OntoMerge: A System for Merging DL-Lite Ontologies

Zhe Wang\(^1\), Kewen Wang\(^1\), Yifan Jin\(^1\), and Guilin Qi\(^2\)\(^3\)

\(^1\) School of ICT, Griffith University, Nathan, QLD 4111, Australia
\(^2\) School of CSE, Southeast University, Nanjing, 211189, China
\(^3\) State Key Lab for Novel Software Technology, Nanjing University, Nanjing, China

Abstract. This paper proposes a novel approach to merging DL-Lite ontologies by utilising an alternative semantic characterisation instead of classical DL models. It is shown that the complexity of our merging operators is on the same level as the complexity of propositional belief merging. Moreover, a new algorithm is developed for transforming the reasoning problem of DL-Lite merging into the evaluation of Quantified Boolean Formulas (QBFs). We present our system OntoMerge, which can effectively verify the logical consequences of merging multiple ontologies, without first computing the merging result.

1 Introduction

Ontologies are widely used for sharing and reasoning over domain knowledge, and their underlying formalisms are often description logics (DLs). Various ontologies developed by multiple authors under different settings may contain overlapping and possibly incoherent domain knowledge. The term “ontology merging” may have different meanings for researchers in different research communities. A broad understanding of ontology merging encompasses several topics including ontology matching and integration. Significant research effort has been made in ontology matching [4,10], but relatively few work has been done on integration of ontologies using a logical approach. In this paper, we focus on a different aspect of ontology merging whose ultimate goal is to obtain a single consistent ontology that includes the information from two or more heterogeneous ontologies.

Given that OWL 2 is the latest W3C standard for ontology languages [8][\(^4\)], it is important to develop various methods for merging and managing OWL 2 ontologies. OWL 2 has three profiles: OWL 2 QL, OWL 2 EL, and OWL 2 RL. Especially, the logical foundations of OWL 2 QL is a well known family of lightweight DLs, DL-Lite [1].

Approaches to merging DL ontologies have been proposed in the literature [14], and some other topics are also in the scope of ontology merging but under different names, such as ontology debugging [12], repair [11], and conservative extension [6]. Extant approaches to ontology merging can be improved from several aspects: (1) Conservative extension is a safe condition for union of two

\[^4\]http://www.w3.org/TR/owl2-overview/
ontologies, but it does not resolve incoherence. (2) Some approaches are purely syntactic and lack of a suitable semantic justification, \textit{i.e.}, it is unclear how close the result of merging is to the source ontologies semantically. On the other hand, the standard semantic structures, DL models, often are infinite structures and are infinite in number, which makes it difficult to work directly with models. (3) Special merging techniques that utilise the advantages of certain lightweight DLs, such as DL-Lite, are rarely exploited.

As a balance between the syntax-based approach and model-based approach, an alternative semantic characterisation, called \textit{features}, has been proposed in [17]. While the form of features is syntactic, the semantics defined in terms of features is extremely close to the classical semantics of DL-Lite. However, the structure of a feature is still too complex for efficient merging. In this paper, we first introduce simplified forms of features and then propose a novel approach to merging DL-Lite TBoxes. Moreover, given the close relation of features to the classical semantics of DL-Lite, our approach possesses a nice semantic justification, which is a major difference of the new approach to others. For the computational aspects, we transform the entailment problem of ontology merging in DL-Lite into the satisfiability of Quantified Boolean Formulas (QBFs). Thus our approach to ontology merging can be implemented using off-the-shelf QBF solvers. We present our system OntoMerge, which effectively check logical consequences of merging multiple ontologies, without first computing the merging results.

2 A New Semantic Characterisation

In our approach, it is sufficient to consider a finite yet large enough signature. A \textit{signature} $S$ is a union of four disjoint finite sets $S_C$, $S_R$, $S_I$ and $S_N$, where $S_C$ is the set of atomic concepts, $S_R$ is the set of atomic roles, $S_I$ is the set of individual names and $S_N$ is the set of natural numbers in $S$. We assume 1 is always in $S_N$. In the following sections, if not specified, we assume $S$ to be a fixed large enough signature. Formally, a DL-Lite$_{bool}$ language has the following syntax [1]:

$$
R \rightarrow P \mid P^- \quad \quad S \rightarrow P \mid \neg P \\
B \rightarrow \top \mid A \mid \geq n \ R \quad C \rightarrow B \mid \neg C \mid C_1 \sqcap C_2
$$

where $n \in S_N$, $A \in S_C$ and $P \in S_R$. $B$ is called a \textit{basic concept} and $C$ is called a \textit{general concept}. $B_S$ denotes the set of basic concepts on $S$. We write $\bot$ for $\neg \top$, $\exists R$ for $\geq 1 R$, and $C_1 \sqcup C_2$ for $\neg (\neg C_1 \sqcap \neg C_2)$. For $P \in S_R$, let $R^- = P$ if $R = P^-$, and $R^+ = P$ whenever $R = P$ or $R = P^-$. A TBox $\mathcal{T}$ is a finite set of concept axioms of the form $C_1 \sqsubseteq C_2$, where $C_1$ and $C_2$ are general concepts. An ABox $\mathcal{A}$ is a finite set of membership assertions of the form $C(a)$ or $S(a,b)$, where $a, b$ are individual names. In this paper, an ontology is represented as a DL TBox.

The classical DL semantics are given by models. A TBox $\mathcal{T}$ is \textit{consistent} with an ABox $\mathcal{A}$ if $\mathcal{T} \cup \mathcal{A}$ has at least one model. A concept or role is \textit{satisfiable}
in \( T \) if it has a non-empty interpretation in some model of \( T \). A TBox \( T \) is \textit{coherent} if all atomic concepts and atomic roles in \( T \) are satisfiable. Note that a coherent TBox must be consistent. TBox \( T \) \textit{entails} an axiom \( C \sqsubseteq D \), written \( T \models C \sqsubseteq D \), if all models of \( T \) satisfy \( C \sqsubseteq D \). Two TBoxes \( T_1, T_2 \) are \textit{equivalent}, written \( T_1 \equiv T_2 \), if they have the same models.

Now, we introduce a semantic characterisation for DL-Lite TBoxes in terms of \textit{types}. A \textit{type} \( \tau \subseteq B_S \) is a set of basic concepts over \( S \), such that \( \top \in \tau \), and \( n R \in \tau \) implies \( m R \in \tau \) for each pair \( m, n \in S_N \) with \( m < n \) and each (inverse) role \( R \in S_R \cup \{ \overline{P} \mid P \in S_R \} \). Type \( \tau \) \textit{satisfies} basic concept \( B \) if \( B \in \tau \), \( \neg C \) if \( \tau \) does not satisfy \( C \), and \( C_1 \sqcap C_2 \) if \( \tau \) satisfies both \( C_1 \) and \( C_2 \). Given a TBox \( T \), type \( \tau \) satisfies \( T \) if \( \tau \) satisfies concept \( \neg C_1 \sqcup C_2 \) for each axiom \( C_1 \sqsubseteq C_2 \) in \( T \). This satisfaction relation for types is natural but not sufficient for characterizing the semantics of DL-Lite. For example, it cannot capture the the non-propositional inference from \( \exists R \sqsubseteq \bot \) to \( \exists R^\bot \sqsubseteq \bot \). This observation leads to the following definition of a new satisfaction relation.

**Definition 1.** A \textit{type} \( \tau \) \textit{s} a \textit{type model} (\textit{T-model}) of \( T \) if the following two conditions are satisfied: (1) \( \tau \) satisfies \( T \), and (2) if \( T \models \exists R \sqsubseteq \bot \) then \( \exists R^\bot \not\models \tau \).

The set of T-models of \( T \) is denoted as \( \text{TM}(T) \).

Note that for a coherent TBox \( T \), \( \text{TM}(T) \) is exactly the set of all types satisfying \( T \). Let \( \text{TM}(\Pi) = \text{TM}(T_1) \times \cdots \times \text{TM}(T_n) \) for \( \Pi = \langle T_1, \ldots, T_n \rangle \).

**Proposition 1.** Given a TBox \( T \), we have the following results:

- \( T \) is consistent iff \( \text{TM}(T) \neq \emptyset \).
- A concept \( C \) is satisfiable wrt \( T \) iff a type in \( \text{TM}(T) \) exists satisfying \( C \).
- \( T \models C \sqsubseteq D \) iff \( \text{TM}(T) = \emptyset \) or all types in \( \text{TM}(T) \) satisfy \( C \sqsubseteq D \).
- \( T \equiv T' \) iff \( \text{TM}(T) = \text{TM}(T') \) for any TBox \( T' \).

Given an individual \( b \) and an ABox \( A \), we say that type \( \tau \) is a \textit{type of \( b \)} w.r.t. \( A \) if there is a model \( I \) of \( A \) such that \( \tau = \{ B : b^I \in B^I, B \in B_S \} \). For example, given \( A = \{ A(b), \neg B(c), C(d) \} \), type \( \tau = \{ A, B \} \) is a type of \( b \), but not a type of either \( c \) or \( d \) in \( A \). For convenience, we will say a \textit{type of \( b \)} when the ABox \( A \) is clear from the context. Let \( \text{TM}_b(A) \) be the set of all the types of \( b \) in \( A \) if \( b \) occurs in \( A \); and otherwise, \( \text{TM}_b(A) \) be the set of all the types. A set \( M \) of T-models \textit{satisfies} an ABox \( A \) if there is a type of \( b \) in \( M \), i.e., \( M \cap \text{TM}_b(A) \neq \emptyset \), for each individual \( b \) in \( A \).

**Proposition 2.** Given a TBox \( T \) and an ABox \( A \), \( T \cup A \) is consistent iff \( \text{TM}(T) \cap \text{TM}_b(A) \neq \emptyset \) for each \( b \) in \( A \).

### 3 Merging Operator

In this section, we introduce an approach to merge DL-Lite ontologies to obtain a coherent unified ontology.

An ontology profile is of the form \( \Pi = \langle T_1, \ldots, T_n \rangle \), where \( T_i \) is the ontology from the source n.o. \( i \) (\( 1 \leq i \leq n \)). There are two standard definitions
of integrity constraints (ICs) in the classical belief change literature \cite{2}, the consistency- and entailment-based definitions. We also allow two types of ICs for merging, namely the \textit{consistency constraint} (CC), expressed as a set $\mathcal{A}_c$ of data, and the \textit{entailment constraint} (EC), expressed as a TBox $\mathcal{T}_e$. For example, suppose we need to merge several TBoxes to create an ontology for academic publications. Then, the CC $\mathcal{A}_c$ can contain assertions like Book(SN3721) and hasAuthor(SN3721, J.Smith), and the EC $\mathcal{T}_e$ may contain axioms such as Author $\sqsubseteq \neg$Book. In practice, the users are allowed to specify (n)either or both type of the constraints. We assume the IC is self-consistent, that is, $\mathcal{T}_e \cup \mathcal{A}_c$ is always coherent. For an ontology profile $\Pi$, a CC $\mathcal{A}_c$ and an EC $\mathcal{T}_e$, an ontology merging operator is a mapping $(\Pi, \mathcal{T}_e, \mathcal{A}_c) \mapsto \nabla(\Pi, \mathcal{T}_e, \mathcal{A}_c)$, where $\nabla(\Pi, \mathcal{T}_e, \mathcal{A}_c)$ is a TBox, s.t. $\nabla(\Pi, \mathcal{T}_e, \mathcal{A}_c) \cup \mathcal{A}_e$ is consistent, and $\nabla(\Pi, \mathcal{T}_e, \mathcal{A}_c) \models \mathcal{T}_e$.

In classical model-based merging, merging operators are often defined based on certain notions of \textit{model distances} \cite{13,15}. We use $S \triangle S'$ to denote the symmetric difference between two sets $S$ and $S'$, i.e., $S \triangle S' = (S \setminus S') \cup (S' \setminus S)$. Given a set $S$ and a tuple $S = (S_1, \ldots, S_n)$ of sets, the \textit{distance} between $S$ and $S'$ is defined to be a tuple $d(S, S') = (S \triangle S_1, \ldots, S \triangle S_n)$. For two $n$-element distances $d$ and $d'$, $d \preceq d'$ if $d_i \subseteq d'_i$ for each $1 \leq i \leq n$, where $d_i$ is the $i$-th element in $d$. Given two sets $S$ and $S'$, define $\sigma(S, S') = S$ if $S'$ is empty, and otherwise, $\sigma(S, S') = \{ e_0 \in S \mid \exists e'_0 \in S' \text{ s.t. } \forall e \in S, \forall e' \in S', d(e, e') \neq d(e_0, e'_0) \}$. In \cite{5}, given a collection $\Psi = \{ \varphi_1, \ldots, \varphi_n \}$ of propositional formulas, and some ECs expressed as a propositional theory $\mu$, the result of merging $\varphi_1, \ldots, \varphi_n$ w.r.t. $\mu$ is the theory whose models are exactly $\sigma(\text{mod}(\mu), \text{mod}(\Psi))$, i.e., those models satisfying $\mu$ and having minimal distances to $\Psi$.

Inspired by classical model-based merging, we introduce a merging operator in terms of T-models. For an ontology profile $\Pi$ and an EC $\mathcal{T}_e$, we could define the T-models of the merging result to be a subset of $\text{TM}(\mathcal{T}_e)$ (so that $\mathcal{T}_e$ is entailed) consisting those T-models which have minimal distance to $\Pi$, i.e., $\sigma(\text{TM}(\mathcal{T}_e), \text{TM}(\Pi))$. However, this straightforward adoption does not take the CC into consideration, and the merging result defined in this way may not be coherent. For example, let $\mathcal{T}_1 = \{ A \sqsubseteq \neg B \}$, $\mathcal{T}_2 = \{ \top \sqsubseteq B \}$, $\mathcal{T}_e = \emptyset$, and $\mathcal{A}_c = \{ A(a), B(a) \}$. Then, $\sigma(\text{TM}(\mathcal{T}_e), \text{TM}(\{\mathcal{T}_1, \mathcal{T}_2\}))$ consists of only one type $\{B\}$. Clearly, the corresponding TBox $\{ A \sqsubseteq \bot, \top \sqsubseteq B \}$ does not satisfy the CC, and it is not coherent.

Note that in the above example, once the merging result satisfy the CC, then it is also coherent, because both concepts $A$ and $B$ are satisfiable. In general, it is also the case that coherency can be achieved by applying certain CC to merging. We introduce an auxiliary ABox $\mathcal{A}^1$ in addition to the initial CC $\mathcal{A}_c$, in which each concept and each role is explicitly asserted with a member. That is, $\mathcal{A}^1 = \{ A(a) \mid A \in S_C, a \in S_I \text{ is a fresh individual for } A \} \cup \{ P(b, c) \mid P \in S_R, b, c \in S_I \text{ are fresh individuals for } P \}$. From the definition of CCs, the merging result $\mathcal{T}$ must be consistent with all the assertions in $\mathcal{A}^1$, which assures all the concepts and roles in $\mathcal{T}$ to be satisfiable. The following lemma is based on this observation.

\textit{Lemma 1.} $\mathcal{T}$ is coherent iff $\mathcal{T} \cup \mathcal{A}^1$ is consistent for any TBox $\mathcal{T}$. 
To ensure the coherence of merging result, we only need to include \( A^\dagger \) into the CC. For the merging result to be consistent with the CC \( A_c \), from Proposition 2, the T-model set \( M \) of the merging result needs to satisfy \( A_c \). That is, \( M \) needs to contain a type of \( b \) for each individual \( b \) in \( A_c \). However, \( \sigma(\text{TM}(T_e), \text{TM}(II)) \) does not necessarily satisfy this condition, as can be seen from the above example: \( \text{TM}_b(A_c) \) consists of a single type \{A, B\} and \( \sigma(\text{TM}(T_e), \text{TM}(II)) \cap \text{TM}_b(A_c) = \emptyset \). Intuitively, for the merging result to satisfy the CC, type \{A, B\} need to be added to the T-models of merging result. In general, the T-models of merging result can be obtained by extending (if necessary) the set \( \sigma(\text{TM}(T_e), \text{TM}(II)) \) with at least one type of \( b \) w.r.t. \( A_c \) for each individual \( b \) in \( A_c \), and if there are multiple such types, choose those with minimal distances. Based on the above intuitions, we define TBox merging as follows.

**Definition 2.** Let \( \Pi \) be an ontology profile, \( T_e \) be a TBox, and \( A_c \) be an ABox. Denote \( A^* = A_c \cup A^\dagger \). The result of merging \( \Pi \) w.r.t. the EC \( T_e \) and the CC \( A_c \), denoted \( \nabla(\Pi, T_e, A_c) \), is defined as follows

\[
\text{TM}(\nabla(\Pi, T_e, A_c)) = \sigma(\text{TM}(T_e), \text{TM}(II)) \cup \\
\bigcup_{b \text{ occurs in } A^*} \sigma(\text{TM}(T_e) \cap \text{TM}_b(A^*), \text{TM}(II)).
\]

From the definition, the T-models of the merging result are constituted with two parts. The first part contains those T-models of \( T_e \) with minimal distances to \( II \). The second part are the types of \( b \), for each individual \( b \) in \( A^* \), which are added to the first part for the satisfaction of the CC. These types are also required to be T-models of \( T_e \) and have minimal distances to \( II \). It is clear from Proposition 1 that the result of merging is unique up to TBox equivalence.

### 4 QBF Reduction

In this section, we consider a standard reasoning problem for TBox merging, namely the entailment problem: whether or not the result \( T \) of merging entails a given TBox axiom \( \alpha \). We present a QBF reduction for this problem, which allows us to make use of the off-the-shelf QBF solvers. We assume that every TBox in the TBox profile is coherent, and in this case, the T-models of a TBox \( T \) are exactly those satisfying \( T \).

We achieve this in three steps. Firstly, we introduce a novel propositional transformation for DL-Lite TBoxes. The transformation is inspired by [1], which contains a transformation from a DL-Lite TBox into a theory in the one variable fragment of first order logic. Considering T-models instead of classical DL models allows us to obtain a simpler transformation to propositional logic than theirs to first order logic.
Function $\phi(\cdot)$ maps a basic concept to a propositional variable, and a general concept (resp., a TBox axiom) to a propositional formula.

\[
\begin{align*}
\phi(\bot) &= \bot, \\
\phi(A) &= p_A, \\
\phi(\geq n \, R) &= p_{nR}, \\
\phi(\neg C) &= \neg \phi(C), \\
\phi(C_1 \cap C_2) &= \phi(C_1) \land \phi(C_2), \\
\phi(C_1 \sqcup C_2) &= \phi(C_1) \lor \phi(C_2).
\end{align*}
\]

Here, $p_A$ and $p_{nR}$ are propositional variables. We use $V_S$ to denote the set of propositional variables corresponding to the basic concepts over $S$, and we omit the subscript $S$ in what follows for simplicity.

Naturally, given the mapping $\phi(\cdot)$, an arbitrary propositional model may not correspond to a type. We define a formula $\eta$ whose models are exactly the set of types. Let

\[
\eta = \bigwedge_{R^+ \in S_R} \bigwedge_{m,n \in S_N \text{ with } m < n \text{ and } m < k < n \text{ for no } k \in S_N} p_{nR} \implies p_{mR}.
\]

Intuitively, $\eta$ enforces the second half of the definition of a type to propositional models. More specifically, $\text{mod}(\eta) = \{ \phi(\tau) \mid \tau \text{ is a type}\}$ where $\phi(S)$ stands for for $\{\phi(B) \mid B \in S\}$ for a set $S$ of basic concepts.

Given a coherent DL-Lite TBox $T$, let $\phi(T) = \bigwedge_{\alpha \in T} \phi(\alpha) \land \eta$. The models of $\phi(T)$ correspond to the T-models of $T$. For a DL-Lite ABox $A$ and an individual name $b$ in $A$, let

\[
\phi_b(A) = \bigwedge_{C(b) \in A} \phi(C) \land \bigwedge_{P \in S_R} (p_u \land p_v -)
\]

where $u$ and $v$ are respectively, the maximal number in $S_N$ s.t. $u \leq |\{ c_i \mid P(b, c_i) \in A\}|$ and $v \leq |\{ c_i \mid P(c_i, b) \in A\}|$. Note that we are not transforming an ABox into a propositional theory, but using $\phi_b(A)$ to encode the types of $b$ in $A$.

**Lemma 2.** Given a coherent TBox $T$ and an ABox $A$, then,

1. $\text{mod}(\phi(T)) = \{ \phi(\tau) \mid \tau \in \text{TM}(T)\}$;
2. $\text{mod}(\phi_b(A) \land \eta) = \{ \phi(\tau) \mid \tau \in \text{TM}_b(A)\}$.

Secondly, as we have a transformation from T-models to propositional models, we can encode (minimal) distances between them using QBFs, by extending the encoding in \cite{[3]}, which was introduced for a different purpose. In particular, we need to encode the distances between the models of $\phi$ and the models of $\varphi_1, \ldots, \varphi_n$ ($n \geq 1$), where $\phi$ and $\varphi_i$’s are propositional formulas in signature $V$.

We make $n$-fresh copies of $V$ to, informally, encode the models of $\varphi_1, \ldots, \varphi_n$, respectively; let $V^i = \{ p^i_j \mid p \in V \}$ for $1 \leq i \leq n$ where each $p^i$ is a fresh variable for $p$, and $V^N = \bigcup_{1 \leq i \leq n} V^i$. For a propositional formula $\varphi$ and $1 \leq i \leq n$, $\varphi^i$ denotes the formula obtained from $\varphi$ by replacing each occurrence of $p$ with $p^i$.

We also need another $n$-fresh copies of $V$, $V^d = \{ p^d_j \mid p \in V \}$ to represent the distances. An assignment to $V^d$ is expected to capture the difference between a
A model of \(\phi\) and a model of \(\varphi_i\), and in particular, \(p^i_d\) is assigned true if the truth values of \(p\) and \(p^i\) are different. Let \(V_d^N = \bigcup_{1 \leq i \leq n} V_d^i\). Define

\[
F(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle) = \phi \land \bigwedge_{1 \leq i \leq n} \left( \varphi^i_1 \land \bigwedge_{p \in V} \left( p \leftrightarrow \neg p^i \right) \right).
\]

A model \(M\) of \(F(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)\) consists of the assignments to three sets of variables \(V, V^N\) and \(V^N_d\). For a set \(S \subseteq V \cup V^N \cup V^N_d\), \(m(S)\) is the set obtained from \(S\) by eliminating the super- and subscripts. Then, \(M \cap v\) is a model of \(\phi\) and \(m(M \cap v^i)\) is a model of \(\varphi_i\). From \((p \leftrightarrow \neg p^i) \rightarrow p^i_d\), \(m(M \cap v^i)\) subsumes the symmetric difference between the former two models. We use \(\rightarrow\) instead of \(\leftrightarrow\) here, as we will further constraint the assignments of \(v^N_d\) to minimal distances.

Furthermore, define a QBF

\[
D(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle) = (\exists V \exists V^N F(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)) \land \\
\bigwedge_{1 \leq i \leq n} \left( p^i_d \rightarrow \neg \exists V \exists V^N \left( F(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle) \land (p \leftrightarrow p^i) \right) \right),
\]

where \(\exists V\) with \(V = \{p_1, \ldots, p_k\}\) is an abbreviation for \(\exists p_1 \ldots \exists p_k\). A model \(M_d\) of \(D(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)\) is an assignment to \(V^N_d\) representing a minimal distance between the models of \(\phi\) and the model tuples of \(\langle \varphi_1, \ldots, \varphi_n \rangle\). The first conjunct of \(D(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)\) says that there is a model of \(\phi\) and a model tuple of \(\langle \varphi_1, \ldots, \varphi_n \rangle\) such that the distance between them is subsumed by \(m(M_d)\). The second conjunct checks that \(m(M_d)\) is minimal, i.e., there is no distance properly subsumed by \(m(M_d)\).

**Lemma 3.** Given propositional formulas \(\phi\) and \(\varphi_1, \ldots, \varphi_n\), let \(MD(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)\) be the set of minimal distances (w.r.t. \(\leq\)) between \(\phi\) and \(\langle \varphi_1, \ldots, \varphi_n \rangle\) (of the form \(\langle M \triangle M_1, \ldots, M \triangle M_n \rangle\) with \(M \in \text{mod}(\phi)\) and \(M_i \in \text{mod}(\varphi_i)\)). Then,

\[
\text{mod}(D(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle)) = \{ M_d \subseteq V^N_d \mid \exists d \in Md(\phi, \langle \varphi_1, \ldots, \varphi_n \rangle) \text{ s.t. } m(M_d \cap V^i_d) = d \text{ for each } 1 \leq i \leq n \}.
\]

Finally, with the encoding of minimal distances, we can encode the T-models of the merging and we are ready to encode the entailment relation. Given an ontology profile \(\Pi = \langle T_1, \ldots, T_n \rangle\) and a TBox \(T_c\), all the types in \(\sigma(\text{TM}(T_c), \text{TM}(\Pi))\) satisfy TBox axiom \(\alpha\) iff QBF \(\neg \exists V^N \left( E(\Pi, T_c, \alpha) \right)\) evaluates to true, with

\[
E(\Pi, T_c, \alpha) = D(\phi(T_c), \langle \phi(T_1), \ldots, \phi(T_n) \rangle) \land \\
\neg \forall V \left( (\exists V^N F(\phi(T_c), \langle \phi(T_1), \ldots, \phi(T_n) \rangle)) \rightarrow \phi(\alpha) \right).
\]

This QBF can be understood as follows. A model \(M\) of \(E(\Pi, T_c, \alpha)\) is an assignment to \(V^N_d\), and represents, by the first conjunct of \(E(\Pi, T_c, \alpha)\), a minimal distance between the T-models of \(T_c\) and the T-model tuples of \(\Pi\). The second conjunct states the non-entailment, that is, not all of the T-models of \(T_c\) having such a distance satisfy \(\alpha\). The QBF as a whole essentially says that there
does not exist a minimal distance \( d \) such that a type (in \( \sigma(TM(T_e), TM(II)) \)) selected with \( d \) fails to satisfy \( \alpha \). Similarly, given an ABox \( A \) and an individual \( b \) in \( A \), all the types in \( \sigma(TM(T_e) \cap TM_b(A), TM(II)) \) satisfy \( \alpha \) iff QBF

\[
\neg \exists V_d^N E_b(II, T_e, A, \alpha)
\]

where \( E_b(II, T_e, A, \alpha) \) is obtained from \( E(II, T_e, \alpha) \) by replacing \( \phi(T_e) \) with \( \phi(T_e) \wedge \phi_b(A) \).

Now, we can reduce the entailment problem for TBox merging to QBF as follows.

**Theorem 1.** Let \( II \) be an ontology profile with coherent source TBoxes, \( T_e \) be a TBox and \( A_c \) be an ABox in DL-Lite\( ^N \)bool. Let \( A^* = A_c \cup A^I \). Given a TBox axiom \( \alpha \), we have \( \nabla(II, T_e, A_c) \models \alpha \) iff the following QBF evaluates to true

\[
\neg \exists V_d^N (E(II, T_e, \alpha) \lor \bigvee_{b \text{ occurs in } A^*} E_b(II, T_e, A^*, \alpha))
\]

(1)

5 System Architecture

We have implemented the algorithm for checking entailment in ontology merging, called OntoMerge, which is publicly available for test at [http://www.ict.griffith.edu.au/~kewen/OntoMerge/](http://www.ict.griffith.edu.au/~kewen/OntoMerge/). The ultimate goal of our system OntoMerge is to transform the entailment over merged ontologies to the validity of QBFs as given in Eq.(1). If the input ontologies are \( T_1, \ldots, T_n \) and the input axiom is \( \alpha \), then the corresponding QBF can be split into two parts: one is the formula \( E(II, T_e, \alpha) \), and the other is a disjunction of formulas of the form \( E_b(II, T_e, A^*, \alpha) \). The size of the first part is only determined by the sizes of the input ontologies, the input EC and the axiom \( \alpha \), and the size of \( \alpha \) is often small compared to the ontologies. In the second part of the resulting QBF, the number of disjuncts is essentially determined by the number of unsatisfiable concepts in the union of input ontologies (as will be explained later) and the input CC. Thus, the number of unsatisfiable concepts plays a crucial role in the complexity of the reduction algorithm.

In OntoMerge, we first check whether the given ontologies can be simply joined without causing any incoherency using an off-the-shelf DL reasoner (HermiT [9] is used in the current program). The set of unsatisfiable concepts will be stored for being used later. If the union of input ontologies is coherent, the entailment checking can be done by the DL reasoner; otherwise, the system generates the QBF specified in Eq.(1). For this purpose, the system will first scan all input ontologies to obtain the set of basic concepts occurring in these ontologies and assign a propositional variable with each basic concept. Then the QBF is generated in QBF 1.0 format. The structure of OntoMerge is depicted in Figure 1.

However, as most of efficient QBF solvers accepts only input QBFs in the prenex normal form, the QBF generated in this way cannot be directly fed to a QBF solver. So we need to convert the QBF into the prenex normal form first. Unfortunately, the standard translation from a given QBF into its prenex normal

\[\text{http://qbflib.org/boole.html}\]
form is very inefficient and thus several heuristics based on the specific QBF are used to optimize the efficiency in our implementation. For instance, from Eq. (1), we can see that the major source for introducing a large number of new variables is from the construction of the ABox $A^*$. So we introduce new variables for only those concepts that are unsatisfiable and add new assertions for such new variables to $A^*$ in the reduction algorithm. This optimization significantly reduces the number of new variables.

Once the QBF is generated, a publicly available program is used to convert it from QBF 1.0 format to QDIMACS format or ISCAS format and then an efficient QBF solver is used to decide the validity of the QBF.

6 Experimental Results

To test the efficiency of OntoMerge, we used the DL-Lite$^\forall$ $^{[\ref{6}]}$ fragment of the medical ontology Galen $^{[\ref{8}]}$, which is of medium size. Our experimental results show that the system is relatively efficient, while further optimization is still under way. Specifically, various randomly modified fragments of Galen were merged using OntoMerge and the following three types of experiments were conducted:

6 http://www.qbflib.org/qdimacs.html
7 http://logic.pdmi.ras.ru/~basolver/rtl.html
8 http://www.co-ode.org/galen/
– Fixed total number of axioms in the input ontologies but varied number of unsatisfiable concepts.
– Fixed total number of unsatisfiable concepts but varied total number of axioms in the input ontologies.
– Fixed number of unsatisfiable concepts but varied total number of input ontology axioms.

A PC with Intel Core 2 Duo E8400, 4GB RAM, running Linux Mint 13 64bit, and CirQit QBF solver [7] were used in our tests. For each test, the time is limited to one hour (i.e. the program will be terminated after one hour no matter a result is returned or not).

![Figure 2](image)

**Fig. 2.** Total axioms:42, unsatisfiable concepts from 1 to 18

Figure 2 shows the experimental results in the first set of tests. In each test, 42 axioms were randomly selected from Galen ontology and they were separated into two sub-ontologies for merging. Then assertions were inserted into one of them so that some concepts became unsatisfiable in the union of these two ontologies. The number of unsatisfiable concepts varied from 1 to 18. From this figure we can see that the program is quite fast according to the time used for generating the QBF and the time used for deciding the validity of the QBF. This is because when the number of unsatisfiable concepts is increased, the size of the QBF generated is increased linearly. However, when the number of unsatisfiable concepts is over 18, OntoMerge can still generate the QBF but the QBF solver is unable to decide the validity of such a QBF.

In the second set of tests, we fixed the number of unsatisfiable concepts to 1 but increased the total number of axioms from 26 to 174. Figure 3 shows that when the number of axioms is increased, the time cost for generating QBF increases faster than in the first set of tests. This is partly due to the fact that with the increase of the total axioms, the size of the QBF significantly increases too. Similar to the case for the first set of tests, when the total number of axioms is over 174, the QBF solver is failed again.

In the third set of tests, we fix the ratio of total number of axioms to the number of unsatisfiable concepts to around 0.15 but let the total number of axioms varied from 26 to 92 as well as the number of unsatisfiable concepts.
Fig. 4. Ratio of unsatisfiable concepts to total axioms = 0.15

Figure 4 shows a similar pattern to Figure 3. When the total number of axioms is over 92, the QBF solver failed to return an answer.

7 Conclusion

We have developed a new semantic approach for merging ontologies in DL-Lite\textsuperscript{N} instead of using classical DL models, we used types as an alternative semantic characterisations for TBoxes, and defined a TBox merging operator for DL-Lite. We presented algorithms to reduce standard reasoning problems of DL-Lite merging to the evaluation of QBFs and thus provided a novel way of reasoning with the result of ontology merging using efficient QBF solvers. We have implemented a preliminary merging system OntoMerge, and reported some experimental results in the paper. Currently, we are extending the approach in two directions: (1) merging DL-Lite ontologies with both TBoxes and ABoxes, and (2) merging ontologies in expressive DLs.

Acknowledgement: We would like to thank the three anonymous referees for their helpful comments. Special thanks to Rodney Topor for various discussions with him. This project was partially supported by the ARC grants DP1093652 and DP110101042. Guilin Qi was partially supported by the NSFC grant 61272378.
References