ObjectIntersectionOf(ObjectMinCardinality(n,R,A), ObjectMaxCardinality(n,R,A))]^p_q

3.4 Data Property Restrictions

Name	E	$[E]^p_q$	
DataSomeValuesFrom	DataSomeValuesFrom(DPE,DR)	$\exists x([DPE]^p_x \wedge [DR]^x)$	new variable x, see remark below
DataAllValuesFrom	DataAllValuesFrom(DPE,DR)	$\forall x(owl{:}Thing(p) \wedge ([DPE]^p_x \rightarrow [DR]^x))$	new variable x, see remark below
Literal Value Restriction	DataHasValue(DPE,l)	$[DPE]^p_l$	

Remark: *owl:Thing* has been added to DataAllValuesFrom in PR #19.

Technically, OWL 2 allows more than one data property expressions to occur, i.e., DataSomeValuesFrom(DPE$_1$...,DPE$_n$,DR) is valid. But since in OWL 2 data ranges are unary, this feature is not supported by OWL 2 DL.

3.5 Data Property Cardinality Restrictions

Name	E	$[E]^p_q$
Minimum Cardinality	DataMinCardinality(n, DPE, DR)	$\exists x_1, \ldots, x_n(x_1 \neq x_2 \wedge \ldots \wedge x_1 \neq x_n \wedge x_2 \neq x_3 \wedge \ldots \wedge x_{n-1} \neq x_n$ $\wedge [DR]^{x_1} \wedge \ldots \wedge [DR]^{x_n} \wedge [DPE]^p_{x_1} \wedge \ldots \wedge [DPE]^p_{x_n})$
Maximum Cardinality	DataMaxCardinality(n, DPE, DR)	$\forall x_1, \ldots, x_{n+1}(([DR]^{x_1} \wedge \ldots \wedge [DR]^{x_{n+1}} \wedge [DPE]^p_{x_1} \wedge \ldots \wedge [DPE]^p_{x_{n+1}})$ $\rightarrow \neg(x_1 \neq x_2 \wedge \ldots \wedge x_1 \neq x_{n+1} \wedge x_2 \neq x_3 \wedge \ldots \wedge x_n \neq x_{n+1}))$
Exact Cardinality	DataExactCardinality(n, DPE, DR)	$[DataMinCardinality(n, DPE, DR)]^p_q \wedge [DataMaxCardinality(n, DPE, DR]^p_q$

4 Property Expressions

Corresponds to section 6 in OWLFS.

This is translated by the Property Expression Translator.

Name	E	$[E]^p_q$
InverseObjectProperty	inverse P	$[P]^q_p$

5 Data Ranges

Corresponds to section 7 in OWLFS.

Name	E	$[E]^p_q$
DataIntersectionOf	DataIntersectionOf$(DR_1, \ldots, DR_n)$	$[DR_1]^p_q \wedge \ldots \wedge [DR_n]^p_q$
DataUnionOf	DataUnionOf$(DR_1, \ldots, DR_n)$	$[DR_1]^p_q \vee \ldots \vee [DR_n]^p_q$
DataComplementOf	DataComplementOf(DR)	$\neg[DR]^p_q \wedge rdfs : Literal(p)$
DataOneOf	DataOneOf$(l_1, \ldots, l_n)$	$p = l_1 \vee \ldots \vee p = l_n$
DatatypeRestriction	DatatypeRestriction$(DT, F_1\ l_1 \ldots F_n\ l_n\ )$	$[DT]^p_q \wedge [F_1]^p_{l_1} \wedge \ldots \wedge [F_n]^p_{l_n}$

6 Entities, Literals, and Anonymous Literals

Corresponds to section 5 in OWLFS.

Name	E	$[E]^p_q$	
Class	C	C(p)	
Data type	DT	DT(p)	
Object property	P	P(p,q)	
Property expression chain	propertyExpressionChain$(R_1, ...R_n)$	$\exists x_1, x_2, \ldots, x_{n-1}([R_1]^p_{x_1} \wedge [R_2]^{x_1}_{x_2} \wedge \ldots \wedge [R_n]^{x_{n-1}}_q)$	can be used instead of object properties for a subObjectPropertyAxiom
Data property	P	P(p,q)	
Annotation property			skip for now
Facet	F	F(p,q)	
Individual	a	a	
Anonymous Individual	a	a	
Literal	l	l	
Entity Declaration			skip for now, used for background axioms

Class names are mapped to unary predicates (on objects), object property names to binary predicates (on objects). Both individual names and anonymous individual names are mapped to individual constants.

Example:

```
Class: Frog
  SubClassOf: Green and hasPart some Leg
```

$[SubClassOf(Frog, OIntersectionOf(Green, OSomeValuesFrom(hasPart, Leg)))]$
$<=> \quad \forall x([Frog]^x \rightarrow [OIntersectionOf(Green, OSomeValuesFrom(hasPart, Leg))]^x)$

$$<=> \quad \forall x(Frog(x) \rightarrow [OIntersectionOf(Green, OSomeValuesFrom(hasPart, Leg))]^x)$$
$$<=> \quad \forall x(Frog(x) \rightarrow ([Green]^x \wedge [OSomeValuesFrom(hasPart, Leg)]^x))$$
$$<=> \quad \forall x(Frog(x) \rightarrow (Green(x) \wedge [OSomeValuesFrom(hasPart, Leg)]^x))$$
$$<=> \quad \forall x(Frog(x) \rightarrow (Green(x) \wedge \exists y([hasPart]_y^x \wedge [Leg]^y)))$$
$$<=> \quad \forall x(Frog(x) \rightarrow (Green(x) \wedge \exists y(hasPart(x, y) \wedge Leg(y))))$$