Applied Intelligence (2023) 53:20382–20401
https://doi.org/10.1007/s10489-023-04545-0

Methods of managing the evolution of ontologies and their alignments

Marcin Pietranik[1] · Adrianna Kozierkiewicz[1]

Accepted: 24 February 2023 / Published online: 12 April 2023

Abstract
Nowadays, none can expect that knowledge about some part of reality will not change. Consequently, a representation of such evolving knowledge (for example, ontologies) also changes. Such changes entail that applications incorporating such knowledge may become compromised and yield wrong results. An example of such an application is ontology alignment which can be informally described as a set of connections between two ontologies. Those connections mark elements from two ontologies that relate to the same parts of reality. In changing one of the corresponding ontologies, such connections may become invalid. One may designate the ontology alignment once again from scratch for altered ontologies. However, such an approach is time and resource-consuming. The paper comprehensively presents our ontology evolution and alignment maintenance framework. It can be used to preserve the validity of ontology alignment using only the analysis of changes introduced to maintained ontologies. The precise definition of ontologies is provided, along with a definition of the ontology change log. A set of algorithms that allow revalidating ontology alignments have been built based on such elements.

Keywords Ontology alignment · Ontology evolution · Knowledge management

1 Introduction

A widely known informal definition created by Thomas Gruber states that ontology is a formal description of a shared conceptualization of a domain of discourse. In other words, for some selected areas of knowledge, it is possible to create multiple different decompositions into basic objects (referred to as concepts). Additionally, definitions of various interactions in which these objects may participate can be added. In consequence, two separately developed ontologies may significantly differ in many different details, although describing the same universe of discourse.

Matching the contents of two ontologies is crucial if there is a requirement to provide methods for communicating between two knowledge-based systems that incorporate ontologies as a backbone. This problem is a broadly discussed topic called ontology alignment or ontology mapping [9]. A common approach to the described problem assumes that the aligned ontologies do not change in time. In other words, no alteration is applied to them by their authors. Such an approach significantly simplifies the task, entailing that once an alignment between two ontologies is designated, it will not need any further maintenance. However, in many application domains, such an assumption is not plausible. It is impossible to expect that no modification will be introduced to ontologies that participate in communication between two knowledge-based systems. Therefore, it is also impossible to expect that the designated alignment will not become invalid due to the changes in the mapped ontologies. When some aligned ontologies change, their alignment may be designated again from scratch. However, processing whole ontologies can be very time-consuming, especially when such ontologies are large [26].

In this paper, we want to concentrate on answering two questions. The first one focuses on identifying a situation when ontologies evolved significantly enough

Adrianna Kozierkiewicz is contributed equally to this work.

✉ Marcin Pietranik
marcin.pietranik@pwr.edu.pl

Adrianna Kozierkiewicz
adrianna.kozierkiewicz@pwr.edu.pl

1 Faculty of Information and Communication Technology, Wroclaw University of Science and Technology, Wybrzeże Wyspiańskiego 27, Wrocław, 50-370, Poland

to possibly invalidate their alignment. The second one addresses updating the existing alignment of two ontologies using only the information about their evolution.

In the initial stages of our work, we started with the essential elements - by analyzing what kind of information can be included within ontologies and how they can be expressed. We distinguished three types of elements present within ontologies that allow us to organize the content of ontologies into a level of concepts, a level of relations, and a level of instances.

The first one concerns defining the aforementioned basic objects occurring within the selected universe of discourse. These objects represent a common class of elements, for example, *a person, a book, or an article*. Such elements may interact with other elements, for example a person *writes* a book, which *contains* articles. Definitions of these interactions form a level of relations. A final level of instances contains definitions of particular materializations of concepts. For example, an instance of a concept *Person* can be John or Mary.

According to the described structuring of ontologies, the ontology alignment framework can only complete if it allows for mapping and integrating ontologies on each of the described levels. These mechanisms have been developed in our previous publications [12], forming a consistent Framework for Ontological Knowledge Integration (referred to further as FOKI). This foundation has been extended and adjusted to manage the evolution of ontologies and their alignments. The following paper is a broad overview of this framework and an extensive summary of our earlier work.

The main contribution is built on detailed definitions of ontologies and their elements and how to track changes that may be applied to them. We have developed a set of consistent algorithms that allow revalidation and updating ontology alignments in case one of the mapped ontologies evolve. Using the proposed methods may assert that the maintained alignment never becomes obsolete.

What is worth emphasizing is that the primary purpose of all of the procedures presented in this article is not to outperform any ontology alignment method in terms of the correctness of designated mappings. The base approach to ontology mapping uses whole ontologies to select elements that relate to the same objects from their domain. The algorithms described in this article address different issues. Their main goal is updating the ontology alignment designated before any changes have been introduced to ontologies utilizing only information about introduced changes. They are expected to return the alignments with at least similar quality (according to some assumed measures) to the ones designated by processing whole ontologies from scratch using some ontology alignment systems.

Providing this contribution includes a broad experimental verification of the developed methods, which incorporates two novel ontology alignment quality assessment measures [29]. They are used in conjunction with a widely accepted dataset provided by the Ontology Alignment Evaluation Initiative [4], which are considered a state of the art benchmarks for verifying any ontology-related tools.

The remainder of the paper is as follows. In the next part, an overview of related works in the fields of ontology alignment and ontology evolution is given. Section 3 contains mathematical foundations of our work. The main contribution of the following article can be found in Section 4, which is split into two subsections. The first one presents the criteria for updating ontology alignments. These elements have been previously described in our earlier article [22] and are included for clarity reasons. The second contains algorithms that can be used to achieve this goal, which has been introduced earlier in [21, 23, 24]. However, this paper provides a more detailed explanation of developed methods, focusing on their evaluation. The verification of the presented methods is provided in Section 5. The experimental results include data collected from a dedicated environment that calculates the quality of concept alignments using novel measures developed in our previous article [29]. An overview of our upcoming research plans and a summary are given in Section 6.

2 Related works

As a flexible knowledge representation method, ontologies have been helpful in many applications. They allow describing the complexity of knowledge - relations, hierarchies, and other dependencies between objects or concepts. However, the content of some ontology may need to be updated due to changing requirements that appear in the real world. Such changes must be reflected in created ontologies by introducing and maintaining changes over time without purging them. Therefore, methods of managing the ontology evolution process are highly valued. If some evolving ontology is mapped into another ontology (using ontology alignment techniques), their evolving content inevitably affects the mappings between them, thus entailing that such mapping needs to be revalidated. Obviously, the alignment can be designated from scratch. A better approach would be based only on analyzing changes introduced to ontologies and updating only affected mappings, thus maintaining them. However, it is only possible to develop such an approach with reliable methods of managing ontology evolution. Therefore, This section of the article overviews the two research fields: *(i)* methods of managing the ontology evolution process and *(ii)* methods of maintaining alignment between evolving ontologies.

2.1 Methods of managing the ontology evolution process

One of the most prominent applications where managing the ontology evolution process was crucial is structuring genomes. Research on this topic is conducted by The Gene Ontology Consortium, which focuses on developing Gene Ontology (GO) [6] - a part of the Open Biomedical Ontologies (OBO) [34]. Several publications are addressing the issues related to temporal changes applied to these ontologies. In [15], authors identified several scenarios during which the created Gene Ontology (GO) may change. Predefined post-processing procedures and clean-up rules assert the consistency of the final version of evolving ontology. Similar research, but focusing on topic-independent ontologies, may be found in [7].

Updating and revalidating evolving biomedical ontology mappings is the main topic of [2]. The authors provided a set of refinement algorithms based on calculating similarities between added concepts and their neighborhoods in the final state of ontology. The verification has been performed using a domain ontology named Logical Observation Identifiers Names and Codes (LOINC) in English and Spanish variants. However, the evaluation addressed only several added mappings without analyzing their quality.

Authors of [34] describe the approach to organizing cooperative work on included ontologies and coordinating changes applied to them over time. This approach differs from the one proposed in this paper because it does not incorporate the history of changes applied to ontologies. Therefore, if some error is introduced to ontology, it is impossible to recover the ontology to its previous state.

In [20], authors present a comprehensive approach to ontology evolution. The research addresses the automatic detection of ontology changes, then a method for rewriting queries that may be sent to evolving ontologies is presented. Those rewrites are classified into two groups: the ones that may be soundly performed automatically and those that require user involvement. In the latter case, the authors provide the best over-approximations of processed queries. The presented approach significantly reduced ontology maintenance costs.

The approach presented in [31] tackles the problem from a different perspective. Changes applied to some ontology in its consecutive versions are used to create a graph of relevance. This graph orders versions of ontology not in terms of their chronology, but relevancy, computed based on four criteria: conceptualization, usage frequency, abstraction, and completeness. A tool called Consistology is described - it was developed to aid users in managing ontology evolution.

In [36], authors propose a systematic solution based on an adaptation of OWL2 language by introducing a notion of time to support temporal versioning of the ontology schema. Therefore, only the evolution on the level of the concept is addressed. The methods proposed in this paper broadly treat ontologies by covering all of the levels of abstraction described in Section 1.

In [1], a completely different approach to maintaining ontology evolution can be found. Authors reject solutions based on versioning of ontologies by defining the so-called Historical Knowledge Graph, which is knowledge from all versions of an ontology unified in a single graph. Such a solution relieves maintenance procedures from analyzing subsequent versions and tracking changes. All the expressed knowledge is freely available, simplifying inference processes. The downside of such approach is the necessity of storing and maintaining rapidly expanding ontology, which may become cumbersome.

A survey of different types of changes that may appear in ontologies and their semantical implications can be found in [8]. The research can be treated as preliminary work to perform the analysis of the evolution of not only ontologies but also their alignment. A broad description of related issues may be found [10]. The continuation of this research can be found in [11]. It includes a comprehensive approach to determining differences between two states of the same ontology, focusing on providing the invertibility of applied modifications. The proposed algorithms were evaluated using large life science ontologies- the Gene Ontology [6] and the NCI Thesaurus [33].

2.2 Methods of maintaining alignment between evolving ontologies

Preliminary research on maintaining ontology alignment under ontology evolution can be found in [30], where authors identified different types of changes that can be applied to ontologies. On top of such classification, the authors provide an overview of potential change operations that may be applied to ontology alignment. The proposed approach is verified using large biomedical ontologies. The obtained results are analyzed both quantitatively and qualitatively.

In [35], a method for updating ontology alignments is presented. Hover, the presented method addresses only mapping new concepts added to the managed ontologies, and no revalidation of mappings connecting altered concepts is performed. The element which differs from the framework presented in this article is a set of criteria utilizing the degree of significance of changes introduced to ontologies. The developed functions can answer the question of when the alignment revalidation should happen since the ontology alteration may not always be equally significant. In other words, revalidating ontology alignments may not be necessary for every small change in the mapped ontology. To the best of our knowledge, there is very little research on related issues, and a similar approach has only been developed in [27]. However, the authors focus not on ontologies but on a similar tool known as the Knowledge Graphs. Authors provide tools for predicting the impact of their changes. Similarly, tracking the evolution of Knowledge Graphs is a subject of [17].

Research from [3] describes a method for identifying the most critical attributes present within mapped and evolving concepts. Authors claim that only changes in these attributes should be followed by ontology mapping reconciliation to assert consistent ontology maintenance. The presented approach is significantly different from the ideas provided in this article. Our work treats ontologies holistically and does not focus on atomic elements.

The authors of [25] focus on the inter-organizational environment for engineering ontologies. The DOGMA-MESS system is described, which allows for scalable, community-driven collaboration on ontologies. The approach is based on ontology alignment and meaning negotiation based on user involvement. A substantial amount of research addresses the issue of user experience and very little to automatize the whole process. Such a feature vastly differs from the ideas presented in this article.

To the best of our knowledge, the most comprehensive approach to managing the evolution of ontology alignment can be found in [18] and its continuation in [19]. Authors focus on detecting changes in ontologies (both on schema and instance level) to find potential inconsistencies that may be accidentally introduced. These inconsistencies are addressed by applying so-called deduced changes to keep the ontology in a consistent state. In [18], a set of recovery and roll-back algorithms can be found, while in [19], authors provide mapping reconciliation algorithms that use the change history log. The authors' work is similar to the following article. However, in our work, we have separated algorithms that process changes in concepts, instances, and relations, while in [18, 19], ontologies are always treated as a whole structure. Such an approach, despite its holistic nature, has a few disadvantages. First, the integrated solution is complicated to deploy in any distributed environment. Ontologies must always be treated as indivisible wholes. Therefore, even minimal changes in ontologies cannot be quickly processed and addressed, but the comprehensive procedure must be executed every time.

Secondly, the split algorithms proposed in this article can handle different types of ontology changes independently from each other. Even if those changes affect one another, they will still be tracked and resolved. However, if they do not affect each other, the proposed decoupling entails higher solution flexibility. Third, it is impossible to track potential weaknesses of the procedure that may appear on different levels of ontologies. In other words, one cannot say if the proposed solution process changes that appear on the level of concepts with the same effectiveness as changes on the level of instances.

Finally, our framework is equipped with change significance measures that can be used to track how many and how extensive changes have been applied to ontologies. With such functions, it is easy to compare their outcomes with some assumed significance threshold value, making it easy to decide if the designated alignment between the maintained ontologies needs revalidation. In other words, the defined functions can be used as criteria for triggering further actions. To the best of our knowledge, in the literature, there is no ontology alignment maintenance framework that approaches the field in such a holistic way.

3 Basic notions

3.1 Ontology definition

We define a pair *(A,V)*, where A is a set of attributes describing objects and V is a set of valuations of such attributes (their domains), where: $V = \bigcup_{a\in A} V_a$. This pair defines the so-called "closed world" on top of which it is possible to define the (A, V)-based ontology as a quintuple:

$$O = (C, H, R^C, I, R^I) \tag{1}$$

where:

- C is a finite set of concepts,
- H is a concepts' hierarchy, a distinguished relation between concepts,
- R^C represents a set of concepts relations $R^C = \{r_1^C, r_2^C, ..., r_n^C\}$, $n \in N$, such that every $r_i^C \in R^C$ $(i \in [1, n])$ is a subset of a cartesian product, $r_i^C \subset C \times C$,
- I is a set of instances' identifiers,
- $R^I = \{r_1^I, r_2^I, ..., r_n^I\}$ is a set of relations between concepts' instances.

A set of all (A, V)-based ontologies will be denoted as $\tilde{O}$. Each concept c of the set C has a following structure:

$$c = (id^c, A^c, V^c, I^c) \tag{2}$$

where:

- id^c is an identifier of the concept c,
- A^c denotes a set of its attributes, $(A^c \subset A)$
- V^c represents a set of attributes domains, where $V^c = \bigcup_{a\in A^c} V_a$,

- I^c is a set of concepts' c instances.

Attributes from set A alone carry no explicit meaning. They can only be interpreted if they are part of some chosen concept. Moreover, when the same attribute is included as a part of several different concepts, its meaning may vary from one concept to another. To achieve this level of expressivity without restricting the use of the same attribute in multiple concepts, introduce a notion of attributes semantics. We assume the existence of a set D_A, which contains atomic descriptions of attributes and a sub-language (L_S^A) of the propositional calculus consisting of elements from D_A and basic logic operators (conjunction, disjunction, and negation). To grant attributes meaning, we define a function:

$$S_A : A \times C \rightarrow L_S^A \tag{3}$$

It assigns a logic sentence from L_S^A to attributes when they are included in a specific concept. It can increase the expressiveness of bare attributes or define constraints on the attribute values. For better understanding consider an attribute *birthday* within a concept *Person*. The following semantics can be defined: $S_A(birthday, Person)$: $day\_of\_birth \wedge month\_of\_birth \wedge year\_of\_birth \wedge age \wedge zodiac\_sign$. Providing values of the function S_A manually is extremely time-consuming and error-prone. We have developed a semi-automatic method in our other publications (e.g., [13]), however, describing it falls outside the scope of the following article.

The overall semantic of a concept c is given by its context. Formally it can be defined as a conjunction of the semantics of its attributes:

$$ctx(c) = S_A(a_1, c) \wedge S_A(a_2, c) \wedge ... \wedge S_A(a_n, c) \tag{4}$$

The set I from (1) contains identifiers of instances that can be assigned to concepts from the set C. Formally, these assignments are expressed using the set I^c from (2), which members are tuples of the form:

$$i = (id^i, v_c^i) \tag{5}$$

where: id^i is an identifier, and v_c^i is a vector of values of type V^c and v_c^i gives the instance i specific valuations of attributes included in the concept c.

By *Ins(c)* we denote a set of identifiers of instances of a concept c, formally: $Ins(c) = \{id^i \mid (id^i, v_c^i) \in I^c\}$. The reverse function Ins^{-1} given an instance identifier returns a subset of concept set to which it has been assigned, formally: $Ins^{-1}(i, C) = \{c \mid c \in C \wedge i \in c\}$. Therefore, the set I containing instances identifiers has the property $\forall_{c \in C} Ins(c) \subseteq I$.

The next element in ontologies are relations between concepts. The set R^C is responsible for describing which concepts can be connected. Similarly to concepts, relations from the set R^C have no semantics. Therefore, by analogy to L_S^A, we define a set D_R with atomic descriptions of relations and a sub-language L_S^R of the propositional calculus built from the content of D_R and logic operators of conjunction, disjunction, and negation. This allows us to introduce a function $S_R : R^C \rightarrow L_S^R$ that assigns a logic sentence from L_S^R to a relation from the set.

Every relation r_j^C connecting two concepts from the set R^C has a corresponding relation r_j^I in the set R^I. It contains pairs of instances of concepts that are linked by the relation r_j^C. It is denoted using the same index *j*. Such an approach allows us to define what exactly is connected easily. For example, If we consider relation from the set R^C i.e. *is_married* it can connect two concepts like i.e. *Man* and *Woman*. A corresponding relation from the set R^I describes connections of specific instances. For example, a pair $(John, Ann)$ would indicate that *John* (an instance of a concept *Man*) is a husband of *Ann* (an instance of a concept *Woman*).

The hierarchy of concepts H is a distinguished relation between concepts, representing a rooted tree. For clarity purposes, it is excluded from the set R^C, despite being similarly defined as $H \subset C \times C$. Its elements must meet some specific constraints. A pair of concepts $c_1 = (id^{c_1}, A^{c_1}, V^{c_1}, I^{c_1})$ and $c_2 = (id^{c_2}, A^{c_2}, V^{c_2}, I^{c_2})$ can only be included in H only if:

1. $|A^{c_2}| \geq |A^{c_1}|$
2. $\forall a \in A^{c_2} \exists a' \in A^{c_1} : S_A(a, c_2) \implies S_A(a', c_1)$
3. $Ins(c_2) \subseteq Ins(c_1)$

If a pair of concepts (c_1, c_2) is included in H we can refer to these two concepts as connected with a subsumption relation, entailing that c_2 is a subclass of c_1. Obviously such relation is transitive, therefore, if there exists a chain of concepts $c_1, \ldots, c_n$ such that $\forall_{i \in [1, n-1]}(c_i, c_{i+1}) \in H$ then obviously a pair (c_1, c_n) also fulfils conditions for subsumption, thus we can write $c_1 \leftarrow c_n$.

In the set C there exists an abstract concept *Thing* with a structure defined according to (2) as $\{Thing, \emptyset, \emptyset, \bigcup_{id \in I} \{(id, \emptyset)\}\}$. It is a root of hierarchy H, and all other concepts are its subclasses. It allows us to introduce a notion of depth of ontology O denoted as $Depth(O)$, which can be calculated as the depth of the tree represented by the concept hierarchy H.

Similarly, for a particular concept $c \in C$ by $Depth(O, c)$, we will denote the length of the shortest path in the tree (represented by the concept hierarchy H) from the abstract class *Thing* to the given concept c. $Subtree(O, c)$ denotes a set containing all of the concepts descendants from c, which formally fulfills the following conditions:

1. $\forall_{c' \in Subtree(O,c)} c \leftarrow c'$

2. $\neg\exists c' \in C \setminus Subtree(O, c) : c \leftarrow c'$

As described above, our ontology model consists of several elements: concepts, relations, and instances. All these elements, along with their integration methods, form a Framework for Ontological Knowledge Integration (further referred to as FOKI). In this paper, we present the extension of the FOKI framework with the method of managing ontologies' evolution and their alignments.

In the real world, expecting some ontology to remain static throughout its maintenance is impossible. Informally speaking, some changes will have to be applied sooner or later, and the particular ontology will evolve. Therefore, we need to introduce a notion of time. In further parts of the following article, an ordered set of discrete moments in time, representing a timeline of every evolving ontology, will be denoted as $TL = \{t_n \mid n \in \mathbb{N}\}$. $TL(O)$ will denote a subset of TL, containing only moments in which the ontology O has changed. A version of the ontology O at time $t_m \in TL(O)$ will be denotes as $O^{(m)}$. When two versions of one ontology are available $O^{(m-1)}$ and $O^{(m)}$ to represent the fact that the former is earlier than the latter, we will use the following notation: $O^{(m-1)} \prec O^{(m)}$. It can also be used for particular elements of ontologies e.g. $c^{(m-1)} \prec c^{(m)}$ denotes that $c^{(m-1)}$ is earlier than $c^{(m)}$. An ontology O in its initial state is denoted as $O^{(0)} = (C^{(0)}, H^{(0)}, R^{C(0)}, I^{(0)}, R^{I(0)})$.

The above assumptions allow us to define an ontology repository - a sequence of versions of a particular ontology. Formally:

$$Rep(O) = \left\{O^{(m)} \mid m \in TL(O)\right\} \tag{6}$$

3.2 Ontology change log

To express changes that appeared while maintained ontologies evolve, we need a tool that can tell us *what changed when*. For this task, we define an ontology log which provides a complete picture of changes applied to some ontology. It is defined as a set of results of $diff$ functions, calculated for every pair of subsequent ontology versions from the timeline $TL(O)$:

$$\begin{aligned}\text{Log}(O) = \Big\{\Big\langle & diff_C(O^{(m-1)}, O^{(m)}),\\ & diff_I(O^{(m-1)}, O^{(m)}),\\ & diff_{R^C}(O^{(m-1)}, O^{(m)})\Big\rangle \mid m \in TL(O) \setminus \{0\}\Big\}\end{aligned} \tag{7}$$

The function $diff_C$ takes as input two states of the set of ontology concepts ($C^{(m-1)}$ and $C^{(m)}$). It returns a triplet of sets - a set of concepts added to the ontology, a set of concepts removed from it, and a set of altered concepts (which includes pairs of the same concept in its earlier and later state):

$$\begin{aligned}diff_C(O^{(m-1)}, O^{(m)}) = \Big\langle & new_C(C^{(m-1)}, C^{(m)}),\\ & del_C(C^{(m-1)}, C^{(m)}),\\ & alt_C(C^{(m-1)}, C^{(m)})\Big\rangle\end{aligned} \tag{8}$$

where:

1. $new_C(C^{(m-1)}, C^{(m)}) = C^{(m)} \setminus C^{(m-1)}$
2. $del_C(C^{(m-1)}, C^{(m)}) = C^{(m-1)} \setminus C^{(m)}$
3. $alt_C(C^{(m-1)}, C^{(m)}) = \Big\{(c^{(m-1)}, c^{(m)}) \mid c^{(m-1)} \in C^{(m-1)} \wedge c^{(m)} \in C^{(m)} \wedge ctx(c^{(m-1)}) \neq ctx(c^{(m)})\Big\}$

On the instance level, the function $diff_I$ cannot be solely based only tracking changes of the set I. This set contains only instance identifiers and, according to (2), concrete values of instances (materializations of particular concepts) are defined within structures of concepts. Thus, the function $diff_I$ must treat the ontology O in a more holistic manner. It takes as input two versions of sets of concepts and instances, and analogous to (8) returns three sets. The first contains instance identifiers added to the ontology or newly assigned to some concept. The second set obtains instance identifiers removed from the ontology or which assignments to some concept have been deleted. The third set contains identifiers of instances which valuations in some concepts have been altered. It also provides concepts to which such instance has been assigned in their earlier and later versions. Formally, $diff_I$ is defined as follows:

$$\begin{aligned}diff_I(O^{(m-1)}, O^{(m)}) = \Big\langle & new_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)}),\\ & del_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)}),\\ & alt_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)})\Big\rangle\end{aligned} \tag{9}$$

where:

1. $new_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)}) = \Big\{i \mid (i \in I^{(m)} \wedge i \notin I^{(m-1)}) \vee (Ins^{-1}(i, C^{(m-1)}) = \emptyset \wedge Ins^{-1}(i, C^{(m)}) \neq \emptyset)\Big\}$
2. $del_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)}) = \Big\{(i \mid i \in I^{(m-1)} \wedge i \notin I^{(m)}) \vee (Ins^{-1}(i, C^{(m-1)}) \neq \emptyset \wedge Ins^{-1}(i, C^{(m)}) = \emptyset)\Big\}$

3. $alt_I(I^{(m-1)}, C^{(m-1)}, I^{(m)}, C^{(m)}) = \left\{(i, c^{(m-1)}, c^{(m)}) \mid i \in c^{(m-1)} \wedge i \in c^{(m)} \wedge v^i_{c^{(m-1)}} \neq v^i_{c^{(m)}}\right\}$

On the level of relations, the function $diff_{R^C}$ takes as its input the set of concept relations R^C in its earlier and later state. It designates three sets describing relations added to the ontology, relations removed from the ontology, and changed concepts' relations. Formally it is defined below:

$$diff_{R^C}(O^{(m-1)}, O^{(m)}) = \Big\langle new_{R^C}(R^{C(m-1)}, R^{C(m)}), del_{R^C}(R^{C(m-1)}, R^{C(m)}), alt_{R^C}(R^{C(m-1)}, R^{C(m)})\Big\rangle \quad (10)$$

where:

1. $new_{R^C}(R^{C(m-1)}, R^{C(m)}) = R^{C(m)} \setminus R^{C(m-1)}$
2. $del_{R^C}(R^{C(m-1)}, R^{C(m)}) = R^{C(m-1)} \setminus R^{C(m)}$
3. $alt_{R^C}(R^{C(m-1)}, R^{C(m)}) = \left\{(r^{(m-1)}, r^{(m)}) \mid r^{(m-1)} \in R^{C(m-1)} \wedge r^{(m)} \in R^{C(m)} \wedge \frac{|r^{(m-1)} \cap r^{(m)}|}{|r^{(m-1)} \cup r^{(m)}|} \neq 1 \vee S_R(r^{(m-1)}) \neq S_R(r^{(m)}))\right\}$

3.3 Ontology alignment

Informally speaking, for communication using two or more ontologies, some "bridge" is needed, which allows finding connections between considered ontologies. In other words, between two independent (A, V)-based ontologies $O_1 = (C_1, H_1, R^{C_1}, I_1, R^{I_1})$ and $O_2 = (C_2, H_2, R^{C_2}, I_2, R^{I_2})$ there exist a set of correspondences, called alignment, defined in the following way:

$$Align(O_1, O_2) = \{Align_C(O_1, O_2), Align_I(O_1, O_2), Align_R(O_1, O_2)\} \quad (11)$$

Of course, it is possible to determine the separate set of correspondences for each ontology level (concepts, instances, and relations). However, we do not consider the alignment of instance relations. The reason for that is twofold - first, instances are a materialization of concepts in independent ontologies. Therefore, their relations are tightly bound to these ontologies, and mapping them does not carry any particular meaning. Secondly, the Ontology Alignment Evaluation Initiative does not provide any test datasets that could be used to verify the algorithms addressing the alignment of instance relations.

Formally, we define an ontology alignment on the concept level as a set containing tuples in the following form:

$$Align_C(O_1, O_2) = \{(c_1, c_2) \mid c_1 \in C_1 \wedge c_2 \in C_2 \wedge \lambda_C(c_1, c_2) \geq T_C\} \quad (12)$$

where:

- c_1, c_2 are concepts from O_1 and O_2 respectively,
- $\lambda_C(c_1, c_2)$ is a value of a degree to which concept c_1 can be aligned into the concept c_2, a vast majority of alignments between two ontologies include only mappings of concepts that are equivalent with 100% certainty.
- T_C represents an assumed threshold

The alignment on the instance level is a set of sets of alignments of instances belonging to two already aligned concepts:

$$Align_I(O_1, O_2) = \{AL_{O_1,O_2}(c_1, c_2) \mid (c_1, c_2) \in Align_C(O_1, O_2)\} \quad (13)$$

where:

- $AL_{O_1,O_2}(c_1, c_2) = \{(i_1, i_2) \mid i_1 \in c_1 \wedge i_2 \in c_2 \wedge \lambda_I(v^{i_1}_{c_1}, v^{i_2}_{c_2})) \geq T_I\}$
- λ_I denotes a degree to which the instance i_1 can be aligned to i_2,
- T_I is some assumed threshold

The alignment on the relation level is defined as follows:

$$Align_R(O_1, O_2) = \{(r_1, r_2) \mid r_1 \in R^{C_1} \wedge r_2 \in R^{C_2} \wedge \lambda_R(r_1, r_2) \geq T_R\} \quad (14)$$

where:

- λ_R is a degree to which relation r_1 can be aligned to r_2,
- T_R denotes the assumed value of threshold

All alignment definitions can be extended by using a superscript to introduce the aspect of time. For example, $Align(O_1^{(m)}, O_2^{(n)})$ denotes an alignment of ontologies O_1 and O_2 in their respective states in time m and n $(m, n \in TL)$.

4 Updating ontology alignment

4.1 Measuring the degree to which ontologies change

In this section article, to simplify the notation, we will skip the the indexes that describe moments in time, and by O and O' we will denote an ontology O and its elements from (1) in its previous and current states respectively.

Ontologies may evolve. However, the changes made in the ontology may have different impacts. Some changes may have a significant influence on existing mappings, and some of them may be marginal. It means that even if we observe many changes applied to ontologies, these changes do not necessarily entail that alignments between such ontologies are no longer valid.

The function $diff$ can be used to designate the evolution of some ontology, however, it does not show how significantly such ontology changed. The first contribution of our research is developing a tool that can be used to verify if alignment revalidation is required. For each ontology level, we define a function that reflects a degree of change significance. On the level of concepts, such function Ψ_C must meet postulates defined below:

- **P1.** $\Psi_C(C, C') \in [0, 1]$
- **P2.** $\Psi_C(C, C') = 0 \iff diff_C(O, O') = \langle \emptyset, \emptyset, \emptyset \rangle$
- **P3.** $\Psi_C(C, C') = 1 \iff diff_C(O, O') = \langle C', C, \emptyset \rangle$

The first postulate states that the values of the function Ψ_C must come from a range [0,1]. Two further postulates address edge conditions. **P2** addresses a situation in which the change significance is minimal (equal to 0). Such a situation occurs if no modification on the concept level has been applied - no new concepts have been added, no concepts have disappeared, and no concepts have been modified.

P3 addresses the opposing situation to **P2**. The value of change significance is maximal (equal to 1) only if the ontology has completely changed. Such a situation takes place only when all old concepts have been deleted (hence the first component of $diff_C$ equals C) and all present concepts are new (hence the second component of $diff_C$ equals C'). If every concept from the earlier state has been removed and every concept from the newer state is new, then no modifications have been applied. Therefore, the third component of $diff_C$ is an empty set.

The degree of changes applied to concepts can be calculated in the following way:

$$\Psi_C(C, C') = \frac{\mid new_C(C, C') \mid + \mid del_C(C, C') \mid + \sum\limits_{(c,c') \in alt_C(C,C')} d_s(ctx(c), ctx(c'))}{\mid C' \mid + \mid del_C(C, C') \mid} \tag{15}$$

The denominator in the (15) will always be higher the numerator thus making Psi_C meets **P1**. If no changes has been applied to concepts $diff_C(O, O') = \langle \emptyset, \emptyset, \emptyset \rangle$ then the nominator will equal 0 and in consequence the whole equation will also equals 0 thus **P2** will be met. If every concept from the ontology has been changed ($diff_C(O, O') = \langle C', C, \emptyset \rangle$) then the nominator and the denominator will be equal. In such situation the value of (15) would be equal to 1, which fulfils **P3**.

A function d_s, used in the above equation, determines a distance between two logic formulas. At first, it transforms both input formulas to the disjunctive normal form (DNF). It allows treating the rewritten formulas as sets of groups of symbols. It is possible to calculate a Jaccard's distance measure between two such groups. The function d_s creates a cartesian product of groups from both input formulas and calculates a distance for each pair. The final result is an average of partial values distance. For details, please refer to our previous publication [28].

A function Ψ_I, which calculates a degree of change significance on the level of instances, accepts sets of concepts C and C' as its input. This is enforced by the factthat changes applied to instances do not cover only additions or removals of instance identifiers from the set I, but also changes in valuations of instances. These changes are tightly bound to concept definition - namely, values of v_c^i from (5). The function Ψ_I must meet the following postulates:

- **P1.** $\Psi_I(I, C, I', C') \in [0, 1]$
- **P2.** $\Psi_I(I, C, I', C') = 0 \iff diff_I(O, O') = \Big\langle \emptyset, \emptyset, \emptyset \Big\rangle$
- **P3.** $\Psi_I(I, C, I', C') = 1 \iff diff_I(O, O') = \Big\langle I', I, \emptyset \Big\rangle$

P1 states that the values of the function Ψ_I must come from a range [0,1]. **P2** considers a situation where the significance of a change equals 0 (minimal value). This situation occurs when no alteration has been introduced to the considered ontology. The last postulate **P3** addresses the opposite situation - it occurs if all instances from the initial state have been deleted and all instances from the last state are new.

The final formal definition of Ψ_I from (16) can be found below:

$$\Psi_I(I, C, I', C') = \frac{\mid new_I(I, C, I', C') \mid + \mid del_I(I, C, I', C') \mid + \sum\limits_{(i,c,c') \in alt_I(I,C,I',C')} dist(v_c^i, v_{c'}^i)}{\mid I' \mid + \mid del_I\ (\ I, C, I', C' \mid} \tag{16}$$

It utilizes a helper function *dist*, which calculates a distance between two vectors of values that represent instances according to the definition on (5). Those vectors' contents are heterogenous and may include values from different domains (e.g., integers, captions, decimals). Thus, it is impossible to provide a generic definition of *dist*, which must be customized for every concept structure in maintained ontology to compare instances of those concepts. Some examples may be found in [14].

The denominators of all three components of the (16) are the same, which allows us to treat the whole expression as a sum of their nominators divided by the denominator. It will always be a higher value, thus making Psi_C meets **P1**. If no changes has been applied to instances $diff_I(O, O') = \langle\emptyset, \emptyset, \emptyset\rangle$ then the nominator will equal 0 and in consequence the whole equation will also equals 0 thus **P2** will be met. Suppose every instance from the ontology has been changed either by completely altering existing ones or removing all of them and adding new ones. In that case, the nominator and the denominator will be equal. In such situation the value of (16) will equal 1, which fulfils **P3**.

The last function Ψ_R, which provides the means to estimate how significantly relations have changed, must meet the following postulates:

- **P1.** $\Psi_R(R^C, R^{C'}) \in [0, 1]$
- **P2.** $\Psi_R(R^C, R^{C'}) = 0 \iff diff_{R^C}(O, O') = \left\langle \emptyset, \emptyset, \emptyset \right\rangle$
- **P3.** $\Psi_R(R^C, R^{C'}) = 1 \iff diff_{R^C}(O, O') = \left\langle R^{C'}, R^C, \emptyset \right\rangle$

The first postulate **P1** states that the values of the function Ψ_R must be constrained to the range [0,1]. **P2** addresses a situation where the degree of the significance of change is minimal - no new relation has appeared, no relation has been removed, and no relation has changed.

The last postulate **P3** considers a situation when the ontology has been wholly modified on the level of relations-every relation from the earlier state has been deleted, and every relation in a later state is new. Therefore, the value of the function should be maximal.

Ψ_R is defined according to (17).

$$\Psi_R(R^C, R^{C'}) = \frac{|\, new_{R^C}(R^C, R^{C'})\,| + |\, del_{R^C}(R^C, R^{C'})\,|}{|\, R^{C'}\,| + |\, del_{R^C}(R^C, R^{C'})\,|} + \frac{\sum\limits_{(r,r')\in alt_{R^C}(R^C, R^{C'})} d_s(S_R(r), S_R(r'))}{|\, R^{C'}\,| + |\, del_{R^C}(R^C, R^{C'})\,|} \tag{17}$$

Identically as (16) the (17) can be treated as one expression due the identical denominators of its components. This denominator will always be higher than the numerator, thus making Psi_R meets **P1**. If no changes has been applied to relations $diff_{R^C}(O, O') = \langle\emptyset, \emptyset, \emptyset\rangle$ then **P2** will be met because the nominator will equal to 0, making the whole expression equals 0. Suppose every relation from the ontology has been changed. In that case, the nominator and the denominator will be equal, making the whole expression equal to 1. This property asserts that Ψ_R meets **P3**.

While maintaining two ontologies and their alignment, when a particular measure (Ψ_C, Ψ_I or Ψ_R) surpasses some assumed threshold (due to the significant changes applied to aligned ontologies), then the appropriate procedure should be used to ensure the validity of the alignment. The choice of these threshold values depends on specific ontology applications and the required level of sensitivity. The choice of these thresholds would produce very different results, and the legitimate question of how to better set their values naturally arises. However, we claim that it is very domain dependent and lies beyond the scope of this article. However, the flexibility of having three different functions describing different types of changes in ontologies may prove helpful in various use cases.

4.2 Methods for updating ontology alignment

This section presents algorithms for updating ontology alignment based on the evolution of mapped ontologies. The procedures may be treated as subsequent steps to calculating the degree of change significance described earlier.

The following section will be split into subsections dedicated to concepts, instances, and relations. Every presented algorithm takes as an input a result of a comparison of two states of the source ontology O_1 and O_1' (utilising functions $diff_C$, $diff_I$, $diff_R$) and a mapping $Align(O_1, O_2)$ between the source ontology (in its initial state O_1) and a target ontology O_2. All algorithms return the updated and revalidated alignment $Align(O_1', O_2)$ compatible with the latest versions of given ontologies.

4.2.1 Updating ontology alignment on the level of concepts

As easily seen in (8), ontology evolution may involve adding new concepts and modifying or deleting them. Therefore, preserving the validity of the alignment $Align_C(O_1, O_2)$ and ensuring its correctness in time $Align_C(O_1', O_2)$ can be divided into three sub-procedures. The first one (Algorithm 1) is a simple procedure of removing all of the mappings that connect concepts deleted from the ontology O. Its computation complexity is low and dependent only on the size of $del_C(C_1, C_1')$, therefore, we can claim that this complexity is linear.

Algorithm 2 addresses adding new concepts to the ontology O_1. Such concepts must be compared with

```
Input  : $diff_C(O_1, O_1')$, $Align_C(O_1, O_2)$
Output: $Align_C(O_1', O_2)$
1 begin
2     $del := \{(c_1, c_2) \in Align_C(O_1, O_2) \mid c_1 \in del_C(C_1, C_1')\}$;
3     $Align_C(O_1', O_2) := Align_C(O_1, O_2) \setminus del$;
4     return $Align_C(O_1', O_2)$;
5 end
```

Algorithm 1 Removing stale mappings of deleted concepts from the existing alignment.

concepts from the target ontology O_2 and checked if they can be aligned. We accomplish this goal by creating an additional set containing concepts mappings taken from the cartesian product of concepts added to O_1 and concepts from O_2 for which the alignment degree function λ_C is higher than the given threshold τ. We incorporate the concept alignment degree function λ_C (Line 2) developed in our earlier publications [28]. However, the method may be replaced with any ontology mapping tool from the literature (e.g., [32]). The algorithm requires comparing all added concepts with concepts from the target ontology, therefore, we can claim that the final complexity of Algorithm 2 is not higher than polynomial. Since the size of $new_C(C_1, C_1')$ is usually small and significantly lower than the size of C_2, therefore the efficiency of the algorithm is satisfactory.

```
Input  : $diff_C(O_1, O_1')$, $Align_C(O_1, O_2)$, $\tau$
Output: $Align_C(O_1', O_2)$
1 begin
2     $adds = \{(c_1', c_2) \mid (c_1', c_2) \in new_C(C_1, C_1') \times C_2 \wedge \lambda_C(c_1', c_2) \geq \tau\}$
3     $Align_C(O_1', O_2) := Align(O_1, O_2) \cup adds$;
4     return $Align_C(O_1', O_2)$;
5 end
```

Algorithm 2 Adding new mappings to concept alignment.

Algorithm 3 addresses the issue of changes applied to structures of concepts. Their new state may entail one of three three situations. The first involves alignments of a particular, changed concept, which may no longer be correct and therefore, must be removed. This issue is addressed in Lines 4-5 where mappings are checked if the value of their alignment degree changed significantly enough (Line 4). If that is the case, then such mappings are removed (Line 5). The second situation addresses violating taxonomic completeness of mappings, which states that if some concept taken from the ontology has a predecessor within the concepts' hierarchy, then they can be both mapped to a concept from the second ontology. If such a situation occurs then the alignment must be completed (Lines 7-11). The opposite situation is checked in Lines 12-15 where the algorithm removes from the alignment excessive mappings of concepts between which there are no taxonomical connections. In Line 16 the algorithm checks all previously unprocessed concepts (track of which is held using $\widetilde{C_2}$ initialised in Line 6) from the target ontology for any additional mapping which should be added to the final alignments.

The described procedure is the most complex. It requires checking if altered concepts from the source ontology O_1 can be mapped into concepts from the target ontology O_2, which entails processing the set of its concepts. Therefore, its complexity is not higher than polynomial with regard to the number of concepts in both ontologies. However, the number of altered concepts in the source ontology O_1 is usually much smaller than the set of concepts in the target ontology O_2. Thus, we can claim that the algorithm depends only on the number of concepts in the target ontology O_2.

```
Input  : $diff_C(O_1, O_1')$, $Align_C(O_1, O_2)$, $\epsilon$, $\tau$
Output: $Align_C(O_1', O_2)$
1 begin
2     $Align_C(O_1', O_2) := Align_C(O_1, O_2)$;
3     for $(c_1, c_1') \in alt_C(C_1, C_1')$ do
4         $dels := \{(c_1, c_2) \in Align_C(O_1, O_2) \mid \exists c_1' \in C_1' : (c_1, c_1') \in alt_C(C_1, C_1') \wedge \mid \lambda_C(c_1, c_2) - \lambda_C(c_1', c_2) \mid \geq \epsilon\}$;
5         $Align_C(O_1', O_2) := Align_C(O_1, O_2) \setminus dels$
6         $\widetilde{C_2} := C_2$
7         if $\exists(b', c_1') \in H_1' \wedge \exists(c_1', c_2) \in Align_C(O_1', O_2)$ then
8             if $\neg\exists(b', c_1) \in H_1$ then
9                 $Align_C(O_1', O_2) := Align_C(O_1', O_2) \cup \{(b', c_2)\}$
                  $\widetilde{C_2} := \widetilde{C_2} \setminus \{c_2\}$
10            end
11        end
12        if $\exists(b, c_1) \in H_1\ ) \wedge \neg\exists(b', c_1') \in H' \wedge \exists(b', c_2) \in Align_C(O_1', O_2)$ then
13            $Align_C(O_1', O_2) := Align_C(O_1', O_2) \setminus \{(b', c_2)\}$
14            $\widetilde{C_2} := \widetilde{C_2} \setminus \{c_2\}$
15        end
16        $Align_C(O_1', O_2) := Align_C(O_1', O_2) \cup \{(c_1', c_2) \mid (c_1', c_2) \in \{c_1'\} \times \widetilde{C_2} \wedge \lambda_C(c_1', c_2) \geq \tau\}$
17    end
18    return $Align_C(O_1', O_2)$;
19 end
```

Algorithm 3 Updating alignments of modified concepts.

4.2.2 Updating ontology alignment on the level of instances

The Algorithm 4 addresses the issue of changes applied to instances of concepts and how they affect their mappings. It requires three elements as input:

1. the instance alignment between a source ontology O_1 in its earlier state and the target ontology O_2: $Align_I(O_1, O_2) = \{AL_{O_1,O_2}(c_1, c_2) \mid (c_1, c_2) \in Align_C(O_1, O_2)\}$
2. the current state $Align_C(O_1', O_2)$ of the alignment on the concept level
3. the result of the difference function on the concept level $diff_C(O_1, O_1')$

Initially (Lines 2-3), the algorithm deletes alignments of instances of removed concepts. Next (Line 4-8), the algorithm generates a set of concept pairs added to the source ontology mapped with concepts from the target ontology. Their instances are checked to see if they can be aligned with each other. Then, the algorithm processes changes applied to the source ontology (Line 9-10)- for every modified concept, a set of corresponding alignments is designated. Every iteration (Lines 11-24) checks if a current concept has attributes removed or added. The mapping degree function λ_I is used, and its outcome is compared with a threshold. In Lines 29-33, the algorithm removes alignments of instances that have been removed. The final section of the algorithm (Lines 34-41) checks if earlier alignments of the altered instance are incorrect - if so, they are removed. The algorithm also checks if any added concepts' instances can be aligned to instances of corresponding concepts in the target ontology, and the updated alignment is returned (Line 42).

To process instances and their valuations in particular concepts, the algorithm must analyze both a set of instances and a set of concepts. Obviously, they are narrowed down only to instances in which valuations have changed. Such operation entails relatively high complexity with regard to the number of changed instances from the ontology change log, sizes of sets of concepts, and sizes of sets of instances. Since the size of the change log is usually not very big, we can claim the algorithm's complexity is quadratic. It depends only on the sizes of the two latter sets.

4.2.3 Updating ontology alignment on the level of relations

The algorithm that updates the ontology alignment on the relation level $Align_R(O_1, O_2)$ to its new state $Align_R(O_1', O_2)$ can be divided into three scenarios: *(i)* removing stale mappings; *(ii)* revalidating and updating preserved mappings; *(iii)* adding new mappings.

The first scenario presented on Algorithm 5 addresses an issue concerning relations that have been removed from the source ontology. Consequently, all their mappings should also be removed (Lines 3). The described algorithm is a straightforward procedure with a small, linear computation complexity. It depends only on the size of the set of deleted relations, which is usually relatively small.

Since modifying relations within a source ontology and adding new relations entail finding new mappings for such relations, the last two scenarios can be implemented in one procedure, which is presented on Algorithm 6.

```
Input  : $Align_I(O_1, O_2), Align_C(O_1', O_2), diff_C(O_1, O_1'), \tau, \epsilon$
Output: $Align_I(O_1', O_2)$
1  begin
2      $del := \{AL_{O_1,O_2}(c_1, c_2) \mid AL_{O_1,O_2}(c_1, c_2) \in Align_I(O_1, O_2) \wedge c_1 \in del_C(O_1, O_1')\}$
3      $Align_I(O_1', O_2) := Align_I(O_1, O_2) \setminus del$
4      $new := \{(c_1', c_2) \mid c_1' \in new_C(O_1, O_1') \wedge (c_1', c_2) \in Align_C(O_1', O_2)\}$
5      for $(c_1', c_2) \in new$ do
6          $AL_{O_1,O_2}(c_1', c_2) := \{(i_1', i_2) \mid i_1' \in c_1' \wedge i_2 \in c_2 \wedge \lambda_I(v_{c_1}^{i_1'}, v_{c_2}^{i_2})) \geq \tau\}$
7          $Align_I(O_1', O_2) := Align_I(O_1', O_2) \cup \{AL_{O_1,O_2}(c_1', c_2)\}$
8      end
9      for $(c_1, c_1') \in alt_C(O_1, O_1')$ do
10         for $AL_{O_1,O_2}(c_1', c_2) \in Align_I(O_1', O_2)\}$ do
11             if $\mid A^{c_1} \mid \neq \mid A^{c_1'} \mid$ then
12                 for $(i_1', i_2) \in AL_{O_1,O_2}(c_1', c_2)$ do
13                     if $\mid \lambda_I(v_{c_1}^{i_1}, v_{c_2}^{i_2}) - \lambda_I(v_{c_1}^{i_1'}, v_{c_2}^{i_2}) \mid \geq \epsilon$ then
14                         $AL_{O_1,O_2}(c_1', c_2) := AL_{O_1,O_2}(c_1', c_2) \setminus \{(i_1, i_2)\}$
15                     end
16                 end
17             end
18             else
19                 $aux = \{i_2 \mid i_2 \in Ins(c_2) \wedge \neg\exists i_1'' : (i_1'', i_2)) \in AL_{O_1,O_2}(c_1', c_2)\}$
20                 for $(i_1', i_2) \in Ins(c_1') \times aux$ do
21                     if $\lambda_I(v_{c_1}^{i_1'}, v_{c_2}^{i_2})) \geq \tau$ then
22                         $AL_{O_1,O_2}(c_1', c_2) := AL_{O_1,O_2}(c_1', c_2) \cup \{(i_1', i_2))\}$
23                     end
24                 end
25             end
26         end
27     end
28 end
```

Algorithm 4 Updating ontology alignment on the instance level.

```
28 begin
29     for $i_1 \in del_I(I, I')$ do
30         for $c_1 \in Ins^{-1}(i_1)$ do
31             $AL_{O_1,O_2}(c'_1, c_2) := AL_{O_1,O_2}(c'_1, c_2) \setminus \{(i_1, i_2) \mid (i_1, i_2) \in AL_{O_1,O_2}(c_1, c_2)\}$
32         end
33     end
34     for $c'_1 \in \{c'_1 \mid c'_1 \in C'_1 \wedge c_1 \in C_1 \wedge I^{c_1} \neq I^{c'_1} \wedge \exists c_2 \in C_2 : (c'_1, c_2) \in Align_C(O'_1, O_2)\}$ do
35         for $AL_{O_1,O_2}(c'_1, c_2) \in Align_I(O'_1, O_2)$ do
36             $dels = \{(i'_1, i_2) \mid (i'_1, i_2) \in AL_{O_1,O_2}(c'_1, c_2) \wedge \lambda_I(v_{c_1}^{i'_1}, v_{c_2}^{i_2}) < \tau\}$
37             $new := \{i'_1 \mid i'_1 \in I'_1 \wedge \neg\exists(i'_1, i_2) \in AL_{O_1,O_2}(c'_1, c_2)\}$
38             $adds = \{(i'_1, i_2) \mid (i'_1, i_2) \in ((new_I(I_1, I'_2) \cap Ins(c'_1) \cup new) \times Ins(c_2) \wedge \lambda_I(v_{c_1}^{i'_1}, v_{c_2}^{i_2}) \geq \tau\}$
39             $AL_{O_1,O_2}(c'_1, c_2) := AL_{O_1,O_2}(c'_1, c_2) \setminus dels \cup adds$
40         end
41     end
42     return $Align_I(O'_1, O_2)$;
43 end
```

Algorithm 4 (continued)

```
Input  : $Align_R(O_1, O_2), diff_R(O_1, O'_1)$
Output: $Align_R(O'_1, O_2)$
1 begin
2     $del := \{(r_1, r_2) \mid (r_1, r_2) \in Align_R(O_1, O_2) \wedge r_1 \in del_R(R^{C_1}, R^{C_1'})\}$;
3     $Align_R(O'_1, O_2) := Align_R(O_1, O_2) \setminus del$;
4     return $Align_R(O'_1, O_2)$;
5 end
```

Algorithm 5 Removing stale mappings of deleted relations from the existing alignment.

The algorithm begins (Lines 3-6) by designating a set of alignments of modified relations (Line 3), which are fed to the appropriate alignment function. The calculated result is confronted with some assumed threshold. If the criterion is not met, such a relation pair is removed from the alignment. Then, the algorithm designates a set of relations from the source ontology which were modified but previously not aligned with any relation from the target ontology (Line 5). This set is then summed with a set of new relations added to the source ontology (Line 6). We do not consider adding new alignments involving previously aligned relations since we assume that the alignment of two ontologies before one evolved is considered sound and complete.

The algorithm starts (Line 7) to search for any new mappings. It is possible to incorporate any alignment procedure - for simplicity, in this paper, we use a mapping method created in our previous publication [28]. It is built on top of a distance function, also used in (17), which can be used to compare relation semantics. The algorithm iterates through the set of relations from the target ontology (Line 8). Every relation is checked if the degree to which it can be aligned to the relation from the target ontology is higher than a threshold value (Line 9). New mappings are added to the alignment (Line 10). The algorithm also adds mappings of relations that are more general than the current one (Lines 13-17).

The algorithm must compare every relation added to or changed in the source ontology with every relation in the target ontology. The alterations of the source ontology are usually relatively small. Therefore, the algorithm's complexity depends mainly on the number of relations in the target ontology but is not higher than quadratic.

5 Experimental verification

5.1 Evaluation procedure

Ontology Alignment Evaluation Initiative (OAEI) [37] is a non-profit organization that, since 2004, has hosted annual campaigns evaluating ontology matching technologies. These recurring events are based on a provided benchmark dataset, consisting of a large number of ontologies grouped into thematic sets (referred to as "tracks") along with the pre-prepared reference alignments, which are treated as the only correct. Alignments designated by campaign participants are confronted with those reference alignments, which allows for calculating Precision, Recall, and F-measure [5].

The provided dataset is widely accepted in the literature as a state-of-the-art benchmark. Therefore, we have also chosen to use it to verify the methods proposed in Section 4.2. However, we have found several limitations of the approach proposed by OAEI when attempting to use it in the context of ontology evolution.

Firstly, we noticed that the OAEI's benchmarks are focused solely on the concept level of ontologies. Issues concerning the level of relations and instances are treated very vaguely. There exist a track pre-prepared solely for the level of instances, however, the artificially introduced complexity (e.g., translating attribute values into different languages) renders it inappropriate in the scenario of ontology evolution. Moreover, to the best of our knowledge, no other ontology dataset focused on relations or instances exists.

```
Input  : $Align_R(O_1, O_2), diff_R(O_1, O_1'), \tau, \epsilon$
Output: $Align_R(O_1', O_2)$
begin
    $Align_R(O_1', O_2) := Align_R(O_1, O_2)$;
    $\widetilde{alt} = \{(r_1, r_2) \mid (r_1, r_2) \in Align_R(O_1, O_2) \wedge (r_1, r_1') \in alt_{RC}(R^{C_1}, R^{C_1'})\}$
    $Align_R(O_1', O_2) := Align_R(O_1', O_2) \setminus \{(r_1, r_2) \mid (r_1, r_2) \in \widetilde{alt} \wedge \mid \lambda_R(r_1, r_2) - \lambda_R(r_1', r_2) \mid > \tau\}$;
    $\widetilde{alt}^+ := \{r_1' \mid (r_1, r_1') \in alt_{RC}(R^{C_1}, R^{C_1'}) \wedge \neg\exists(r_1, r_2) \in Align_R(O_1, O_2)\}$
    $\widetilde{new} = \widetilde{alt}^+ \cup new_{RC}(R^{C_1}, R^{C_1'})$
    for $r_1' \in \widetilde{new}$ do
        for $r_2 \in R^{C_2}$ do
            if $\lambda_R(r_1', r_2) \geq \epsilon$ then
                $Align_R(O_1', O_2) := Align_R(O_1', O_2) \cup \{(r_1', r_2)\}$
            end
        end
        if $\exists r_1'' \in R^{C_1'} : (r_1'' \leftarrow r_1')$ then
            if $\exists(r_1', r_2) \in Align_R(O_1', O_2)$ then
                $Align_R(O_1', O_2) := Align_R(O_1', O_2) \cup \{(r_1'', r_2)\}$
            end
        end
    end
    return $Align_R(O_1', O_2)$;
end
```

Algorithm 6 Revalidating the existing and adding new mappings of relations.

Secondly, we expected that the benchmark data would change and evolve each year to simulate real-world applications of ontologies and reflect constantly changing business requirements. However, no such changes were introduced in the dataset provided by OAEI - only new benchmark ontologies have been added to the pool.

The two limitations entailed that we could not directly use datasets provided by OAEI in our experiment. Therefore we decided to propose a different approach. At first, we selected a "Conference Track" [39] from the OAEI dataset, which includes 16 ontologies from the domain of conference organization. Then, we applied a series of semi-random changes using a set of pre-prepared evolution scenarios. The modification of the ontology can consist of 3 different operations: adding a new concept and removing or modifying an existing concept. Removing a concept does not require any additional explanations. The indicated number of concepts is chosen in a random way and simply deleted from the ontology. Adding a new concept is more complicated because this operation requires preserving a general sense and consistency of the ontology. For this purpose, we have selected one ontology related to the conference and the indicated number of concepts have been copied from the chosen to the evolving ontology. As it was mentioned in Section 3.2 by the modification of concepts we understand the changes made in their structure in terms of their contexts. In our experiment, we apply random mutation to the concepts' structure by adding or removing attributes. Similarly to the adding concepts, the attributes from the related ontology are drawn and added to the evolving ontology. The process of removing attributes is self-explanatory.

The number and type of alteration depend on the aim of the experiment. In the first one, the number of modifications has been directly indicated by predefined experimental scenarios. The purpose of the second one is to verify how the number of modifications in source ontologies influences the alignments. Thus, in the subsequent steps, we applied more changes by adding new concepts. In the beginning, only a 10% change has been applied and in the end, the number of concepts in the source ontology has been doubled. The last experiment applied a scenario based on adding 10 related concepts.

These changes resulted in different versions of initial ontologies, which can be treated as a simulation of ontology evolution. We split the experiment into three parts. Each phase started by designating an initial alignment between different ontologies. We used the LogMap system [16], [38], which according to OAEI is one of the most prominent alignment systems. Next, for each modified ontology pair their initial alignment has been revalidated using algorithms proposed in Section 4.2 which were then compared with a mapping determined from scratch using LogMap. The main idea of the experiment is presented in Fig. 1.

At this point, we needed to decide which ontology alignment assessment method we should use to evaluate obtained mappings. The most widely used Precision, Recall, and F-measure cannot be used since there is no reference alignment. Moreover, such reference alignment cannot be created manually due to the random nature of the introduced changes. Naturally, a small portion of alignments could be computed manually by domain experts, and those can then be used for evaluation, however, such an approach is always biased and may render inconclusive results.

Therefore, we needed a method that does not require a pre-prepared reference alignment. We claim that when two similarly-sized alignments of ontologies need to be compared without knowing what correct correspondences are, it is worth analyzing the properties of mapped concepts and their placement within ontologies. Based on such information, it is possible to judge the quality of these alignments. Thus, we used a novel approach to assessing

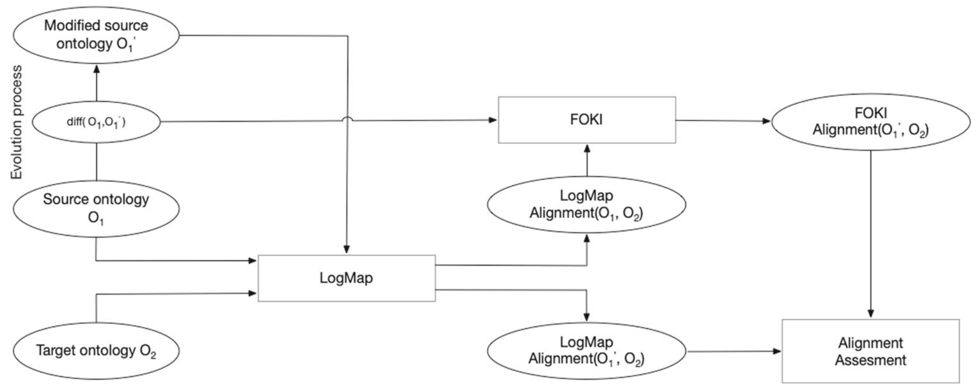

Fig. 1 The general idea of the experiment

ontology mappings to evaluate the quality of alignments collected during the experimental verification. It was proposed in our previous publication [29] and includes two measures ξ_D and ξ_C. Both have been useful compared to the most widely used Precision, Recall, and F-measure. Here, we will only provide baseline definitions for completeness purposes; for details, please refer to [29].

The first one, ξ_D, utilizes a criterion based on the depth of classes in the hierarchy of mapped ontologies. It is built on top of an obvious remark that states that the deeper a concept is placed within a taxonomy, the more detailed its description and the more refined knowledge it expresses. This entails that finding correspondences of such elements is more complicated than finding mappings of classes located shallow in the hierarchy. Therefore, we claim that alignments of concepts deeper in the hierarchy should be treated as more valuable than alignments of general concepts, which represent generic facts.

Formally, the depth-based measure ξ_D which reflects the described considerations, is defined in the following way:

$$\xi_D(O_1, O_2, Align_C(O_1, O_2)) = \sum_{(c_1,c_2)\in Align_C(O_1,O_2)} \gamma_D\,(O_1, O_2, Align_C(O_1, O_2), (c_1, c_2)) \tag{18}$$

where γ_D is defined as follows:

$$\gamma_D(O_1, O_2, Align_C(O_1, O_2), (c_1, c_2)) = \begin{cases} \frac{1}{Depth(O_1)-Depth(O_1,c_1)+1} + \frac{1}{Depth(O_2)-Depth(O_2,c_2)+1} \\ \text{if } (c_1, c_2) \in Align_C(O_1, O_2) \\ 0 \quad \text{otherwise} \end{cases} \tag{19}$$

As with every tool, this also has some limitations. Let us assume the existence of a specific concept that has been split into two neighboring concepts located at the same level of the taxonomy. Suppose the mapping is enriched with two mappings of such new concepts. In that case, they will be treated as equally relevant due to the position of the connected concepts in the hierarchy. The proposed measure will, therefore, not take into account the enrichment of the description in the ontology. However, this is directly due to its specificity based on the depth of the hierarchy.

The second measure ξ_C is based on the continuity of mapped classes. Lets assume a situation in which we have two alignments of ontologies $O_1 = (C_1, H_1, R^{C_1}, I_1, R^{I_1})$ and $O_2 = (C_2, H_2, R^{C_2}, I_2, R^{I_2})$. The first one contains entries with concepts from O_1 that belong to the same subtree in H_1. A second alignment contains concepts from O_1, which are spread across the hierarchy H_1. In other words, one of the given alignments connects concepts from O_1, which are closely clustered.

In contrast, the second alignment connects concepts that are likely unrelated. Assuming the existence of a reference alignment, both competing alignments could have comparable values of Precision and Recall. However, the former alignment should be treated as more valuable from the user's point of view. In other words, when an alignment contains mappings of unrelated classes, it may look chaotic and incoherent. In contrast, a smaller alignment that covers a focused ontologies fragments (from the taxonomic point of view) can bring more benefits to the end user.

In order to define ξ_C we introduce some auxiliary notions:

- $Subtree(O_1, Align_C(O_1, O_2))$ denotes a set of subtrees of concepts from the ontology O_1 that participates in the alignment $Align_C(O_1, O_2)$. It is calculated using the formula:

$$Subtree(O_1, Align_C(O_1, O_2)) = \bigcup_{(c_1,c_2)\in Align_C(O_1,O_2)} \{Subtree(O_1, c_1)\} \tag{20}$$

- $\gamma_C(t) = \mid t \mid^2$ is the rating function that scores a given subtree with a certain number of points, where $|t|$ is the number of classes in subtree t. The square in the definition of γ_C is used to increase the difference between scores obtained by two subtrees.

Having the above, we defined the measure ξ_C in the formula below:

$$\xi_C(O_1, O_2, Align_C(O_1, O_2)) = \sum_{t \in SubTrees(O_1, Align_C(O_1,O_2))} \gamma_C(t) + \sum_{t \in SubTrees(O_2, Align_C(O_1,O_2))} \gamma_C(t), \tag{21}$$

Additionally, along with the measures ξ_D and ξ_C we have used the accuracy measure, which indicates how similar results are determined by both approaches. It is calculated as the number of common concept mappings between two ontologies divided by the number of all mappings found by both methods:

$$Accuracy = \frac{\mid Align_{LogMap}(O_1', O_2) \cap Align_{FOKI}(O_1', O_2) \mid}{\mid Align_{LogMap}(O_1', O_2) \cup Align_{FOKI}(O_1', O_2) \mid} \tag{22}$$

The measures defined on (18) and (21) form a novel framework for assessing the quality of ontology alignment. Due to the limited space available, we cannot present either example or their comprehensive analysis. For details, please refer to [29].

Reference alignments and measures of Precision, Recall, and F-measure are the ideal solution when some initial tests of ontology alignments are performed (like the ones organized by OAEI). This makes such an approach unusable in the context of this article since there are neither reference ontologies nor reference alignments created to simulate ontology evolution and assess the updated versions of their mappings.

Bear in mind that none of the measures defined on (18), (21) and (22) are not intended to assess the correctness of mappings. They can be used to assess a mapping quality based on the taxonomic structure of participating ontologies. Therefore, using ξ_D and ξ_C to verify algorithms from Section 4.2 emphasizes that the main purpose of all of the developed procedures is not outperforming any baseline ontology alignment methods in terms of Precision, Recall, and F-measure. Their fundamental goal is updating the ontology alignment designated before any changes have been introduced to ontologies. They are expected to return alignments with at least similar quality to the ones designated from scratch by ontology alignment systems that require processing whole ontologies. In other words, the main goal of the experiment illustrated in Fig. 1 is simulating ontology evolution and comparing the quality of updated alignments (in terms of defined measures ξ_C, ξ_D and *Accuracy*) with alignments designated from scratch by LogMap.

In the first part of the experimental procedure, we wanted to check how different changes in source ontologies influence changes in alignment. For this purpose, we applied different types of modification in source ontologies, like adding new concepts, removing existing concepts, or modifying them. The second part of the experiment aims to observe how the different number of modifications in the source ontologies affects the final alignment. In the last part, we examine alterations in different source ontologies.

5.2 Different type of modifications in the source ontology

In the first experiment, we aimed to verify how different modifications of a source ontology influence the alignments between two ontologies. The experiment was composed of several steps. Initially, for two ontologies, an alignment between them has been created using LogMap. Then we applied random alterations to the source ontology according to predefined scenarios. After applying the modifications, we have launched the algorithms from Section 4.2 in order to adjust the initial alignments. Simultaneously, a new alignment has been designated once more using LogMap. All alignments have been compared using the assessment functions ξ_D and ξ_C, and the accuracy measure.

In the procedure we used the *the Conference Track* [39] from *OAEI* datasets. It contains 16 ontologies from the conference organization domain. Due to heterogenous character of origin of these ontologies, they are suitable for ontology matching task. Moreover, the nature of the domain is simple to understand, which allows for easy initial verification of the obtained results. From the whole set we have randomly chosen the *CMT*[1] ontology as a target ontology, and the *confOf*[2] ontology as a source ontology. The collected results for different scenarios are presented in Table 1.

This experiment allows us to compare alignments designated from scratch by LogMap and the FOKI framework. The accuracy values from Table 1 showed that both approaches determined different sets of mappings. The assessment of the alignment quality determined by both methods has been estimated by the defined measures ξ_D and ξ_C. Therefore, we collected four samples of data: LogMap ξ_D, LogMap ξ_C, FOKI ξ_D and FOKI ξ_C. In this experiment we wanted to statistically verify two hypothesis stating that: (i) means of LogMap ξ_D and FOKI ξ_D are equal; and (ii) means of LogMap ξ_C and FOKI ξ_C are equal.

In order to achieve this goal, we assumed the significance level $\alpha = 0.05$. Then some prerequisites had to be checked. Firstly, we verified if the samples LogMap ξ_D,

[1] http://oaei.ontologymatching.org/2021/conference/data/cmt.owl

[2] http://oaei.ontologymatching.org/2021/conference/data/confof.owl

Table 1 Different scenarios for a single pair of ontologies

Scenario	LogMap ξ_D	LogMap ξ_C	FOKI ξ_D	FOKI ξ_C	Accuracy
No changes	2.949	16.00	2.949	16.00	1.000
Removing 5 concepts	1.699	8.00	1.699	8.00	1.000
Adding 5 related concepts	4.250	29.00	3.550	23.00	0.889
Modifying 5 concepts	2.366	14.00	1.699	10.00	0.571
Adding and removing 5 concepts	3.466	22.00	2.099	13.00	0.625
Adding and modifying 5 concepts	3.333	22.00	2.633	17.00	0.666
Modifying and removing 5 concepts	1.116	6.00	0.450	4.00	0.333
Adding, removing and modifying 5 concepts	2.183	16.00	0.850	9.00	0.333
Removing 20 concepts	1.283	4.00	1.283	4.00	1.000
Adding 20 related concepts	3.199	17.00	3.449	18.00	0.778
Modifying 20 concepts	1.700	8.00	0.450	6.00	0.222
Removing 40 concepts	0.000	0.00	0.000	0.00	1.000
Adding 40 related concepts	3.883	20.00	2.783	15.00	0.462

LogMap ξ_C, FOKI ξ_D and FOKI ξ_C come from the normal distribution. We have used the Shapiro-Wilk test. For all data, we received $p-value$ greater than 0.05 and statistical test values: 0.971, 0.971, 0.974, 0.978, respectively, which allows us to conclude that all the samples come from the normal distribution. Then, we used the Fisher test, which confirmed the equality of variances of samples LogMap ξ_D and FOKI ξ_D ($F = 1.079$, $p - value = 0.89$) and the equality of variances of samples LogMap ξ_C and FOKI ξ_C ($F = 1.575$, $p - value = 0.43$).

Eventually, we have chosen the t-test. For samples of LogMap ξ_D and FOKI ξ_D we obtained $t - value = 1.22$ and $p - value = 0.234$. For samples ξ_C and FOKI ξ_C we obtained $t - value = 1.01$ and $p - value = 0.32$. Such results confirm that using the FOKI framework determines mappings of at least not worse quality than LogMap in terms of the assumed quality measures ξ_D and ξ_C.

5.3 Different number of modifications in the source ontology

The result obtained in the second experiment showed how modifications of the source ontology influence the alignments. Identically as in the Section 5.2 we used the *the Conference Track* [39] from *OAEI* datasets, which includes 16 ontologies from the conference organization domain. From those 16 ontologies, we randomly chose two ontologies different than the ones used in the experiment described in Section 5.2 - the *edas*[3] and *ekaw*[4] have been picked respectively as the source and the target ontology.

The source ontology has been modified randomly by applying increasing changes in the subsequent steps. In the beginning, only a 10 % change has been applied, and in the end, the number of concepts in the source ontology has been doubled. The obtained results are presented in Table 2.

The results presented in Table 2 confirmed the hypothesis that the number of alterations in source ontologies does not influence the alignments determined by our approach and LogMap. The very high accuracy measure proves it. Identically as in the experiment from Section 5.2 we collected four samples of data: LogMap ξ_D, LogMap ξ_C, FOKI ξ_D and FOKI ξ_C. All the statistical analyses have been done for the assumed significance level $\alpha = 0.05$.

Initially, we checked the normality distribution of all of them, obtaining the $p - value$ greater than the assumed significance level $\alpha = 0.05$ and statistic values respectively equal to: 0.93, 0.951, 0.859, 0.856. This proved that the samples LogMap ξ_D, LogMap ξ_C, FOKI ξ_D and FOKI ξ_C come from the normal distribution.

Next, we checked the variances of considered samples using the Fisher test. For the pair of samples LogMap ξ_D

Table 2 Different number of modification in the source ontology *edas*

Amount of changes	LogMap ξ_D	LogMap ξ_C	FOKI ξ_D	FOKI ξ_C	Accuracy
10%	7.095	210.00	7.095	210.00	1.00
20%	7.879	229.00	7.879	229.00	1.00
30%	8.079	254.00	8.079	254.00	1.00
40%	8.412	259.00	8.888	277.00	0.952
50%	8.495	275.00	8.721	268.00	0.913
60%	8.495	275.00	8.721	268.00	0.913
70%	8.579	294.00	8.255	294.00	0.920
80%	9.371	318.00	8.538	273.00	0.926
90%	9.371	318.00	8.538	273.00	0.926
100%	8.705	298.00	8.538	273.00	0.962

[3] http://oaei.ontologymatching.org/2021/conference/data/edas.owl
[4] http://oaei.ontologymatching.org/2021/conference/data/ekaw.owl

and FOKI ξ_D the test yielded values $p - value = 0.49$ and $F = 1.61$. For the pair LogMap ξ_C and FOKI ξ_C we obtained values $p - value = 0.287$ and $F = 2.09$. It allows us to claim that the variances of tested samples are equal.

Eventually, all of the t-test requirements were met. The t-test calculated for samples LogMap ξ_D and FOKI ξ_D equals 0.45 (with a $p - value$ equal to 0.656). For samples LogMap ξ_C and FOKI ξ_C the t-test yielded value equal to 0.805 (and the $p - value = 0.43$). Thus, we cannot reject the hypothesis that LogMap and our approach determine mappings with the same quality in terms of the assumed quality measures ξ_D and ξ_C.

5.4 Different source ontologies

In the last experiment, we have chosen one target ontology $confOf$. We have applied the same modifications in the remaining ontologies from the conference track. The altered ontologies have served as source ontologies according to Fig. 1. The obtained results are presented in Table 3.

Analogous to the prior experiments, we have used the accuracy measure, which showed that both approaches determined a similar, but not identical, set of mappings. The assessment of the quality of the alignment determined by both methods has been estimated by the measures ξ_D and ξ_C defined in Section 5.1, which provided us with four samples LogMap ξ_D, LogMap ξ_C, FOKI ξ_D and FOKI ξ_C. As in Sections 5.2 and 5.3 all the statistical analyses have been made for the significance level of $\alpha = 0.05$.

In the first step, we checked the normality distribution of samples. The Shapiro-Wilk test returned values higher than the assumed α for samples LogMap ξ_D and FOKI ξ_D (0.4625 and 0.2, respectively), which proves that both samples come from the normal distribution. However, the $p - value$ calculated for samples LogMap ξ_C and FOKI ξ_C are lower than the assumed α. It allows us to claim that those samples do not come from the normal distribution.

The Fisher test confirmed the equality of variances of samples LogMap ξ_D and FOKI ξ_D. It yielded a value the $F = 1.0663$ and the $p - value = 0.91$, which consequently allowed us to use the t-test. The collected result of the test is equal to 0.21. The $p - value = 0.983$ proves that both compared methods determined aliments with the same quality concerning the measure ξ_D.

For non-normal samples LogMap ξ_C and FOKI ξ_C, we used the non-parametric Mann-Whitney U test. We have obtained $p - value$ equal 0.729. It means that the LogMap and the FOKI both determine mappings with statistically equal quality in terms of the measure ξ_C.

5.5 Discussion

All of the conducted experiments and collected results unequivocally show that the procedures described in Section 4 of the following paper return promising results. As shown in Tables 1, 2 and 3, they do not strictly outperform LogMap in terms of Accuracy, a measure based on the depth of classes or measure based on the continuity of mapped classes. However, we statistically proved that our approach determines mappings not worse in terms of the assumed quality measures ξ_D and ξ_C than LogMap.

The analysis of Table 1 allows us to check our approach for edge cases. If nothing has changed or almost all concepts have been removed, Accuracy equals 1. Additionally, our approach gives entirely consistent results to LogMap in case of removing concepts. The ontology evolution scenario based on adding concepts is a more complicated procedure; therefore, both verified methods differ slightly. However,

Table 3 Different source ontologies

Source ontology	LogMap ξ_D	LogMap ξ_C	FOKI ξ_D	FOKI ξ_C	Accuracy
confious	1.233	26.000	1.233	24.000	0.648
conference	9.555	157.000	9.412	144.000	0.770
cmt	6.983	39.000	6.983	39.000	0.875
crs	7.683	100.000	7.683	100.000	0.875
edas	9.017	94.000	9.017	94.000	0.552
ekaw	11.845	220.000	11.845	220.000	0.700
iasted	5.726	55.000	5.726	55.000	0.530
linklings	4.833	45.000	6.117	47.000	0.500
micro	7.667	61.000	7.467	54.000	0.840
MyReview	8.117	55.000	8.117	55.000	0.830
OpenConf	7.567	48.000	6.983	44.000	0.550
paperdyne	7.767	57.000	7.850	55.000	0.710
pcs	5.950	44.000	5.950	44.000	0.810
sigkdd	6.250	43.000	6.083	40.000	0.860

in our previous work [22], we have noticed that LogMap does not consider attributes and their modifications in its alignment determination process. Thus, LogMap and our developed procedure are significantly different in the case of concept modification.

Table 2 gives us rather apparent indications that the more changes applied, the smaller value of Accuracy (alignments determined by our approach and LogMap differ more), and the higher value of quality measures ξ_D and ξ_C. These quality measures are based on the depth of classes and the continuity of mapped classes, respectively. Their higher value means more specific and valuable alignment. Adding a new concept enriches the ontology, which entails richer alignment. This part of the experiment also demonstrated the usefulness of defined quality measures in situations where Precision, Recall, and F-measure can not be calculated because of the absence of the reference alignment.

The alignments determined by tested methods are sometimes not consistent. We can conclude that the bigger the source ontology, the more significant the difference between alignments. Table 3 contains ontologies with different numbers of concepts - from 14 to 141. A higher Accuracy value has been reached for smaller ontologies with a number of concepts lower than 30 (e.g., CRS, PCS, or CMT) than for bigger ontologies with a number of concepts higher than 100 (e.g., Iasted, Edas).

In conclusion, our results allow us to claim that it is possible to update ontology alignment using only information about the evolution of ontologies and not whole ontologies. This property can be very valuable in practical applications of ontologies. Re-launching alignment procedures from scratch can be too time-consuming, proving the usefulness of the developed procedures, which can be especially important if the size of ontologies is big.

6 Future works and summary

In this paper, we describe an expansion of our Framework for Ontological Knowledge Integration (FOKI) with novel methods for maintaining ontology alignment during the evolution of mapped ontologies. We have created the set of algorithms, for the levels of concepts, relations, and instances that can assert and preserve the validity of mappings of two ontologies. Consequently, there is no need to conduct an ontology mapping procedure from scratch if any of the participating ontologies change.

The developed algorithms were experimentally verified using the state-of-the-art dataset provided by the Ontology Alignment Evaluation Initiative. The collected results were evaluated using two novel approaches to assessing the quality of ontology alignments- the measure built on top of the criterion based on the depth of the mapped classes and the measure involving the criterion based on the continuity of mapped classes. The experiments showed the usefulness of our framework, which has been proved by the statistical analysis of the collected results. Our methods return high-quality alignments, similar to baseline ontology alignment methods.

Our upcoming research involves incorporating fuzzy logic into our framework. We intend to use it to designate alignments between two ontologies. During the alignment process, the interpretation of similarity between elements from ontologies is not always clear. High or low values of similarity or alignment degree measure between two elements from two ontologies do not always support a decision about their alignment. Fuzzy inference rules allow for overcoming the issue of such uncertainty.

Furthermore, we plan to extend the experiments to analyze the proposed approach's scalability. Using relatively small ontologies from the OAEI datasets proved the usefulness of our ideas, however, using more large-scale datasets is an obvious next step of our research.

Acknowledgements This research project was supported by grant No. 2017/26/D/ST6/00251 from the National Science Centre, Poland.

Declarations

All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Cardoso SD, Da Silveira M, Pruski C (2020) Construction and exploitation of an historical knowledge graph to deal with the evolution of ontologies. Knowl-Based Syst 105508
2. Destro JM, Reis JC, Torres R, Ricarte I (2019) Ontology Changes-Driven Semantic Refinement of Cross-Language Biomedical Ontology Alignments. SeWeBMeDa@ISWC
3. Dinh D, Reis JC, Pruski C, Silveira M, Reynaud C (2014) Identifying relevant concept attributes to support mapping maintenance under ontology evolution. J Web Semant 29:53–66
4. Euzenat J, Meilicke C, Stuckenschmidt H, Shvaiko P, Trojahn C (2011) Ontology alignment evaluation initiative: six years of

experience. In: Journal on data semantics XV. Springer, Berlin, pp 158–192

5. Euzenat J (2007) Semantic precision and recall for ontology alignment evaluation. Ijcai 7:348–353
6. Gene Ontology Consortium (2019) The gene ontology resource: 20 years and still GOing strong. Nucleic Acids Res 47(D1):D330–D338
7. Groß A, Hartung M, Thor A, Rahm E (2012) How do computed ontology mappings evolve? - A case study for life science ontologies. EvoDyn@ISWC
8. Flouris G, Manakanatas D, Kondylakis H, Plexousakis D, Antoniou G (2008) Ontology change: classification and survey. Knowl Eng Rev 23(2):117
9. Harrow I, Balakrishnan R, Jimenez-Ruiz E, Jupp S, Lomax J, Reed J, Romacker M, Senger C, Splendiani A, Wilson J, Woollard P (2019) Ontology mapping for semantically enabled applications. Drug Discov Today 24(10):2068–2075
10. Hartung M, Kirsten T, Rahm E (2008) Analyzing the evolution of life science ontologies and mappings. In: International workshop on data integration in the life sciences. Springer, Berlin, pp 11–27
11. Hartung M, Groß A, Rahm E (2013) COnto–Diff: generation of complex evolution mappings for life science ontologies. J Biomed Inform 46(1):15–32
12. Hnatkowska B, Kozierkiewicz A, Pietranik M (2020) Semi-Automatic Definition of attribute semantics for the purpose of ontology integration. IEEE Access 8:107272–107284
13. Hnatkowska B, Kozierkiewicz A, Pietranik M, Truong HB (2022) Hybrid approach to designating ontology attribute semantics. In: Conference on computational collective intelligence technologies and applications. Springer, Cham, pp 351–363
14. Hnatkowska B, Kozierkiewicz A, Pietranik M (2022) Fuzzy logic framework for ontology instance alignment. In: International conference on computational science. Springer, Cham, pp 653–666
15. Huntley RP, Sawford T, Martin MJ, O'Donovan C (2014) Understanding how and why the Gene Ontology and its annotations evolve: the GO within UniProt. GigaScience 3(1):2047–217X
16. Jiménez-Ruiz E, Grau BC, Zhou Y (2011) LogMap 2.0: towards logic-based, scalable and interactive ontology matching. In: Proceedings of the 4th international workshop on semantic web applications and tools for the life sciences, pp 45–46
17. Keshavarzi A, Kochut KJ (2020) KGDiff: Tracking the evolution of knowledge graphs. In: 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI), Las Vegas, NV, USA, vol 2020, pp 279–286, https://doi.org/10.1109/IRI49571.2020.00047
18. Khattak AM, Latif K, Lee S (2013) Change management in evolving web ontologies. Knowl-Based Syst 37:1–18
19. Khattak AM, Pervez Z, Khan WA, Khan AM, Latif K, Lee SY (2015) Mapping evolution of dynamic web ontologies. Inf Sci 303:101–119
20. Kondylakis H, Plexousakis D (2013) Ontology evolution without tears. J Web Semant 19:42–58
21. Kozierkiewicz A, Pietranik M (2019) Updating ontology alignment on the concept level based on ontology evolution. In: Welzer T, Eder J, Podgorelec V, Kamišalić Latifić A (eds) Advances in Databases and Information Systems. ADBIS 2019. Lecture Notes in Computer Science, vol 11695. Springer, Cham, https://doi.org/10.1007/978-3-030-28730-6_13
22. Kozierkiewicz A, Pietranik M (2019) Triggering ontology alignment revalidation based on the degree of change significance on the ontology concept level. In: Abramowicz W, Corchuelo R (eds) Business information systems. BIS 2019. Lecture notes in business information processing, vol 353. Springer, Cham, https://doi.org/10.1007/978-3-030-20485-3_11
23. Kozierkiewicz A, Pietranik M (2020) Updating ontology alignment on the relation level based on ontology evolution. In: Proceedings of the 15th international conference on evaluation of novel approaches to software engineering - Volume 1: ENASE, 2020, pp 241–248, https://doi.org/10.5220/0009142002410248
24. Kozierkiewicz A, Pietranik M, Nguyen LTT (2020) Updating ontology alignment on the instance level based on ontology evolution. In: Hartmann S, Küng J, Kotsis G, Tjoa AM, Khalil I (eds) Database and expert systems applications. DEXA 2020. Lecture notes in computer science, vol 12392. Springer, Cham, https://doi.org/10.1007/978-3-030-59051-2_20
25. de Moor A, De Leenheer P, Meersman R (2006) DOGMA-MESS: a meaning evolution support system for interorganizational ontology engineering. In: Schärfe H, Hitzler P, Øhrstrøm P (eds) Conceptual structures: inspiration and application. ICCS 2006. Lecture notes in computer science, vol 4068. Springer, Berlin, https://doi.org/10.1007/11787181_14
26. Ochieng P, Kyanda S (2018) Large-scale ontology matching: state-of-the-art analysis. ACM Comput Surv (CSUR) 51(4):1–35
27. Pernischova R (October 2019) The butterfly effect in knowledge graphs: predicting the impact of changes in the evolving web of data Doctoral consortium at ISWC 2019, auckland, 26 october 2019 - 30
28. Pietranik M, Nguyen NT (2014) A Multi-atrribute based framework for ontology aligning. Neurocomputing 146:276–290. https://doi.org/10.1016/j.neucom.2014.03.067
29. Pietranik M, Kozierkiewicz A, Wesołowski M (2020) Assessing ontology mappings on a level of concepts and instances. IEEE Access 8:174845–174859
30. Reis JC, Pruski C, Silveira M, Reynaud C (2014) Understanding semantic mapping evolution by observing changes in biomedical ontologies. J Biomed Inf 47:71–82
31. Sassi N, Jaziri W, Alharbi S (2016) Supporting ontology adaptation and versioning based on a graph of relevance. J Exp Theor Artif Intell 28(6):1035–1059. https://doi.org/10.10800952813X.2015.1056239
32. Shvaiko P, Euzenat J, Jiménez-Ruiz E, Cheatham M, Hassanzadeh O (2018) 2018 CEUR Workshop proceedings 2288. In: Proceedings of the 13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference, OM@ISWC 2018, Monterey, CA, USA, October 8, 2018. CEUR-WS.org 2018
33. Sioutos N, de Coronado S, Haber MW, Hartel FW, Shaiu WL, Wright LW (2007) NCI Thesaurus: a semantic model integrating cancer-related clinical and molecular information. J Biomed Inf 40(1):30–43
34. Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, Lewis S (2007) The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. Nat Biotechnol 25(11):1251–1255
35. Yamamoto VE, dos Reis JC (2019) Updating ontology alignments in life sciences based on new concepts and their context. In: SeWeBMeDa@ ISWC, pp 16–30
36. Zekri A, Brahmia Z, Grandi F, Bouaziz R (2017) Temporal schema versioning in τOWL: a systematic approach for the management of time-varying knowledge. J Decis Syst 26(2):113–137
37. http://oaei.ontologymatching.org/
38. https://www.cs.ox.ac.uk/isg/tools/LogMap/
39. http://oaei.ontologymatching.org/2021/conference/

Dr. Marcin Pietranik received his M.Sc and Ph.D degrees in computer science in 2008 and 2014 respectively and since 2016 he is an assistant professor in Wrocław University of Science and Technology. His scientific interests span across knowledge integration and topics related to ontology management (especially ontology evolution and ontology alignment). He has authored or co-authored over 45 articles and co-organized several conferences. He also has practical background and a strong experience in modern web technologies used within medical projects.

Prof. Adrianna Kozierkiewicz is an associate professor at Wroclaw University of Science and Technology. She received her Ph.D in 2011 and D.Sc. in 2022. Her research interests include ontologies, knowledge management, recommendation systems, consensus theory and other artificial intelligence applications. She is an author and coauthor of over 70 publications in prestigious journals and proceedings of international conferences. She is also an editor of a book entitled: "*Modern approaches for intelligent information and database systems*". She has also been a chair of several conferences, member of program committee of many conferences and member of review boards of some journals. For her research projects, she has been awarded with a few scholarships and grants.