Tree Edit Distance Based Ontology Merging Evaluation Framework

Zied Bouraoui, Sébastien Konieczny, Thanh Ma, and Ivan Varzinczak

CRIL, Artois University & CNRS, Lens, France
{bouraoui,konieczny,ma,varzinczak}@cril.fr

Abstract. Merging structured knowledge has been widely investigated to build common resources in recent years. Indeed, many merging operators have been proposed and developed. However, the majority of them lack comparison and evaluation. Finding ontology sources for evaluation is not an effortless task. To this end, we propose a framework for evaluating the quality of ontology merging operators. The primary strategy starts with an original ontology as a gold standard to create noisy ontologies as datasets and use them to evaluate the merging operators. We generate the noisy ontologies using some perturbations of the tree structure of the original ontology based on tree edit operations. Then, we use tree edit distance to measure the existing merging operators with these noisy sources. We provide the details to assess the merging operators' efficiency in the computation time and their ability to cover (or be close to) the original ontology.

Keywords: Evaluation· Noisy Ontology · Belief Merging · Tree Edit Distance

1 Introduction

Structured knowledge relating to concepts and properties is widely employed in a variety of domains, including natural language processing (NLP) [17], information retrieval (IR) [9], and semantic web [12]. They are generally encoded beneath the backbone of tree structures or knowledge graphs (i.e., a hierarchy of ontology). The main difference is that knowledge graphs are less expressive than ontologies [13]. As such, it might be considered as a simplified variant of ontologies[1].

In the context of ontologies (structured knowledge), researchers and practitioners have to contend with the difficulties of various fresh information emerging from multiple sources. Since these knowledge sources are given by different agents (i.e., ontologies), merging many diverse sources might result in inconsistencies and conflicts. From this point, ontology merging has been widely studied in the last years aimed at obtaining the common consistent source. Ontology merging and alignment have attracted much attention in the literature [20,8,3]. Ontology merging aims to combine two (or more) ontologies having *the same terminology* when handling conflict. In contrast, ontology alignment (or matching) is the process of determining correspondences between terminologies of ontologies. Otherwise, the problem of ontology (or DL) merging is close to the problem of belief merging in a propositional setting [2,21] and several concrete

[1] In this paper, we use ontologies to refer to knowledge graphs.

merging operators have been proposed [15,14]. For instance, Benferhat et al. [3] studied merging assertional bases in *DL-Lite* fragment. They have determined the minimal subsets of assertions to resolve conflicts based on the inconsistency minimization principle. Wang et al. [21] provided an ontology merging operator. Their operators based on a new semantic characterisation. Namely, the minimality of changes is realised via a model (semantic) distance.

In general, numerous merging operators have been proposed and developed in the last years. The postulates are actually used to evaluate the behaviours of operators. However, to the best of our knowledge, there is no evaluation framework to assess and compare the quality of different merging operators. For evaluating operators, this work is motivated by two primary reasons: (1) *evaluating the merging models for inputs from multiple fields (open-domain) is a difficult challenge*. Namely, even-though the open-domain concept names are same, their meanings differ. i.e., $Rock\ is\text{-}a\ Stone$ while $Rock\ is\text{-}a\ Music$ (in the music domain). Naturally, when merging these sources, there will be no meaningful relationship between $Stone$ and $Music$. (2) *Another challenge is to evaluate the merging models that require the same terminologies (signatures)*. In fact, ontologies are naturally constructed based on the differing views of their builder. Therefore, even if they are the same field, the concept names are also different. i.e., $Vacation$ and $Holiday$. At these points, *finding ontology sources for the model evaluation is not an effortless task*. To solve them, noisy sources generated from an initial source are a potential solution for measuring the efficacy of ontology merging operators. In this perspective, we take inspiration Tree Edit Distance (TED)[2] [23,18,22] to create noise trees and measure a distance between these attributed trees (or graphs). Here, the distance has defined as the minimum amount of edit operations *(deletion, insertion, and substitution of nodes and edges)* needed to transform a tree into another. We generate noisy ontologes (NoiOn(s)) via the edit operation.

Taking account of the foregoing, we propose an ontology merging evaluation framework to assess the quality of the ontology merging operators. The idea is to start with an ontology (gold standard), then build some NoiOn(s) (a merging profile), apply the merging operators to them, and compare the results to the gold standard. Particularly, we build the NoiOn(s) generation with "local" perturbation to make sense because neighboring concepts can be related to each other. Hence, we implement the modification of the substructure with a node's parent, child, and siblings to obtain the NoiOn. To this end, we provide an algorithm. Then, we use the existing merging procedures to obtain the merged ontologies. We compare the merged result and the original one to evaluate the operators. Considering the ability of noise reduction after merging will be a key to assess the operators. Note that the ontology is represented as a tree (graph). The generation of NoiOn(s) and the merging of ontologies are two independent processes.

2 Background

Our approach implements on two foundations: (1) we rely on a lightweight Description Logic (DL) framework to encode terminological Boxes of ontologies, (2) we use edit operations and TED for creating NoiOn(s).

[2] http://tree-edit-distance.dbresearch.uni-salzburg.at

2.1 Description Logics

$\mathcal{EL}$ is a family of lightweight DLs, which underlies the Ontology Web Language profile OWL2-EL, that is considered as one of the main representation formalisms to express terminological knowledge [1].

The main ingredients of DLs are individuals, concepts, and role. More formally, let N_C, N_R, N_I be three pairwise disjoint sets where N_C, N_R, N_I denote a set of atomic concepts, atomic relations (roles), and individuals, respectively. In this paper, we consider $\mathcal{EL}_\bot$ concept expressions [11] which are built according to the grammar: $C ::= \quad \top \mid \bot \mid N_C \mid C \sqcap C \mid \exists r.C$ where $r \in N_R$. Let $C, D \in N_C$, $a, b \in N_I$, and $r \in N_R$. An $\mathcal{EL}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ (a.k.a. knowledge base) comprises two components, the TBox (Terminological Box denoted by $\mathcal{T}$) and ABox (denoted by $\mathcal{A}$). The TBox consists of a set of General Concept Inclusion (GCI) axioms of the form $C \sqsubseteq D$, meaning that C is more specific than D or simply C is subsumed by D, and axioms of the form $C \sqcap D \sqsubseteq \bot$, meaning that C and D are disjoint concepts. The ABox is a finite set of assertions on individual objects of the form $C(a)$ or $r(a, b)$.

The semantics is given in terms of interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, which consist of a non-empty interpretation domain $\Delta^\mathcal{I}$ and an interpretation function $\cdot^\mathcal{I}$ that maps each individual $a \in N_I$ into an element $a^\mathcal{I} \in \Delta^\mathcal{I}$, each concept $A \in N_C$ into a subset $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and each role $r \in N_R$ into a subset $r^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, each axiom $C \sqsubseteq D$ into $C^\mathcal{I} \subseteq D^\mathcal{I}$, each $C \sqcap D$ into a subset $C^\mathcal{I} \cap D^\mathcal{I} \in \Delta^\mathcal{I}$, a top concept $\top$ into $\Delta^\mathcal{I}$, and the bottom concept $\bot$ into the empty set $\emptyset$. An interpretation $\mathcal{I}$ is said to be a model of (or satisfies) an axiom Φ, denoted by $\mathcal{I} \models \Phi$. For instance, $\mathcal{I} \models C \sqsubseteq D$ if and only if $C^\mathcal{I} \subseteq D^\mathcal{I}$. Similarly, $\mathcal{I}$ satisfies a concept (resp. role) assertion, denoted by $\mathcal{I} \models C(a)$ (resp. $\mathcal{I} \models r(a, b)$), if $a^\mathcal{I} \in C^\mathcal{I}$ (resp. $(a^\mathcal{I}, b^\mathcal{I}) \in r^\mathcal{I}$). In this paper, we assume that the input ontologies are provided in a specific normal form, which we apply completion rules (see [1] for more details) for classification. Note that the classification process works before the merging operation such that all axioms are in the normal form. We denote a set of ontologies is a profile. This profile is able to use for ontology merging.

2.2 Edit Operation and Tree Edit Distance

We here provide several formal definitions related to the TED to use in the sequels. A rooted tree, denoted by $\mathscr{T}$, is a connected graph with nodes $V(\mathscr{T})$ and edges $E(\mathscr{T}) \subseteq V(\mathscr{T}) \times V(\mathscr{T})$. The root of $\mathscr{T}$ is denoted by $\Re(\mathscr{T})$. Here, we write $\mathscr{T}$ to represent the set of nodes of $\mathscr{T}$ (replacing $\mathscr{T}$ by $V(\mathscr{T})$). For two nodes $v_1, v_2 \in \mathscr{T}$, a parent of v_1, denoted by $p(v_1)$, is the closest ancestor of A. We denote ϑ as a finite alphabet and $lb_\mathscr{T} : \mathscr{T} \to \vartheta$ as a labelling function.

The tree edit distance [23,18,22] between two trees $\mathscr{T}_1$ and $\mathscr{T}_2$ is defined as the minimum cost of edit operations to transform a tree to another. In order to implement edit operations, we denote ϵ as a blank symbol and $\vartheta_\epsilon = \vartheta \cup \{\epsilon\}$ to represent the edit operations. Here, we denote each of the edit operations by **a pair** denoted as $\vartheta_\epsilon \times \vartheta_\epsilon \setminus \{(\epsilon, \epsilon)\}$. We now define edit operations.

Definition 1 (Edit operations). *Let $\mathscr{T}$ be a tree and $v_1, v_2 \in \mathscr{T}$. Edit operations on $\mathscr{T}$ include: Replacing a node labeled v_1 by other node labeled v_2 in $\mathscr{T}$ is denoted by*

$Rep(v_1, v_2, \mathscr{T})$; Deleting a node labeled v_1 in $\mathcal{T}$ such that the children of v_1 will be the children of $p(v_1)$ is denoted by $Del(v_1, \epsilon, \mathscr{T})$; Inserting a node labeled v_1 into $\mathscr{T}$ such that v_1 is a child of v_2 is denoted by $Ins(\epsilon, v_1, v_2, \mathscr{T})$.

Now, let us denote a cost function on edit operations by $dist : \vartheta \times \vartheta \backslash \{(\epsilon, \epsilon)\} \rightarrow \mathbb{R}^+$. Notice that we simply write $dist(v_1, v_2)$ to represent $dist(lb(v_1), lb_2(v_2))$, where lb_1 and lb_2 are labelling functions on two trees $\mathscr{T}_1$ and $\mathscr{T}_2$. *Normally, the edit operation cost of each operation is* 1, i.e., $dist(v_1, v_2) = 1$ or $dist(v_1, \epsilon) = 1$. However, we may also re-define this cost (see [4] for more details). Notice that we will use the cost of each operation equals to 1 for this whole work. Moreover, an edit mapping is a description of how a sequence of edit operations transforms from $\mathscr{T}_1$ into $\mathscr{T}_2$. Now, we present a mapping between two trees for computing the distance between them. An edit mapping between $\mathscr{T}_1$ and $\mathscr{T}_2$, denoted by $M^{\mathscr{T}_1}_{\mathscr{T}_2}$, is a subset of $\mathscr{T}_1 \times \mathscr{T}_2$ (a.k.a. $M^{\mathscr{T}_1}_{\mathscr{T}_2} \subseteq V(\mathscr{T}_1) \times V(\mathscr{T}_2)$). A pair $(v_1, v_2) \in M^{\mathscr{T}_1}_{\mathscr{T}_2}$ is called a node alignment between $v_1 \in \mathscr{T}_1$ and $v_2 \in \mathscr{T}_2$. The set of all mappings between $\mathscr{T}_1$ and $\mathscr{T}_2$ is denoted by $\mathcal{M}(\mathscr{T}_1, \mathscr{T}_2)$. The cost of computing a mapping between the two trees is defined as:

Definition 2. *Let $\mathscr{T}_1$ and $\mathscr{T}_2$ be two trees, $M^{\mathscr{T}_1}_{\mathscr{T}_2}$ be a mapping between them, and $(v_1, v_2) \in M^{\mathscr{T}_1}_{\mathscr{T}_2}$. The cost of a mapping between $\mathscr{T}_1$ and $\mathscr{T}_2$ is defined as follows:* $dist(M^{\mathscr{T}_1}_{\mathscr{T}_2}) = \sum_{(v_1,v_2)\in M^{\mathscr{T}_1}_{\mathscr{T}_2}} dist(v_1, v_2) + \sum_{v_1 \in \mathscr{T}_1} dist(v_1, \epsilon) + \sum_{v_2 \in \mathscr{T}_2} dist(\epsilon, v_2)$.

Intuitively, the cost of mapping is a total of the cost of all edit operations with a mapping $M^{\mathscr{T}_1}_{\mathscr{T}_2}$ to transform $\mathscr{T}_1$ into $\mathscr{T}_2$. Now, we present how to compute the edit tree distance between two trees referring to [19,22] as follows:

Definition 3 (Tree Edit Distance). *Let $\mathscr{T}_1$ and $\mathscr{T}_2$ be two trees. The edit distance between $\mathscr{T}_1$ and $\mathscr{T}_2$ is defined as:* $dist(\mathscr{T}_1, \mathscr{T}_2) = \min\{dist(M^{\mathscr{T}_1}_{\mathscr{T}_2}) \mid M^{\mathscr{T}_1}_{\mathscr{T}_2} \in \mathcal{M}(\mathscr{T}_1, \mathscr{T}_2)\}$.

3 Ontology Merging Evaluation Framework

We propose an evaluation framework to assess the quality of the existing ontology merging operators including generating and merging NoiOn(s). We first establish a profile with many different NoiOn(s) from a single ontology source, then merge them using the merging operator. A noisy ontology builds on the local perturbations. Namely, our framework includes six steps S_i.

S_1 - Given an input ontology, we randomly collect the concepts' number ($rP\%$). These selected concepts are the signatures to build NoiOn(s). S_2 - For each selected concept (from S_1), we filter the local relatives *including Children, Fathers, Siblings*. This process is the collection of concept's nearest neighbors. We focus on the surrounding neighbourhoods of concepts since the close concepts can be interconnected. S_3 - After collecting the concepts and their neighbors, we use the edit operations [23] to create NoiOn(s). Note that we do not generate NoiOn(s) from all nodes that only concentrate on the ($rP\%$) random nodes with their relatives. Here, we call *a local perturbation*. In particular, we re-structure the hierarchy of ontology by two methods: (1) $[DI]$ Delete a node (or a concept) and insert it into an arbitrary position in the ontology hierarchy; (2) $[Sw]$ Swap between two nodes in the ontology hierarchy. S_4 - We here select n-top

Algorithm 1: Generating Noisy Ontologies

input: rP: A Random Percentage of Concepts, n: Number of Noisy Trees, t: Threshold, $\mathcal{O}$: An OriOn
output: $\mathcal{P}$: A Set of Noisy Ontologies

begin
  $\mathscr{T} \longleftarrow \mathbb{H}(\mathcal{O}), \mathcal{P} \longleftarrow \emptyset$
  while $|\mathcal{P}| \leq n$ **do**
    // Randomly collect $r\%$ concepts from ontologies
    $\alpha \longleftarrow \Re_C(\mathcal{O}, rP)$
    // Collect the neighbourhood concepts of the concepts in α
    $V \longleftarrow \boxplus_{Neighbours}(\mathcal{O}, \alpha)$
    // Extract the hierarchical structure of ontology
    $\mathscr{T}^N \longleftarrow \mathbb{H}(\mathcal{O})$
    foreach $(v_C, v_D) \in V \times V$ **do**
      // Randomly select one edit operation for a pair
      // True: use $[DI]$ method, False: use $[Sw]$ method
      if $\Re_M()$ *is* $True$ **then**
        $\mathscr{T}' \longleftarrow Del(v_C, \epsilon, \mathscr{T}^N)$
        $\mathscr{T}^N \longleftarrow Ins(\epsilon, v_C, v_D, \mathscr{T}')$
      else
        $\mathscr{T}^N \longleftarrow Rep(v_C, v_D, \mathscr{T}^N)$
    if $0 < dist(\mathscr{T}^N, \mathscr{T}) \leq t$ *and* $\mathscr{T}^N \notin \mathcal{P}$ **then**
      $\mathcal{P} \longleftarrow \mathcal{P} \cup \mathscr{T}^N$
  return $\mathcal{P}$

NoiOn(s) close to the original one (based on the TED) to create a profile for merging. To this end, we compute the distances between an OriOn and NoiOn(s). Moreover, we use a threshold t to detect the close NoiOn(s) and n to limit the number of NoiOn(s). S_5 - We take advantage of the existing ontology merging method for the framework. Noteworthy, if we only merge the NoiOn(s), the merging result can lose some axioms of the OriOn. Hence, we add m input ontologies into the profile to guarantee that the merging outcome cover fully the original one. Then, a profile includes m OriOn(s) and k NoiOn(s) called *a hybrid profile*. S_6 - After the merged result obtains, we compute the distance between the merged result and the OriOn to measure the quality of merging operators. For this work, we evaluate the two above operators ([6] and [7]).

In the sequel, we explicitly present how to collect NoiOn(s) based on the edit operations to create a profile for merging.

4 Building Ontology Profile

To begin, an ontology structure (tree) is denoted by $\mathscr{T}$. Here, the input of this process is an ontology, and the output is a set of NoiOn(s). Before collecting NoiOn(s), we define $\mathscr{T} \stackrel{\text{def}}{=} \mathbb{H}(\mathcal{O})$, where $\mathcal{O}$ is an ontology and $\mathbb{H}(\mathcal{O}) = \{A \sqsubseteq B \in \mathcal{O} \mid \forall A, B \in N_C\}$ is a function to structure a hierarchical tree. We here take an account of axioms of the form $A \sqsubseteq B$ since one of the most common forms to build the ontology's hierarchy [16,5]. If concepts do not have a father, the "Thing" concept ($\top$) will be assigned as their father.

Collection of Noisy Ontologies We now provide how to collect NoiOn(s) from an input ontology. The idea is to select some concept pairs and edit them. Formally, we represent how to collect the NoiOn(s) by Algorithm 1.

We call Algorithm 1 as $\mathbb{G}(rP, n, t, \mathcal{O})$. The algorithm works as follows: we randomly collect concepts ($rP\%$) from OriOn(s) (line 4). e.g., $rP = 5$ and $n = 80$, then we have $5\% \times 80$ $concepts$ $= 4$ $concepts$. We here define $\alpha = \Re_C(\mathcal{O}, rP) \stackrel{\text{def}}{=} \{C \in \sigma \mid \sigma \subseteq N_C, |\sigma| < nr\}$ where $nr = \frac{rP \times |N_C|}{100}$. Next, we collect the relative nodes in α (Children, Father, Sibling) at line 5. Given $A \in \alpha$, the concepts related to A denote as A's neighbors. We define $\boxplus_{Neighbours}(\mathcal{O}, \alpha) = \{g(\mathcal{O}, A) \mid A \in \alpha\}$ where $g = \{NC, NF, FC, NS\}$. Therein, $NC(\mathcal{O}, A) = \{(A, B) \mid B \in \mathcal{O}, B \text{ is a Child of } A\}$, $NF(\mathcal{O}, A) = \{(B, A) \mid B \in \mathcal{O}, B \text{ is a Father of } A\}$, $FC(\mathcal{O}, A) = \{(B, A) \mid B \in \mathcal{O}, B \text{ is a close ancestor of } A\}$, $NS(\mathcal{O}, A) = \{(A, B) \mid B \in \mathcal{O}, B \text{ is a sibling of } A\}$. From each node pair in $\boxplus_{Neighbours}$, we select randomly one method to modify the ontology, including the Deletion-Insertion operation ($[DI]$) (using $Del(v_C, \epsilon, \mathscr{T}^N)$ and $Ins(\epsilon, v_C, v_D, \mathscr{T}')$ (from line 9 to 10)) and the swap operation ($[Sw]$) (using replacement operation $Rep(v_C, v_D, \mathscr{T}^N)$ (at line 12)). Here, we define $\Re_M()$ is *a binary random function* (return True or False) to select either $[DI]$ or $[Sw]$. We denote $\mathcal{P}$ as a set of NoiOn(s) or a profile. After the NoiOn obtains, we compute the distance between the NoiOn $\mathscr{T}^N$ and the OriOn $\mathscr{T}$ to measure their closeness. A NoiOn is collected into $\mathcal{P}$ ($\mathcal{P} \longleftarrow \mathcal{P} \cup \mathscr{T}^N$) if the distance is less than or equal to a threshold t ($dist(\mathscr{T}^N, \mathscr{T}) \leq t$) at line 13 and 14. We choose the acceptable threshold based on the number of concepts. The goal of threshold is to find a NoiOn close to the original one. Finally, the number of NoiOn(s) depends on the parameter n at line 3 with the condition $|\mathcal{P}| \leq n$. This "$while$" loop works until the number of NoiOn(s) is fully collected. Note that the collection process of NoiOn(s) from line 3 to 14 runs in parallel to improve the computation time. Formally, a set of NoiOn(s) is defined as $\mathcal{P} \stackrel{\text{def}}{=} \mathbb{G}(rP, n, t, \mathcal{O})$. Here, Algorithm 1 runs in a time that is polynomial on the number n of NoiOn(s), given access to an NP oracle in one step (line 3). Indeed, (i) the size of $\mathcal{P}$ depends on threshold t (line 13), (ii), the function Del, Ins, Rep, and $\Re$ are computed in $O(1)$. (iii) the iteration's number from line 7 is in $O(|V \times V|)$. The algorithm complexity is in FP^{NP}.

Collecting a merging profile After collecting the n NoiOn(s), we use the TED to filter the k NoiOn(s) that are closest to the OriOn. To this end, we say $\preceq_{\mathcal{P}}$ is a pre-order[3] on $\mathcal{P}$ such that $\forall \mathscr{T}_1^N, \mathscr{T}_2^N \in \mathcal{P}: \ \mathscr{T}_1^N \preceq_{\mathcal{P}} \mathscr{T}_2^N$ iff $dist(\mathscr{T}_1^N, \mathscr{T}) \leq dist(\mathscr{T}_2^N, \mathscr{T})$ where $\mathscr{T}$ is a hierarchical structure of the OriOn $\mathcal{O}$. Note that, the noisy trees in $\mathcal{P}$ are always distinct since the condition $\mathscr{T}^N \notin \mathcal{P}$ at line 13 (Algorithm 1) is implemented.

Definition 4. *Let $\mathscr{T}$ be an ontology tree and $\preceq_{\mathcal{P}} = \{\mathscr{T}_1^N, \ldots, \mathscr{T}_n^N\}$ be a pre-ordered set of all noisy ontology trees. The set of "top-k" NoiOn trees close to the OriOn is defined as follows: $\mathcal{P}_{Top}^k \stackrel{\text{def}}{=} \{\mathscr{T}_i^N \in \preceq_{\mathcal{P}} \mid 1 \leq i \leq k < n\}$.*

Here, $\mathcal{P}_{Top}^k$ is a profile with k NoiOn(s). e.g., a profile with three noisy trees is $\mathcal{P}_{Top}^3$. As explained in S_5, because noisy trees are generated using randomization, in the worst-case scenario, an axiom of the original tree is not existing in all noisy trees. Therefore, it leads to no source stating that constraint. As a result, they may lose some of the axioms in OriOn. This is why we add some original trees to the profile to ensure that the merged result covers the OriOn. Formally, an extended profile is defined as:

[3] A preorder is a reflexive and transitive relation.

Definition 5. *Let $\mathcal{P}^k_{Top}$ be a set of "k" NoiOn trees and $\mathscr{T}$ be an OriOn tree. A "hybrid" profile extended by adding "m" original trees is defined as follows:* $\mathcal{P}^{k,m} \stackrel{\text{def}}{=} \mathcal{P}^k_{Top} \cup \{\mathscr{T}_i = \mathscr{T} \mid 1 \leq i \leq m\}$.

Note that we denote $\mathcal{P}^{k,m} = \{\mathscr{T}_i\}$ corresponding to $\mathcal{P}^{k,m} = \{\mathcal{O}_i\}$, where $\mathscr{T}_i$ are hierarchical structures of $\mathcal{O}_i$. Now we apply the merging methods to the profile created.

5 Ontology Merging Operators

This section provides some existing merging operators designed for the $\mathcal{EL}$ intending to evaluate them. Namely, two merging operators investigate in this paper, including (1) the model-based ontology merging framework of [6] and (2) the ontology merging method via merging the *Qualitative Constraint Network (QCN)* of [7]. Now, a brief description of the merging frameworks is as follows:

For the (1) merging method, this approach implements with DL $\mathcal{EL}$ to encode the knowledge. Regarding this setting, There are no logical contradictions in ontology because *negative (or bottom) notions* do not exist in them.. This method crucially focuses on handling the semantic conflicts and objects to model-based merging. Here, we denote $\mathbb{M}_{MBM}$ *as a "model-based merging" function* [6]. Regarding the (2) merging method, Bouraoui et al. has proposed an ontology merging method via merging the QCNs. Their merging procedure focuses on the DL $\mathcal{EL}_\bot$. Their approach allows us to benefit from the expressivity and the flexibility of RCC5 while dealing with conflicting knowledge in a principled way. From this approach, we investigate the subsumption ($A \sqsubseteq B$) and disjoints ($A \sqcap B \sqsubseteq \bot$) for our NoiOn(s) merging work. Formally, we denote $\mathbb{M}_{QM}$ *as a "QCN merging" function* [7].

Let us denote $\mathcal{O}^M$ as the NoiOn merged result (using $\mathbb{M}_{MBM}$ and $\mathbb{M}_{QM}$). Note that the merging profiles are always different since the NoiOn(s) generated are different. Note that, let us denote $dist(\mathcal{O}^M, \mathcal{O})$ as $dist(\mathscr{T}^M, \mathscr{T})$, where $\mathscr{T}^M = \mathbb{H}(\mathcal{O}^M)$. An merged result using the existing merging operators is defined as follows:

Definition 6. *Let $\mathcal{P}^{k,m}$ be a "hybrid" profile of NoiOn(s). An ontology merged result is defined as:* $\mathcal{O}^M \stackrel{\text{def}}{=} y(\mathcal{P}^{k,m})$ *where* $y = \{\mathbb{M}_{MBM}, \mathbb{M}_{QM}\}$.

6 Experimental Evaluation

In this section, we describe our implementation and interpret the experimental results.

6.1 Description of Implementation

We implement the framework[4] to assess the operators with 09 practical ontologies[5], including *conference, cmt-2, ekaw, sigkdd, swo, Cree-hydro, pto, human, and mouse*. The information of ontologies is showed in Table 1a. Here, we select these sources since we investigate how the operators reacts as the number of concepts and axioms increases. Recall that the process of merging and generating NoiOn(s) is discrete.

[4] https://github.com/ontologymerging/NoisyOntologyMerging
[5] https://oaei.ontologymatching.org/2021/

Ontology Name	Nbr of Concepts	Nbr of is-a Relations	Nbr of Axioms
conference	57	55	397
cmt-2	30	38	318
ekaw	74	85	341
sigkdd	50	49	193
swo	86	144	349
cree-hydro	80	183	6,252
pto	1,541	1,747	16,148
human	3,304	5,423	30,364
mouse	2,744	4,493	11,043

(a) Number of concepts and axioms in ontologies

Ontology Name	n	k,m	rP	t	TG	TM	
						$\mathbb{M}_{MBM}$	$\mathbb{M}_{QM}$
conference	400	20,5	15%	40	281.96	77.21	61.47
cmt-2	400	20,5	15%	40	129.86	58.68	42.54
ekaw	300	20,5	15%	40	212.47	91.49	85.27
sigkdd	300	15,3	15%	50	244.39	74.52	59.46
swo	200	15,5	10%	50	301.26	439.67	171.23
cree-hydro	200	15,5	10%	80	255.62	682.26	235.43
pto	100	10,5	3%	150	637.18	3,162.98	1,586.68
human	80	10,5	3%	400	1,765.21	7,158.19	3,758.22
mouse	80	10,5	3%	400	1,177.16	6,024.52	2,981.21

(b) Parameters and the computation time

Table 1: Information and quantitative results.

For generating NoiOn(s), we implement the whole framework on pure python (python 3.8). Moreover, we use an Owlready2 library[6] to extract all information of ontology. We take account of the ontology hierarchy as a tree structure. We here use the python library named "networkx"[7] to optimize the paths (transitive relations) in the tree structure. The transitive axioms will be represented explicitly by this library. Note that the preprocessing step carries out to normalize the OriOn's axioms into the $\mathcal{EL}$ normal form [1] before generating and merging the NoiOn(s). In addition, we use the "zss" python library[8] to compute the distance (i.e., TED) between two trees *(using simple_distance())*. We implement the multi-processing procedure to improve the running time for generating noisy ontologies.

For implementing merging operations, instead of taking account of all axioms into merging process, we filter the set of the same axioms, a.k.a. agreements statements and the set of distinguishable axioms, a.k.a. disagreement statements. Here, the disagreement statements will be taken into the merging procedures. This process reduces the number of axioms and improves the running time. Now, we present how to implement merging frameworks: (1) For the $\mathbb{M}_{MBM}$ approach [6], since the process of generating all possible interpretations is huge; therefore, a timeout variable is implemented to seek out the acceptable result. Otherwise, if the number of disagreement statements is large *(i.e., greater than 80 axioms for the human ontology)*, we cluster the axioms independently in order to collect local sub-trees, which we subsequently merge. A local sub-tree corresponds to a sub-hierarchy within a general ontology structure. If a subtree remains enormous (i.e., more than 40 axioms per sub-tree), we split those sub-trees into smaller ones. This step resolves the data enumeration explosion problem. (2) For the $\mathbb{M}_{QN}$ approach [7], along with the subsumption relation, the author investigates the disjoint in this research (i.e., $A \sqcap B \sqsubseteq \bot$). We here use a "PyRCC8" python library[9] to check the consistency of the QCN. As we know, the number of concepts (regions) increases, the enumeration of all possible QCNs will be huge. Therefore, we solve the

[6] https://owlready2.readthedocs.io/en/v0.36/
[7] https://networkx.org/
[8] https://pythonhosted.org/zss/
[9] https://pypi.org/project/PyRCC8/

problem of generating all possible QCNs by the P-MAXSAT [10]. First, we translate the merged QCN into the CNF format using the FCTE encoding. Next, we build a CNF file underlying a DMACS format based on this encoding. Then, we use a MAXSAT solver[10] (using RC2) to enumerate the possible consistent QCNs. After collecting the consistent solutions from the solver, we translate back the CNF models into the QCNs. Leveraging the power of the MAXSAT solver is a suitable selection to improve the processing time in the enumeration process because consistent solutions are always able to be found (with tolerable confidence) that do not need to enumerate all possible cases.

6.2 Quantitative Results

We evaluate the computation time of the two main parts for each dataset, including the time of generating NoiOn(s) denoted as TG and the time of merging them denoted as TM. Therein, we test our proposal with several parameters, including (1) the number of noisy trees generated n, (2) the number of noisy trees selected into a profile k (3) combined with m OriOn(s), (4) the percentage of nodes to generate the NoiOn(s) rP, (5) the threshold t of selecting close NoiOn(s), and (6) finally the merged ontology's number q. These parameters and the computation time are shown in Table 1(b). We investigate our framework on two merging methods, including $\mathbb{M}_{MBM}$ [6], and $\mathbb{M}_{QM}$ [7] (see Section 5).

From the result obtained, the merging time of $\mathbb{M}_{QM}$ is faster than $\mathbb{M}_{MBM}$ (see Table 2). A reason is that the effective existing support tools of $\mathbb{M}_{QM}$ improve the computation time. Regarding the large ontology, such as $human$ and $mouse$, $\mathbb{M}_{QM}$'s merging time is acceptable and effective in practice with about 50-70 minutes. i.e., 3,758.22s for $human$ ontology. At the same time, $\mathbb{M}_{MBM}$ can spend about two hours for around 100 concepts ($3\% \times 3,304 = 99.12$). i.e., 7158.19s for human ontology. In general, the merging time of $\mathbb{M}_{MBM}$ increases rapidly in proportion to the number of ontology (subsumption) axioms, while the number of concepts influences the merging time of $\mathbb{M}_{QM}$. Otherwise, regarding TG, the computation time is also suitable in practice with around 20 minutes for a large ontology (i.e., 1,765.21 for human ontology). Although the number of noisy ontologies collected is high (400 NoiOn(s)), the TG is still around 6-7 minutes. (i.e., it is 377.60s for the "conference" ontology). The threshold t has a direct impact on TG because NoiOn(s) collected depend on this threshold. Moreover, TG also depends on the number of CPUs since we implement this procedure with multi-processing.

6.3 Qualitative Results

First of all, we provide a distance measurement table (see Table 2). It includes the distance between the OriOn and the NoiOn(s) (column 3-5) denoted by $[ON]$ and the distance between the OriOn and merged results denoted by $[OM]$ (column 6-7). Here, the TED is a measurable tool of an operator's noise-canceling ability. From these distances, we can compare and evaluate the quality of the operator. Otherwise, we also shows how the merging operators work with noisy ontologies. In general, the $\mathbb{M}_{MBM}$

[10] https://pysathq.github.io/

Ontology Name	Times	Distance ($[ON]$) (OriOn and NoiOn)			Distance ($[OM]$) (OriOn and Merged Result)	
		Avg	Min	Max	$\mathbb{M}_{MBM}$	$\mathbb{M}_{QM}$
conference	50	12.70	3	27	4.46	5.82
cmt-2	50	11.48	2	25	4.12	5.37
ekaw	50	12.78	2	28	6.78	7.86
sigkdd	50	15.85	3	32	5.83	7.03
swo	40	21.78	6	33	10.18	12.79
cree-hydro	40	33.86	9	67	14.25	16.76
pto	20	43.62	14	107	22.78	25.40
human	20	187.57	62	376	127.68	134.52
mouse	20	143.70	42	307	86.47	91.88

Table 2: A measurement of the distance (TED) to evaluate merging operators.

seems to be better than $\mathbb{M}_{QM}$ *(compare between column 6 and 7 in Table 2)*. However, both $\mathbb{M}_{MBM}$ and $\mathbb{M}_{QM}$ are effective because the $[OM]$ is always smaller than the average distance of $[ON]$. The TEDs have shown that the merge operator reduces the noise of the sources. Intuitively, the difference in the quality of the two operators is not much since the variance of distance between them is small (i.e., regarding $conference$ ontology, 4.46 of $\mathbb{M}_{MBM}$ and 5.82 $\mathbb{M}_{QM}$; the difference is $5.82 - 4.46 = 1.36$). Recall that, the distance of each edit operation is 1, for $human$ ontology, we have 127.68 of $\mathbb{M}_{MBM}$ (or 134.52 of $\mathbb{M}_{QM}$) is corresponding to 127.68 (or 134.52) modifications. Hence, if we compare 127.68 modifications with $30,364$ of input axioms, these changes are quite minimal (implying that the merged ontology is close to the OriOn).

In the following example, we provide the input sub-tree and the result of merging to illustrate that the merging outcome fully covers the original one.

Example 1. Let us take account of a $ExtremityPart$ sub-tree of the "human" conference in (i-ii) of Figure 1. Several new constraints using the $\mathbb{M}_{MBM}$ are as follows: $Toe \sqsubseteq Foot$, $Forearm \sqsubseteq Arm$, $Finger \sqsubseteq Hand$, $Hand \sqsubseteq Arm$, $Foot \sqsubseteq Leg$, others. In this case, the merging result is plausible and acceptable. Since we add $m = 5$ OriOn(s), the main structure holds.

Example 2. Let us have a sub-hierarchy of "Conference" as (iii-v) in Figure 1. Several new constraints using the $\mathbb{M}_{QM}$ are as follows: $InvSpe \sqsubseteq RegAut$, $ActConPar \sqsubseteq RegAut$, $InvSpe \sqsubseteq ConPar$, $ConCon \sqcap PassConPar \sqsubseteq \bot$, $InvSpe \sqcap PassConPar \sqsubseteq \bot$, $RegAut \sqcap PassConPar \sqsubseteq \bot$ showed in the (b) of (ii) of Figure 1. There are some disjoints in these cases, including $Con1ThAut \sqcap ConCoAuth \sqsubseteq \bot$, $EarPaiApp \sqcap LatPaiApp \sqsubseteq \bot$, $ActConPar \sqcap PasConPar \sqsubseteq \bot$.

7 Conclusion

In this paper, we introduced a framework to assess ontology merging operators. Therein, we also provided an algorithm to create noisy ontologies by modifying the structure of a source. Moreover, this framework evaluates ontology merging models on computation time and distance measurement. Finally, an experimental result using the practical ontologies was provided and discussed. Intuitively, most merged outcomes are acceptable and sensible since they cover the OriOn and are constantly cross-checked against

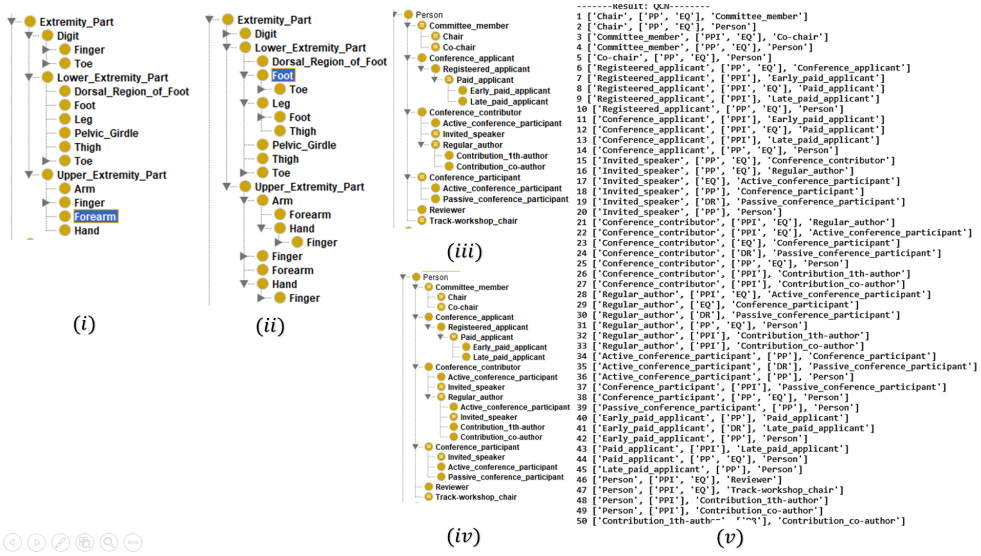

Fig. 1: Results of ontology merging. (i-ii) Structure of $ExtremityPart$ in the "Human" ontology including (i) is an original hierarchy, (ii) is the merged result using $\mathbb{M}_{MBM}$; (iii-iv) Structure of $Person$ in the "Conference" ontology including (iii) is an original hierarchy, (iv) is the merged result using $\mathbb{M}_{QM}$, (v) is the merged QCN result.

the original one (using TED). Otherwise, generating the NoiOn(s) is separate from the ontology merging process. Therefore, the evaluation of the operators is unaffected by any other external factors. Additionally, using TED to evaluate the quality of operators makes sense because we can quantify their capacity to eliminate noise. In the future, we will enhance the process of re-structuring noisy ontologies by taking account of the semantics of concept names rather than randomly selecting them. Additionally, leveraging machine learning is also our next direction to predict the potential axioms for editing ontology structures.

Acknowledgments

This work has benefited from the support of the AI Chair BE4musIA of the French National Research Agency (ANR-20-CHIA-0028) and FEI INS2I 2022-EMILIE.

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the $\mathcal{EL}$ Envelope. In: Proceedings of the 19th International Joint Conferences on Artificial Intelligence (IJCAI'05). p. 364–369 (2005)
2. Benferhat, S., Bouraoui, Z., Lagrue, S., Rossit, J.: Min-based Assertional Merging Approach for Prioritized DL-Lite Knowledge Bases. In: Proceedings of the 8th International Conference on Scalable Uncertainty Management (SUM'14). pp. 8–21 (2014)

3. Benferhat, S., Bouraoui, Z., Papini, O., Würbel, E.: Assertional Removed Sets Merging of DL-Lite Knowledge Bases. In: Proceedings of the 13th International Conference of Scalable Uncertainty Management (SUM'19). pp. 207–220 (2019)
4. Bernard, M., Boyer, L., Habrard, A., Sebban, M.: Learning probabilistic models of tree edit distance. Pattern Recognition pp. 2611–2629 (08 2008)
5. Bobillo, F., Bobed, C., Mena, E.: On the generalization of the discovery of subsumption relationships to the fuzzy case. In: Proceedings of the 26th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'17). pp. 1–6 (2017)
6. Bouraoui, Z., Konieczny, S., Ma, T.T., Varzinczak, I.: Model-based merging of open-domain ontologies. In: Proceedings of the 33rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI'20). pp. 29–34 (2020)
7. Bouraoui, Z., Konieczny, S., Ma, T., Schwind, N., Varzinczak, I.: Region-Based Merging of Open-Domain Terminological Knowledge. In: Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning (KR'22) (2022)
8. Chang, F., wei Chen, G., Zhang, S.: FCAMap-KG Results for OAEI 2019. In: Proceedings of the 18th International Semantic Web Conference (OM@ISWC'19). p. 138–145 (2019)
9. Chen, Z., Yu, S., Shengxian, W., Dianhai, Y.: RLTM: An Efficient Neural IR Framework for Long Documents. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19). pp. 5457–5463 (2019)
10. Condotta, J.F., Nouaouri, I., Sioutis, M.: A SAT Approach for Maximizing Satisfiability in Qualitative Spatial and Temporal Constraint Networks. In: Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning (KR'16). pp. 342–442 (2016)
11. Francesco, K.: Most Specific Consequences in the Description Logic $\mathcal{EL}$. Discrete Applied Mathematics **273**, 172–204 (2020)
12. Hahmann, T., Powell II, R.W.: Automatically extracting owl versions of fol ontologies. In: Proceedings of the 20th International Semantic Web Conference (ISWC'21) (2021)
13. Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G.D., Gutierrez, C., Kirrane: Knowledge Graphs. ACM Computing Surveys p. 1–37 (2021)
14. Konieczny, S., Pérez, R.P.: Merging Information under Constraints: A Logical Framework. Journal of Logic and computation **12**(5), 773–808 (2002)
15. Lin, J., Mendelzon, A.O.: Knowledge Base Merging by Majority, pp. 195–218. Springer, Dordrecht (1999)
16. Movshovitz-Attias, D., Whang, S.E., Noy, N., Halevy, A.: Discovering subsumption relationships for web-based ontologies. In: Proceedings of the 18th International Workshop on Web and Databases (WebDB'15). p. 62–69 (2015)
17. Navigli, R.: Natural language understanding: Instructions for (present and future) use. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18). pp. 5697–5702 (07 2018)
18. Schwarz, S., Pawlik, M., Augsten, N.: A new perspective on the tree edit distance. In: Similarity Search and Applications. pp. 156–170 (2017)
19. Tai, K.C.: The tree-to-tree correction problem. Journal of the ACM **26**(3), 422–433 (Jul 1979)
20. Thiéblin, É., Haemmerlé, O., Trojahn, C.: CANARD complex matching system. In: Proceedings of the 17th International Semantic Web Conference (OM@ISWC'18). p. 138–143 (2018)
21. Wang, Z., Wang, K., Jin, Y., Qi, G.: Ontomerge: A system for merging DL-Lite ontologies. CEUR Workshop Proceedings **969**, 16–27 (01 2012)
22. Zhang, K., Shasha, D.: Simple fast algorithms for the editing distance between trees and related problems. SIAM J. Comput. **18**(6), 1245–1262 (dec 1989)
23. Zhang, K., Statman, R., Shasha, D.: On the editing distance between unordered labeled trees. Information Processing Letters **42**(3), 133–139 (1992)