



Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins



Mapping evolution of dynamic web ontologies

A.M. Khattak^a, Z. Pervez^b, W.A. Khan^c, A.M. Khan^d, K. Latif^e, S.Y. Lee^{c,*}

^a College of Technological Innovations, Zayed University, United Arab Emirates

^b School of Computing, University of the West of Scotland, United Kingdom

^c Department of Computer Engineering, Kyung Hee University, Republic of Korea

^d Department of Computer Science, Innopolis University, Russia

^e School of Electrical Engineering and Computer Science, NUST, Pakistan

ARTICLE INFO

Article history:

Received 31 October 2013

Received in revised form 11 December 2014

Accepted 27 December 2014

Available online xxx

Keywords:

Ontology change

Change management

Change history

Mapping reconciliation

ABSTRACT

Information on the web and web services that are revised by stakeholders is growing incredibly. The presentation of this information has shifted from a representational model of web information with loosely clustered terminology to semi-formal terminology and even to formal ontology. Mediation (i.e., mapping) is required for systems and services to share information. Mappings are established between ontologies in order to resolve terminological and conceptual incompatibilities. Due to new discoveries in the field of information sharing, the body of knowledge has become more structured and refined. The domain ontologies that represent bodies of knowledge need to be able to accommodate new information. This allows for the ontology to evolve from one consistent state to another. Changes in resources cause existing mappings between ontologies to be unreliable and stale. This highlights the need for mapping evolution (regeneration) as it would eliminate the discrepancies from the existing mappings. In order to re-establish the mappings between dynamic ontologies, the existing systems require a complete mapping process to be restructured, and this process is time consuming. This paper proposes a mapping reconciliation approach between the updated ontologies that has been found to take less time to process compared to the time of existing systems when only the changed resources are considered and also eliminates the staleness of the existing mappings. The proposed approach employs the change history of ontology in order to store the ontology change information, which helps to drastically reduce the reconciliation time of the mappings between dynamic ontologies. A comprehensive evaluation of the performance of the proposed system on standard data sets has been conducted. The experimental results of the proposed system in comparison with six existing mapping systems are provided in this paper using 13 different data sets, which support our claims.

© 2015 Published by Elsevier Inc.

1. Introduction

The increasing amount of information available on the web places a heavy computational load on the systems that are designed to access, interpret, manipulate, maintain, merge, integrate, infer, and mine this information [21]. The fundamental requirement of information exchange among applications, systems, system agents, and web services is the development of a

* Corresponding author. Tel.: +82 31 201 2514; fax: +82 31 202 2520.

E-mail address: sylee@oslub.khu.ac.kr (S.Y. Lee).

consistent and comprehensive model for knowledge representation, which is essential for the sharing of knowledge pertaining to research outcomes, sharing information among independent organizations [6], and the exchange of information among healthcare systems [31] and among heterogeneous systems and services [3]. In order to make the sharing of information possible, there is a need to model the information more appropriately while preserving its semantics.

Ontology provides a formal structure (model) with semantics with regard to how an expert perceives the domain of interest. Ontology is defined as *a formal, explicit specification of a shared conceptualization*. Ontology is the main source of semantic web information and its services, which helps to clearly define the meaning of resources and achieve a better understanding of the work that is shared between a human and computer systems [35,40]. Service-Oriented Architecture (SOA) and Semantic Web Services Technology are becoming more mature and are now widely used [10]. The meaningful information and the machine interpretable information that is contained in ontology helps to create semantic web services that are automated with regard to service discovery, selection, and interoperability [15].

Current web information can be viewed as the evolution of traditional web information, which ranges from a collection of web pages to the integration of those pages with services that these sites can use to interoperate with one another. Interoperability is both collaborative and multifaceted and is needed to overcome the problems of incompatibilities among organizations, structures, data, architecture, services, and business rules [51]. However, since the data, architecture, and services are usually provided by autonomous parties, often high interface, structural, and semantic heterogeneities exist with regard to information storage and exchange [8,14,18,20,21,26,32,35,37,43,44,49,56,63]. In order to overcome this issue, we utilize the value of data and schema mapping [8,11,18,32,37,44,47,49,52]; in other words, the mapping among schema or ontology elements is the definition of semantic relatedness. Use of ontology in systems dealing with information extraction from a large and complex structured source of information and web services can yield valuable results [4,8,11,18,20,32,47,59]. The increased use of ontology in Information Systems and Knowledge Sharing Systems also increases the significance of ontology maintenance [21,37]. However, the large and complex structure and the decentralized nature of the web compel communities to create their own ontologies to represent information [14,21,59]. Thus, mediation among distributed and autonomous sources is required for exchange of information [8,18,21,32,44,49,63].

The number of information sources is increasing significantly, and this increases the importance of having a sophisticated mechanism to extract information and to manage the heterogeneity among these information sources. Mediation (mapping) is used to align two or more ontologies (information sources) for the purpose of information sharing [5,8,32,35,44,46,49,57,63]. These mappings are generated by mapping systems with two main concerns: accuracy and efficiency (the time required to produce the mappings). Existing mapping systems, such as Falcon [32], FOAM [18], Lily [63], AgreementMaker [12,13], Prompt [49], H-Match [8], and MAFRA [44], are currently considered the best matching and mapping systems. These systems consume a lot of time when mapping large knowledge databases such as Google Classification,¹ Wiki Classification,² ACM Classification Hierarchy,³ and MSC Classification Hierarchy.⁴ Data-sources are provided by autonomous and independent providers, which means that these data-sources evolve independently from one another and with flexible structures [27]. This results in a change to the existing mapping methodologies, which makes these mappings unreliable with regard to the sharing of information. This is why there is a need for a system that supports mapping for evolving ontologies. Existing systems complete the mapping process by completely re-creating the mappings among the evolved ontologies, which is a very time consuming process.

Re-creation of mappings is required for mapped ontologies that are dynamic and subject to change. Existing systems take more time to re-create mappings as compared to the process of creating the initial mappings as these systems start the mapping process from scratch; however, the changes in the mapped schemas and regenerated mediation are not significant [27]. Consequently, a less time consuming scheme that can be used in the reconciliation of ontology mappings (mapping evolution) in dynamic and evolving ontologies is proposed in this research paper in order to support information exchange and reliable service interoperability. The hypothesis of the proposed approach is to only consider the changed resources in the mapping regeneration process that will not only reduce the time required for mapping regeneration but will also support updated and reliable mappings for information sharing and eliminate stale mappings while preserving the same level of accuracy. To achieve this, our approach uses the Change History Log (CHL) [38] (i.e., local, centralized, and distributed) to map reconciliation in less time than existing systems. The proposed technique drastically reduces the time required for the re-creation of mappings between dynamic ontologies. The CHL is used to store the changes occurring in dynamic ontologies, which are later used for mapping reconciliation. The use of the CHL in ontology matching/mapping helps in the reconciliation of mappings in dynamic and evolving web ontologies by overcoming the staleness problem associated with these mappings and reducing the time required to reconcile these mappings. During the reconciliation of ontology mapping, only the outdated mappings are updated, which saves both time and resources. We have tested the Falcon, Lily, FOAM, Prompt, AgreementMaker, and H-Match algorithms on 13 different data sets that are available online and then extended these algorithms with the proposed scheme by incorporating the use of CHL. Our proposed extensions have been tested on the same data sets and have shown a drastic reduction in the amount of time required for the reconciliation of these mappings. Detailed experimental results that support our claims are provided in this paper.

¹ http://www.google.com/Top/Reference/Libraries/Library_and_Information_Science/Technical_Services/Cataloguing/Classification/.

² http://en.wikipedia.org/wiki/Taxonomic_classification.

³ <http://www.acm.org/about/class/1998/>.

⁴ <http://www.math.niu.edu/~rusin/known-math/index/index.html>.

This paper is arranged as follows. Section 2 is a detailed discussion of the matching algorithms. Section 3 describes our proposed time efficient approach for reconciliation of mapping ontologies. Section 4 compares the experimental results of the proposed scheme with the results of existing systems. Finally, we conclude our discussion in Section 5 and talk about future applications of our proposed extensions.

2. Reconciliation of ontology mappings

Mappings are defined between two ontologies at a time, where one is called the source ontology and the other is called the target ontology. The proposed scheme for mapping reconciliation in dynamic and evolving ontologies is time efficient and eliminates staleness from the mappings by using ontology changes of the evolving ontologies to reconcile the mappings. It is based on the concept of the Change History Log (CHL) [38], which contains all the changes that have occurred in ontology throughout the evolution of the ontology. The change log is required as it details which ontology resource has changed and may have resulted in unreliable elements that exist in the mappings. The unreliable mappings problem is addressed in the proposed scheme in an efficient manner discussed in the following sections. The proposed scheme (for architecture, see Fig. 3) for the reconciliation of ontology mapping has two main components: (1) Change History Log to maintain all of the ontology changes in a semantic structure and (2) Reconciliation of Mappings in order to eliminate unreliable mappings from the existing mappings and to re-establish the mappings for dynamic ontologies. However, before discussing the proposed solution, it is necessary to highlight the nature and dynamics of ontology and ontology mappings during and after evolution, as described in the following subsection.

2.1. Ontology and ontology mapping dynamics

Organizational knowledgebase (in our case ontology) is flexible to accommodate rapid developments, new contributions, data, modification in schema, and also policies for sharing data within an organization and outside [27,28]. These modifications have direct impact on dependent data, applications, and established mappings [28,29] and make changes unreliable. The unreliable mappings are a result of the changed resources, which is why re-alignments are needed. So far the reconciliation of ontology mappings for evolving ontologies has received little attention. The authors in [29] have focused more on mapping evolution based on multiple versions of ontologies than changed resources in the mapped ontologies.

Changes in ontology are broadly categorized into three categories [25,29] that define the dynamics of ontology and ontology during evolution.

- (1) *Extension* represents all the change operations that extend/add to the definitions of resources in ontology as shown in the following axiom.

$$O_1 \leftarrow O_1 \sqcup \{O_1/.Change \equiv \exists X \{X \mid X \in \Delta, X.resources.extension\}\}$$

- (2) *Reduction* represents all the change operations that reduce/delete from the definitions of resources in ontology as shown in the following axiom.

$$O_1 \leftarrow O_1 \sqcup \{O_1/.Change \equiv \exists X \{X \mid X \in \Delta, X.resources.reduction\}\}$$

- (3) *Revision* represents a change operation to concept(s) that revise/update definitions of resources in ontology as shown in the following axiom.

$$O_1 \leftarrow O_1 \sqcup \{O_1/.Change \equiv \exists X \{X \mid X \in \Delta, X.resources.revise = 1 \text{ Change}\}\}$$

Resources will remain unchanged if they are not affected by any of the above change operations. These change operations, as discussed above, will consequently affect the corresponding mappings. To overcome this issue, established mappings also need to evolve/be reconciled for accommodating new changes. The mapping reconciliation process is relatively simple as compared to ontology dynamics and is based on the concept of differential tables [19] where two operations i.e., *addition* and *deletion* are required to represent any change and these are described in the following axioms.

Addition:

$$\text{New Mappings } (M_{\text{new}}) \leftarrow M_{\text{old}} + \text{addedMappings} \equiv \exists X \{X \mid X \text{ is mappings in } O_1 \text{ and } O_2\}$$

$$\text{Reconciled Mappings } (M') \leftarrow M_{\text{new}} - M_{\text{old}}$$

Deletion:

$$\text{New Mappings } (M_{\text{new}}) \leftarrow M_{\text{old}} + \text{deletedMappings} \equiv \exists X \{X \mid X \text{ is deleted mappings in } O_1 \text{ and } O_2\}$$

$$\text{Reconciled Mappings } (M') \leftarrow M_{\text{old}} - M_{\text{new}}$$

The operation for modification of mappings in the reconciliation process is achieved by executing the operations of *Deletion* and *Addition* in sequence. The mapping reconciliation operations highly depend on the dynamics of ontology evolution and ontology evolution dynamics mainly specify which of the mappings dynamics/operations will be activated

for the reconciliation procedure. It might seem obvious that the *Extension* operation of ontology evolution dynamics will activate the *Addition* operation of the mapping reconciliation process; however, it is not necessarily true in all cases. The *Extension* operation can also activate the *Deletion* operation if the *Extension* operation is introducing constraints in ontology which might restrict some resources from participating in mappings which were established before. The detailed process for ontology dynamics and the reconciliation process is presented in the following subsections.

2.2. Change History Log (CHL)

Ontology change management deals with the problem of deciding which modifications to perform in response to a certain need for change. This generally keeps the changing ontology consistent and up to date while all of the required changes are accurately tracked. Different changes have different effects on overall ontology, and most of these changes are discussed in [39]. The number of changes, ranging from concepts to properties, can affect the ontology. These changes need to be represented properly in order to correctly handle explicit and implicit change requirements. This is why we have proposed a scheme for ontology change representation, which is referred to as the Change History Ontology (CHO) [38] and is used to log the ontology change, reason for change, and the change agents. CHO reuses the constructs from existing ontologies [42], ontology design patterns [22], the unified schema developed for mappings in [55], and is strictly bounded by the principle of change in knowledge [1].

To satisfy the validity of a change, a change must have three basic properties i.e., Minimality, Success, and Validity [1,55] and their implementation in CHO is as follow. The *Principle of Minimal Change* enforces the idea that the modifications that are supposed to be applied with ontology should be minimal. The applied changes are kept at the atomic or minimal level in order to avoid any drastic change in ontology. To enforce the minimality of change, the concept of the transaction ACID (atomicity, consistency, isolation, and durability) property is applied [24]. The axiom given below enforces that at a particular time, there should be only one change that satisfies the atomicity property of a change.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, X.\text{resources.lock} = \text{exclusive}\}$$

To add in the atomicity of change, executing it in isolation is implemented using the following axiom.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, X = 1\}$$

After having all the resources' rights and executing them in isolation, the next constraint to be verified before reflecting the results of these changes in dependent ontology is consistency. The constraints are to verify the consistency of ontology and once the change is executed then the results will be completely reflected in the dependent ontology. The axioms for consistency of ontology after a change are given in the *Principle of Validity*, whereas the durability of change implementation in ontology is enforced using the following axiom.

$$\text{Ontology} \equiv \{\{\text{Ontology} - \text{Change}\} \sqcup \{\text{Ontology} + \text{Change}\}\} \sqcap \{\text{Ontology.consistent} = \text{true}\}$$

Based on the above discussion and propositions, the axiom given below is used to enforce the overall minimality of the change. It represents the notion (constraint) for keeping the change at a minimal level i.e., one change at a time.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, X = 1 \text{ targetChange}\}$$

The *Principle of Success* observes and satisfies the priorities of alternate changes. Mostly, there are alternate changes available for a given change request. So before a change is applied, the set of changes (alternate changes) are tested for their final result. A change with minimal effects and complete execution is selected for the final implementation. The axiom given below satisfies implementation of this principle.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, \Delta = \{C1, \dots, Cn\}\} \text{ where } C = \text{ChangeInstance}$$

$$\text{Change} = 1\text{targetChange} \sqcap \min.(\text{Change.Effects})$$

The *Principle of Validity* enforces that when a change is applied then the ontology must evolve to a new consistent state. Any change that cannot satisfy the consistency constraint is not applied to the ontology. The following axiom for the ACID property of consistency is formulated and applied.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, \text{Ontology.consistent}(\text{change}) = \text{true}\}$$

This principle of validity is enforced using the following axiom based on the consistency validation borrowed from the ACID property of transactions.

$$\text{Change} \equiv \exists X \{X \mid X \in \Delta, \Delta = \{C1, \dots, Cn\}\} \text{ where } C = \text{ChangeInstance}$$

$$\text{Change} = 1\text{targetChange} \sqcap \text{Ontology.consistent}(\text{Change}) = \text{true}$$

To enforce the change implementation principles on CHO, CHO is modeled to capture changes at the atomic level and all the changes must be applied in isolation. Moreover, additional constructs are introduced in CHO and some of the notable constructs are discussed in further detail below.

The core elements of CHO are the *OntologyChange* and *ChangeSet* classes. The *OntologyChange* class has a sub-class called *AtomicChange* that represents all of the class, property, and individual level changes at the atomic level, as expressed in Fig. 1. The notion of *ChangeSet* in CHO is introduced from the Change Set Vocabulary [61]. The rationale is that the individual changes are not performed in isolation and are usually part of a particular session. The use of change set(s) is common in versioning systems such as CVS and SVN. A change set contains information about the changes that are made during an ontology engineering session. *ChangeSet* bundles all of the changes of a specific time interval in a coherent manner, as shown in Fig. 1. The *ChangeSet* is responsible for managing all of the ontology changes and arranges them in a time indexed fashion. This time indexing also classifies the *ChangeSet* as both an Instant type and an Interval type. The Instant type *ChangeSet* has the ability to hold only one change that occurred at some instant in time, whereas the Interval type *ChangeSet* holds the changes that occurred within a defined time interval [38].

Ontology development methodologies [33,62] reuse the concepts and patterns from foundational ontologies [22]. These patterns are extremely useful in order to acquire, model, develop, and refine these types of ontologies. In regard to CHO development, the fundamental ontology design patterns are used. The *Participation Pattern* consists of a **participant-in** relation between the ontology resource and the change event and assumes a time index [22]. Time indexing is provided by the temporal location of the change within a defined time interval, whereas the respective location within the ontology space is provided by the participating objects (see Fig. 1). In previous approaches [39,42,50,54], ontology changes are stored sequentially without preserving their dependence or interlinking with other changes. CHO uses *ChangeSet* for grouping and the time indexing of changes in a session in order to preserve coherence of all of the ontology changes that have occurred. A *ChangeSet* is a setting that is used for atomic changes. One ontology resource participates in a change event at a particular time interval. Fig. 1 shows the diagrammatic depiction of this pattern. The listing of all of the ontology changes is maintained in the CHL, which is constructed in such a way that it maintains all of the ontology changes in conformance to the CHO.

Corresponding to the CRUD interfaces in the databases (excluding read), three categories are used in the CHO to represent operations or change types. The change types include Create (such as *ClassAddition*, *PropertyAddition*, and *IndividualAddition*), Update (such as *ClassRenaming*, *PropertyRenaming*, and *IndividualRenaming*), and Delete (such as *ClassDeletion*, *PropertyDeletion*, and *IndividualDeletion*). As stated previously, there are three categories that represent different components of the ontology that are subject to change. These categories are *ClassChange*, *PropertyChange*, and *IndividualChange* [21,38]. Based on these categories, we derive instances of class *OntologyChange*, which are represented with the symbol Δ , by using the following axioms:

$$R_{\Delta} \equiv \exists \text{ChangeTarget.}(\text{Class} \sqcup \text{Property} \sqcup \text{Individual} \sqcup \text{Ontology})$$

$$\Delta \equiv R_{\Delta} \sqcap \forall \text{changeType.}(\text{Create} \sqcup \text{Update} \sqcup \text{Delete}) \sqcap \exists \text{changeAgent.}(\text{Person} \sqcup \text{SoftwareAgent}) \sqcap = 1 \text{changeReason}$$

For instance, the following statement (see Fig. 2) represents the class addition scenario by adding the same class for the range addition scenario of a property. The statement also includes the corresponding *ChangeSet* instance information.

In Fig. 2, the *log* is prefix for CHL, *cho* for CHO, and *human* for *Human* (*nci_anatomy*) ontology. The above statement depicts instances of *ClassAddition* and *RangeAddition* classes, which are defined as a sub-class of *ClassChange* and *PropertyChange*, respectively, and are also elaborated in more detail below.

$$\text{ClassAddition} \sqsubseteq \text{ClassChange} \sqcap \exists \text{changeType.Create}$$

$$\text{RangeAddition} \sqsubseteq \text{PropertyChange} \sqcap \exists \text{changeType.Update}$$

With reference to relational databases, our methodology includes logging techniques that allow for the ability to persistently store these types of changes. This helps in performing the undo/redo, ontology recovery, query reformulation, temporal traceability of ontology changes, and reconciliation of ontology mappings functions as needed. The changes are

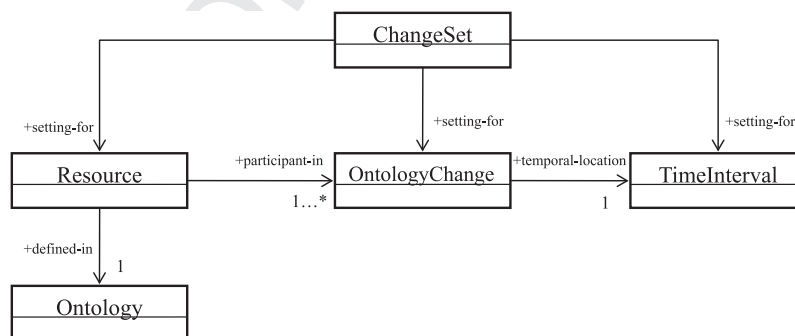


Fig. 1. Participation of ontology resource as an ontology change of a particular time interval in a time-indexed manner. *ChangeSet* is a setting for the changes that have occurred within a defined time interval [38].

log:Interval	
a	cho:ChangeSetType ;
cho:hasChangeSetTypeValue	"Interval" .
log:ChangePerson_Instance_1982	
a	cho:ChangePerson ;
cho:hasAuthorName	"Administrator" .
log:ChangeSet_Instance_2474557	
a	cho:ChangeSet ;
cho:hasChangeAgent	log:ChangePerson_Instance_1982 ;
cho:hasChangeSetType	log:Interval ;
cho:hasChangeBeginTime	00:00:46 ;
cho:hasChangeEndTime	00:03:21 ;
cho:hasChangeReason	"User Request" ;
cho:hasOntology	http://www.uclab.khu.ac.kr/human.owl .
log:ClassAddition_Instance_1224702057078	
a	cho:ClassAddition ;
cho:hasChangedTarget	human:NCL_C12801 ;
cho:hasTimeStamp	1224702057078 ;
cho:isSubClassOf	owl:Thing ;
cho:isPartOf	log:ChangeSet_Instance_2474557 .
log:RangeAddition_Instance_1224702057431	
a	cho:RangeAddition ;
cho:hasChangedTarget	human:UNDEFINED_part_of ;
cho:hasPropertyType	owl:ObjectProperty ;
cho:hasTargetRange	human: NCL_C12801 ;
cho:hasTimeStamp	1224702057431 ;
cho:isPartOf	log:ChangeSet_Instance_2474557 .

Fig. 2. Changes in the Human (nci_anatomy) ontology represented using the N3 notation and stored in the Change History Log (CHL).

preserved in a time-indexed manner using both the CHL and the schema provided by the CHO [38]. When a request is made for any of the above mentioned purposes, the CHL, which contains all of the changes that were made, is accessed to make the requested changes. Each entry in the log is an instance of either the *ChangeSet* or *OntologyChange* class from the CHO. The log also preserves the provenance information with regard to the change, such as who made the changes and when and also why these changes were made.

2.3. Mapping reconciliation procedure

As discussed above, there are different algorithms available to establish mappings between ontologies [8,32,44,49,63]. The existing systems are able to re-establish the mappings between dynamic ontologies after their evolution; however, these systems start their mapping re-establishment process from scratch, which is a time consuming operation. Our contribution is to use the change entries of the ontology (after the evolution) that are stored in the CHL [38] to guide the reconciliation of the mappings between ontologies, which not only helps to eliminate stale mappings, but also takes less time to reconcile mappings in dynamic and evolving ontologies. In this approach, we only concentrate on the resources that have changed between the evolved ontologies. The approach is most suitable for large ontologies that have hundreds or thousands of resources, such as when reconciling mappings (after change) among *Brinkman*, *GTT*, *GEMET*, *NALT*, *Google Classification*, *Wiki Classification*, *ACM Classification Hierarchy*, and *MSC Classification Hierarchy*. The larger the size of the ontology the better, as it becomes more time efficient than any of the algorithms discussed above. The detailed procedure is provided below.

Recreating Mappings: Consider the scenario given in Fig. 3, where two ontologies are mapped and exchange information based on the established mappings. If one or both of the ontologies change (evolve) to another state (see Fig. 3), then the existing mappings are of no use anymore, as they are not reliable and also become stale in this situation. This is why the mappings between these two ontologies need to evolve and why the evolving ontologies need to be up to date. In order to elaborate this concept further, we use two different cases.

Algorithm 1. Mapping reconciliation algorithm for ontology mapping using ontology changes stored in the Change History Log (CHL)

Input: Ontologies O_1 and O_2 for mapping reconciliation, Ontology change information (i.e., Δ_1 and Δ_2) from CHL of both ontologies, i.e., $\Delta_1 \in O_1$ and $\Delta_2 \in O_2$.

Output: Set of mappings for the changed resources is then updated in the original mappings file.

A resource matching threshold is defined as $\psi = 0.70$;

/* Check for change of resources in CHL of both mapped ontologies and read the changes in Δ */

if $\exists \Delta \sqcap \Delta.O_1.CHL.NewChange$ **then**

/* Read the changes in Δ_1 */

$\Delta_1 \leftarrow \{x | \langle CHL_{\Delta}, x \rangle \text{ Change} \}$

endif

if $\exists \Delta \sqcap \Delta.O_2.CHL.NewChange$ **then**

/* Read the changes in Δ_2 */

$\Delta_2 \leftarrow \{x | \langle CHL_{\Delta}, x \rangle \text{ Change} \}$

endif

/* Delete all the mappings from the original mapping file that are subject to change because of the change in the mapped resources. This method uses both Δ_1 and Δ_2 as optional parameters and is used if a change exists in the CHL and is retrieved in Δ */

Execute.delete(Mappings, [Δ_1], [Δ_2])

/* Start mapping reconciliation procedure by calculating the semantic affinity */

if $\exists \Delta_1.Change \sqcap \exists \Delta_2.Change$ **then**

/* Calculate semantic affinity using changed resources of both the ontologies */

R-Map [] $\leftarrow \text{SemanticAffinity}(C_1 \in O_1, \Delta_1, C_2 \in O_2, \Delta_2, \psi)$

else-if $\exists \Delta_1.Change \sqcup \exists \Delta_2.Change$ **then**

/* Calculate semantic affinity using changed resources of one changed ontology represented as Δ' */

R-Map [] $\leftarrow \text{SemanticAffinity}(C_1 \in O_1, C_2 \in O_2, \Delta', \psi)$

endif

/* Update the original mapping file with the reconciled mappings for the changed resources */

Execute.update(Mappings, R-Map[])

Case 1. If one of the ontologies evolves from one state to another, then its mapping with other ontologies will become unreliable as there will be a definite change in the resources that are mapped with the other ontologies. This is the reason why the mappings should be reconciled. We propose the use of the CHL entries to identify the changed resources in the evolved ontology instead of completely recreating the mapping process from scratch, which is a time consuming process. Only the changed resources in the mapping reconciliation process are used to map the changes with the other ontology, and we simply updated the previous mappings with the new ones while simultaneously removing the stale mapping entries. In this case, we only need to alter the method for calculating the Semantic Affinity (SA) by incorporating the change information from the CHL. The modified method, including parameters, is given below:

$$SA(C_1, C_2, \Delta_2, \psi) \begin{cases} C_1 \text{ Resource from Ontology } O_1 \\ C_2 \text{ Resource from changed Ontology } O_2 \\ \Delta_2 \text{ Change information from CHL of Ontology } O_2 \\ \Psi \text{ User defined threshold for resourcematch} \end{cases}$$

Case 2. Consider the second case where both of the ontologies have evolved from one consistent state to another as demonstrated in Fig. 3. This is also the worst case scenario in terms of execution time for mapping reconciliation. In this case, the mapping also needs to evolve in order to accommodate the mappings for the new resources that have changed and to eliminate the stale connections from the already established mappings. Again, we do not need to completely recreate the mappings between both ontologies, as required by the existing systems, which is a time and resource consuming process. We instead reconcile the mappings for the changed resources. As shown in Fig. 3, both ontologies, O_1 and O_2 , have evolved. In order to reconcile the mappings between the evolved ontologies and to remove the stale mappings in a time efficient manner, we use the CHL entries for both ontologies to identify all of the changed resources. Based on identified changes, we then reconcile the mappings for these changed resources, update the old mappings, and remove the unreliable (stale) mappings. This is not only a time efficient technique, but it also eliminates the stale mappings that need to be updated in order for reliable communication and for the exchange of information between systems and/or services.

The inputs for this module are also shown in Fig. 3 and consist of the evolved ontologies, O_1 and O_2 , and the CHL

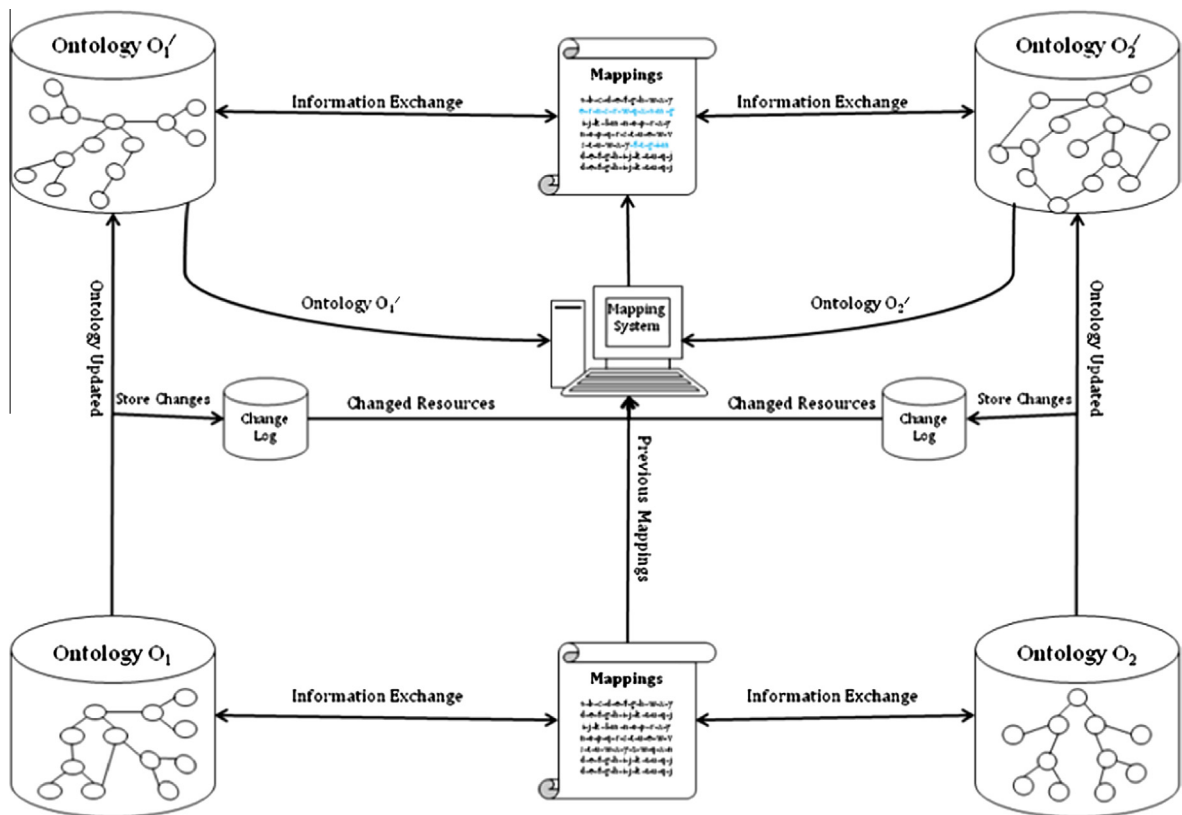


Fig. 3. The overall framework for reconciliation of mappings in dynamic and evolving web ontologies and the process of ontology change storage and the mapping reconciliation process.

entries for both ontologies, that is, Δ_1 and Δ_2 for ontology O_1 and ontology O_2 , respectively. The previous mappings between these two ontologies are also updated in the execution of the proposed algorithm (see Algorithm 1). The SA is calculated by incorporating the change information from the CHL, and thus the modified method including all of the parameters is as follows:

$$SA(C_1, \Delta_1, C_2, \Delta_2, \psi) = \begin{cases} C_1 & \text{Resource from ontology } O_1 \\ \Delta_1 & \text{Change information from CHL of Ontology } O_1 \\ C_2 & \text{Resource from Ontology } O_2 \\ \Delta_2 & \text{Change information from CHL of Ontology } O_2 \\ \Psi & \text{User defined threshold for resource match} \end{cases}$$

The variables Δ_1 and Δ_2 are the changes of both the ontologies that are contained in the CHL. In regard to calculating the SA, these changes are required and are extracted from the CHL using the SPARQL query given below. To determine the latest changes, the *ChangeSet* instances are extracted and sorted in descending order of timestamp as defined in the CHO, and the top most *ChangeSet* instance is then selected. Afterward, all of the changes corresponding to the selected *ChangeSet* instance are retrieved from the CHL.

```
Resource: ← SELECT ?changes ?timeStamp WHERE {?changes docLog:isPartOf changeSetInstance. ?changes docLog:has
TimeStamps ?timeStamp} ORDER BY DESC(?timeStamp)
Δx: ← SELECT ?changedTarget ?isSubClassOf WHERE {Resource docLog:hasChangedTarget ?changedTarget. Resource
docLog:isSubClassOf ?isSubClassOf}
```

After reconciliation, the stale parts of the mappings are removed. The mappings are then updated, as shown in Fig. 3 in the color blue. This process not only eliminates the staleness from the mappings, but it is also more time efficient (as it focuses on the changed resource), making it more suitable for systems and services that deal in information exchange.

3. Implementation and results

In this section, we present in detail the results that were achieved with the proposed extensions to those obtained using the existing mapping systems. The following experimental setup and data sets were used to conduct experiments and tests for to verify our proposed hypothesis that only considering *changed* resources in ontologies will reduce the time required for the mapping regeneration process, overcome the mapping staleness, and maintain the same level of accuracy.

3.1. Preliminaries

In this subsection, we explain the experimental setup established to compare in detail the results that were achieved with the proposed extensions to those obtained using the existing mapping systems,⁵ i.e., Falcon [32], H-Match [8], FOAM [18], Lily [63], AgreementMaker [12,13], and Prompt [49]. Comparing the results obtained from the proposed extensions against those of the existing systems verifies that the amount of time required for the reconciliation of the mappings using the proposed extensions is far less than when using the existing systems. The experiments in the present study were all conducted using a machine that had a 2.66 GH Quad Core processor and 4 GB of primary memory.

3.1.1. Data sets

The data sets used in these experiments were all available online (<http://oei.ontologymatching.org/>), and the ontologies were derived from *Mouse*, *Human*, *Brinkman*, *GTT*, *GEMET*, and *NALT*. Other data sets such as *Health and Food ontology*,⁶ *People + Pets ontology*,⁷ *ACM and Springer ontology*,⁸ *HL7 Classes ontology*,⁹ and *openEHR Classes ontology*¹⁰ have also been used to make detailed comparisons among similar systems.

For the experiments three different categories i.e., relevant, overlapping, and diverse ontologies were considered which can provide a more realistic results for all cases instead of only relevant ontologies. From this perspective, *Human Anatomy Ontology* (from the National Cancer Institute (NCI)) and *Mouse Anatomy Ontology* (from Mouse Genome Informatics (MGI)) included numerous relevant constructs from both domains. Similarly, *Health Ontology*, *Food Ontology*, and *People + Pets Ontology* also shared relevant constructs from their respective domains. *ACM Ontology* and *Springer Ontology* shared overlapping information from their respective publications application domains, whereas *HL7 Classes Ontology* and *openEHR Classes Ontology* were sharing overlapping information from the domain of Hospital Information Management System and health-care. On the other hand *Brinkman Ontology*, *GTT Ontology*, *GEMET Ontology*, and *NALT Ontology* were diverse in nature and had focus on biomedical investigation, glucose tests details, multilingual environment, and national agricultural thesaurus, respectively. The mapping results for all the above mentioned ontologies (except Health, Food, and People + Pets) were available online for verification; however, the focus of the proposed approach was to have immediate tests of the proposed extensions results against existing systems results, as explained in Sections 3.2–3.5 with details.

3.1.2. Experimental setup

The experiments were carried out for both cases as explained in Section 3.2 (i.e., Mapping Reconciliation Procedure). In all of the experiments, a constant similarity value of 0.70 was used as a matching threshold. The numbers of iterations for most of the systems were kept as the default value, with the exception of FOAM [18], which was set to seven iterations per execution; however, this value did not affect the results as these systems were not compared with one another with regard to accuracy. These experiments are by no means a comparison of existing systems, but are in fact the comparison of each individual system with our proposed extensions to that individual system. The experiments were conducted in two modes, which consisted of either complex or atomic level changes [37]. Complex change is a change that consists of several atomic level changes, for example a deletion of a super class will result in a complex change including the deletion of all of the sub-classes of that super class. Atomic change is a simple change, for example when renaming a resource. In these experiments, the changes that were made were mostly the introduction of new resources in the domain ontologies. Fig. 4 illustrates the limitations of the existing systems in that they do not focus on mapping evolution or its effects. The existing systems need more time to recreate the mappings with both complex and atomic changes. A total of 25 complex changes were introduced in each ontology version that was used in the experiments shown in Fig. 4a, whereas the atomic changes for each ontology version used in Fig. 4b are provided in Table 2.

3.2. Comparison using complex changes

In order to test the existing systems with our proposed extensions, a total of 25 random changes, all of which were complex, were introduced to the different ontologies used in our experiments. These changes caused the ontologies (listed in

⁵ The mapping systems selected for use and comparison in this paper are those which showed the best performance in the OAEI'05, OAEI'07, and OAEI'09.

⁶ <http://aims.fao.org/>.

⁷ <http://www.atl.lmco.com/projects/ontology/>.

⁸ <http://mapekus.fiit.stuba.sk/?page=ontologies>.

⁹ <http://web.science.mq.edu.au/~borgun/Software.html>.

¹⁰ <http://trajano.us.es/~isabel/EHR/>.

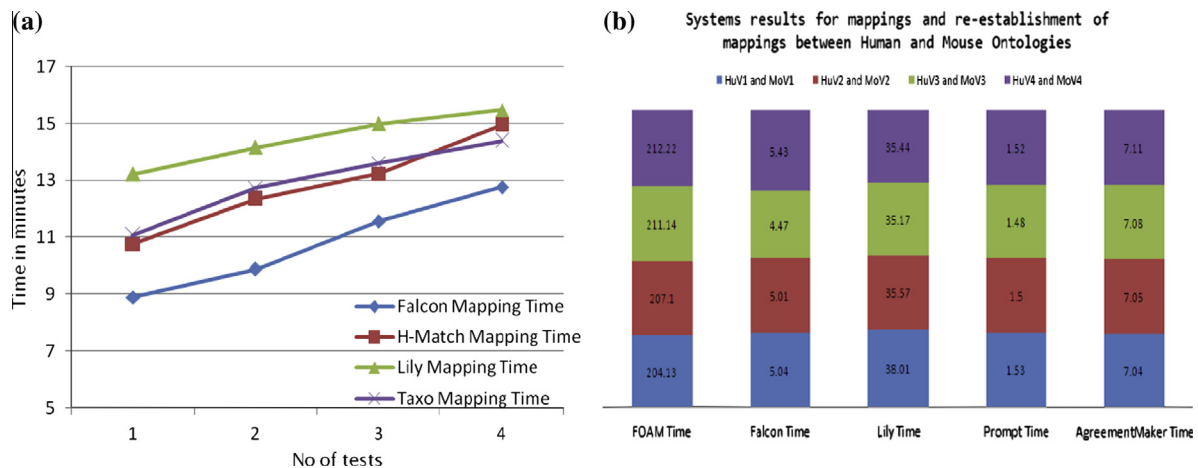


Fig. 4. (a) The mapping and re-establishment of the mapping results with respect to time for the Mouse and Human ontologies using Falcon [32], H-Match [8], Lily [63], and TaxoMap [49] with complex changes. (b) The mapping and recreation of the mapping results with respect to time for the Mouse and Human ontologies using FOAM [18], Falcon [32], Lily [63], AgreementMaker [13], and Prompt [49] with atomic changes.

Table 1

Time analysis of the original Falcon [32] and H-Match [8] with our extensions to Falcon and H-Match using the Change History Log.

Onto1	Onto2	Falcon mapping time (min)	H-match mapping time	Falcon re-mapping time (min)	H-match re-mapping time	Extended Falcon re-mapping time (min)	Extended H-match re-mapping time (min)
Mouse	Human	8.89	10.76 min	9.87	12.35 min	1.08	1.42
Brinkman	GTT	32.40	39.13 min	34.63	41.55 min	3.11	2.78
GEMET	NALT	51.33	1.12 h	53.71	1.17 h	5.36	7.31

Table 1) to evolve from one state to another. In these experiments, the ontologies are considered in full, including their structures and instances. As discussed above, these 25 complex changes were made to every version of the ontologies, which had an effect on both the structure and individuals. The existing algorithms (i.e., Falcon and H-Match) and proposed extensions to these algorithms were tested for both cases.

Case 1. In this scenario, only one of the ontologies evolved from one state to another, while the second ontology remained unchanged. Falcon and H-Match were first used to perform the initial mapping between the ontologies and were then used to re-establish the mappings using the changes in the single ontology. Afterward, the proposed extensions were applied for the changed ontologies to perform mapping reconciliation. As discussed earlier, the existing algorithms start from scratch and thus require more time than the proposed mapping process, as shown in **Table 1**. Our extension to the existing algorithms when using the CHL [38] only considers the changed resources and reconciles the mappings for only the changed resources. The proposed extensions (see **Table 1**) performed better than the existing systems. The amount of computational time (shown in **Table 1**), which is reflected in the columns titled *Extended Falcon Re-Mapping Time* (7th) and *Extended H-Match Re-Mapping Time* (8th) for mapping reconciliation, is better when compared to the columns *Falcon Re-Mapping Time* (5th) and *H-Match Re-Mapping Time* (6th). The results show that the extensions using the CHL drastically reduced the computational time for the reconciliation of mappings in dynamic ontologies.

Case 2. In this scenario, both ontologies evolved from one state to another. The Falcon and H-Match were first used for the initial mappings between the ontologies and then for the re-establishment of the mappings to reflect the changes in both ontologies. The algorithms were tested again to recreate the mappings and then to implement with the proposed extensions. The existing algorithms start from scratch and thus they take more time than the previous

Table 2

Time analysis of the original Falcon [32] and H-Match [8] with our extensions to Falcon and H-Match using the Change History Log.

Onto1	Onto2	Falcon mapping time (min)	H-match mapping time	Falcon re-mapping time (min)	H-match re-mapping time	Extended Falcon re-mapping time (min)	Extended H-match re-mapping time (min)
Mouse	Human	8.89	10.76 min	9.87	12.35 min	2.36	2.96
Brinkman	GTT	32.40	39.13 min	34.63	41.55 min	5.06	4.88
GEMET	NALT	51.33	1.12 h	53.71	1.17 h	9.48	12.39

Table 3

Ontology versions and the number of atomic changes applied to one version that transforms an ontology to another version. All of the ontologies are listed in the 1st row. Numeric values are the number of changes in the ontology when the current version is compared against the previous version.

Ontology versions	Human	Mouse	Health	Food	People + Pet	ACM ontology	Springer ontology
Version1	Original	Original	Original	Original	Original	Original	Original
Version2 = Version1 + No of Changes	283	166	169	122	120	109	176
Version3 = Version2 + No of Changes	112	201	153	161	172	133	114
Version4 = Version3 + No of Changes	123	198	145	114	109	141	106

test (shown in Table 2) when compared against our extensions so that only consider the changed resources and reconcile the mappings accordingly. Our proposed technique helped to save a large amount of computational time (as shown in Table 2) by comparing columns 7 and 8 for mapping reconciliation to columns 5 and 6. The results in both Tables 1 and 2 show that extensions using the CHL reduce the computational time to reconcile the mappings for both cases in dynamic web ontologies.

3.3. Comparison using atomic changes

This section describes the experimental results when the data sets that had changes at the atomic level were tested with the existing systems and our proposed extensions. For these experiments, only the structures of the ontologies were considered for the mapping procedures, and no individuals (instances) were used. Table 3 shows the different versions of the data sets and the number of atomic level changes between these versions. The existing systems, which were Falcon [32], FOAM [18], Lily [63], AgreementMaker [13], and Prompt [49], and our proposed extensions were tested on these data sets for the following two cases.

Case 1. In the first case, only one of the mapped ontologies evolved from one state to another, whereas the second ontology remained unchanged. The existing systems were used to initially map the ontologies and then to recreate the mappings after the changes in the ontology. Afterward, the proposed extensions were applied to reconcile the mapping between the changed ontologies. The existing systems and proposed extensions were all tested in detail using the data sets provided in Table 3, and the results for Case 1 are shown in Fig. 5. The execution times of these systems vary (see Fig. 5) due to the different matching schemes that were used in their implementation. Execution times shown in Fig. 5 are all in minutes and fractions of minutes.

Each graph in Fig. 5 shows the results of the existing systems and the proposed extensions on a particular data set with its different versions. Each graph in Fig. 5 consists of five pairs, which create a total of ten bars. Each alternative pair is the result of comparing the proposed system against the existing system. The 1st bar of each pair shows the execution time of the existing system on each version (differentiated using colors) of the ontology, whereas the 2nd bar of each pair shows the execution time for our proposed extensions for the different versions of the ontology. The percentage of the colored segments occupying each bar represents the percentage of time consumed for that particular execution of mapping and remapping procedure against the others, whereas the value inside represents the exact amount of time in minutes consumed. One very obvious pattern that is visible in each graph in Fig. 5 is that the execution time of the proposed extensions on the initial versions of the ontologies is always the same or slightly greater (max by 24 s) than those of the existing systems. If the ontologies are matched for the first time, as in this case, the proposed system carries out the complete mapping procedure in addition to looking for the changes in the CHL and existing mappings. The detailed experimental results shown in Fig. 5 validate that our proposed extensions drastically reduce the time required for reconciling ontology mappings for this case.

Case 2. As explained earlier, in this case, both ontologies evolved from one state to another. Case 2 is also the worst case for our proposed system as the mapping reconciliation procedure will look for changes in both ontologies and will also execute the mapping reconciliation procedure for both ontologies. The existing systems were first used to check for the initial mappings between the ontologies and then used to recreate the mapping process in order to account for the changes in both ontologies. For mapping reconciliation, the existing systems with the proposed extensions were then tested using the evolved ontologies. Both the existing systems and the proposed extensions were tested in detail using the data sets provided in Table 3 with all of their changes. The results of the detailed experiments for Case 2 are shown in Fig. 6. Execution times are also shown in Fig. 6 and are all represented by minutes and fractions of minutes.

Each graph of Fig. 6 represents the results of the existing systems and the proposed extensions on a particular data set with its different versions. Each graph of Fig. 6 consists of five pairs, which yield ten bars in total. Each alternative pair is the result of comparing the proposed system against the existing system. The 1st bar of each pair shows the execution time of the existing system on each version (differentiated using colors) of the ontology, whereas the 2nd bar of each pair shows the execution time of our proposed extensions for different versions of the ontology, which is the same as Case 1. The per-

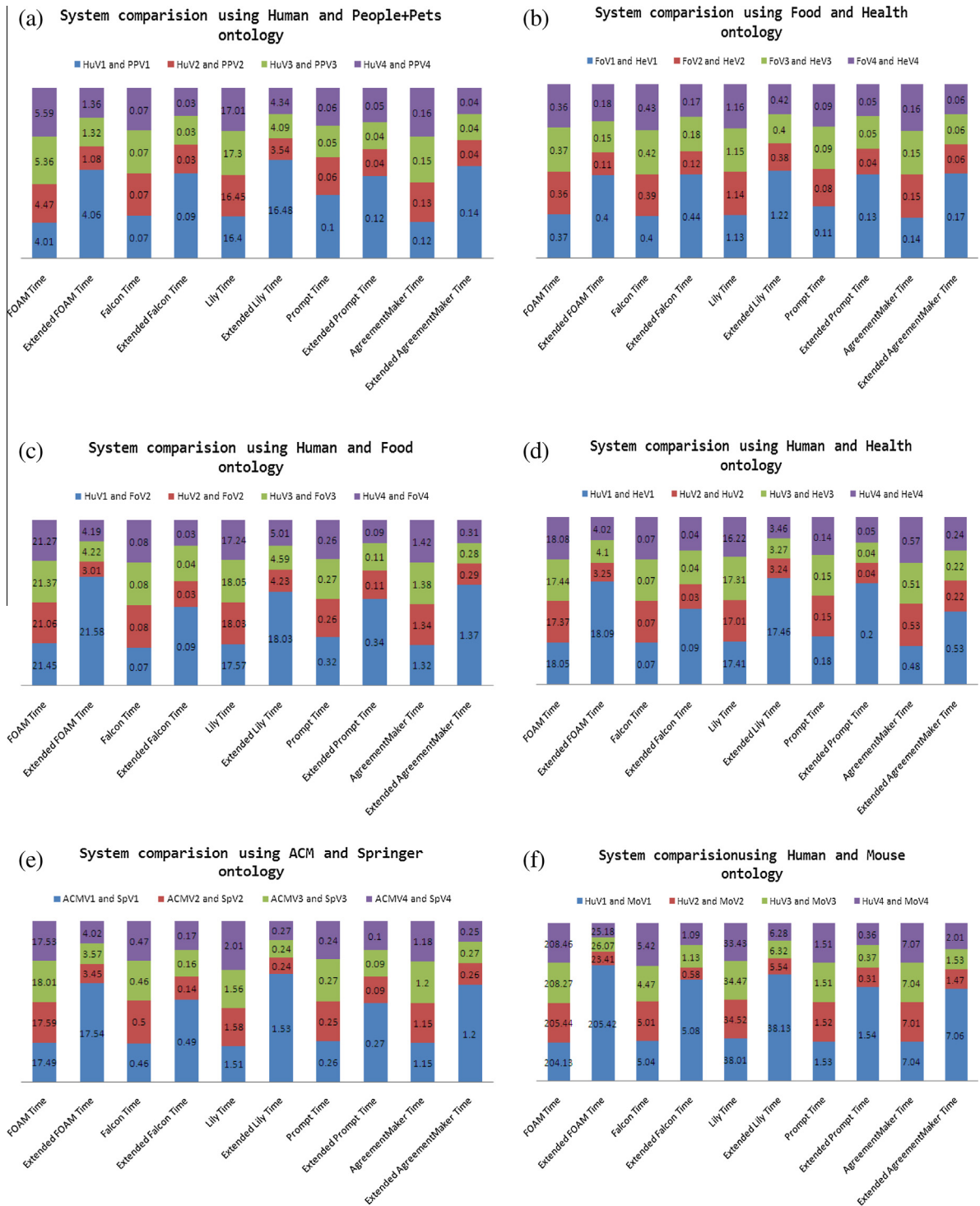


Fig. 5. Detailed comparison of the proposed extensions against Falcon [32], FOAM [18], Lily [63], AgreementMaker [13], and Prompt [49] using a combination of seven different data sets. This figure shows the results for Case 1. Each graph shows the results of the existing systems compared to the proposed extensions. Each graph consists of five pairs, yielding ten bars in total. Alternative pairs are the results of the comparison of the proposed system against the existing system. The 1st bar of each pair shows the execution time (y-axis shows the execution time) of the existing systems, while the 2nd bar shows the execution time for the proposed extensions. Each packet of every bar (stacked column) in the graphs of different colors shows the execution time consumed by the existing systems and the proposed extensions reconcile mappings between various versions of the ontology. In these graphs Hu = Human, Mo = Mouse, Fo = Food, He = Health, ACM = ACM, Sp = Springer, and PP = People + Pets are used as abbreviations for the ontology names where V represents the version number of the ontology. For example, HuV2 represents the second version of the Human ontology.

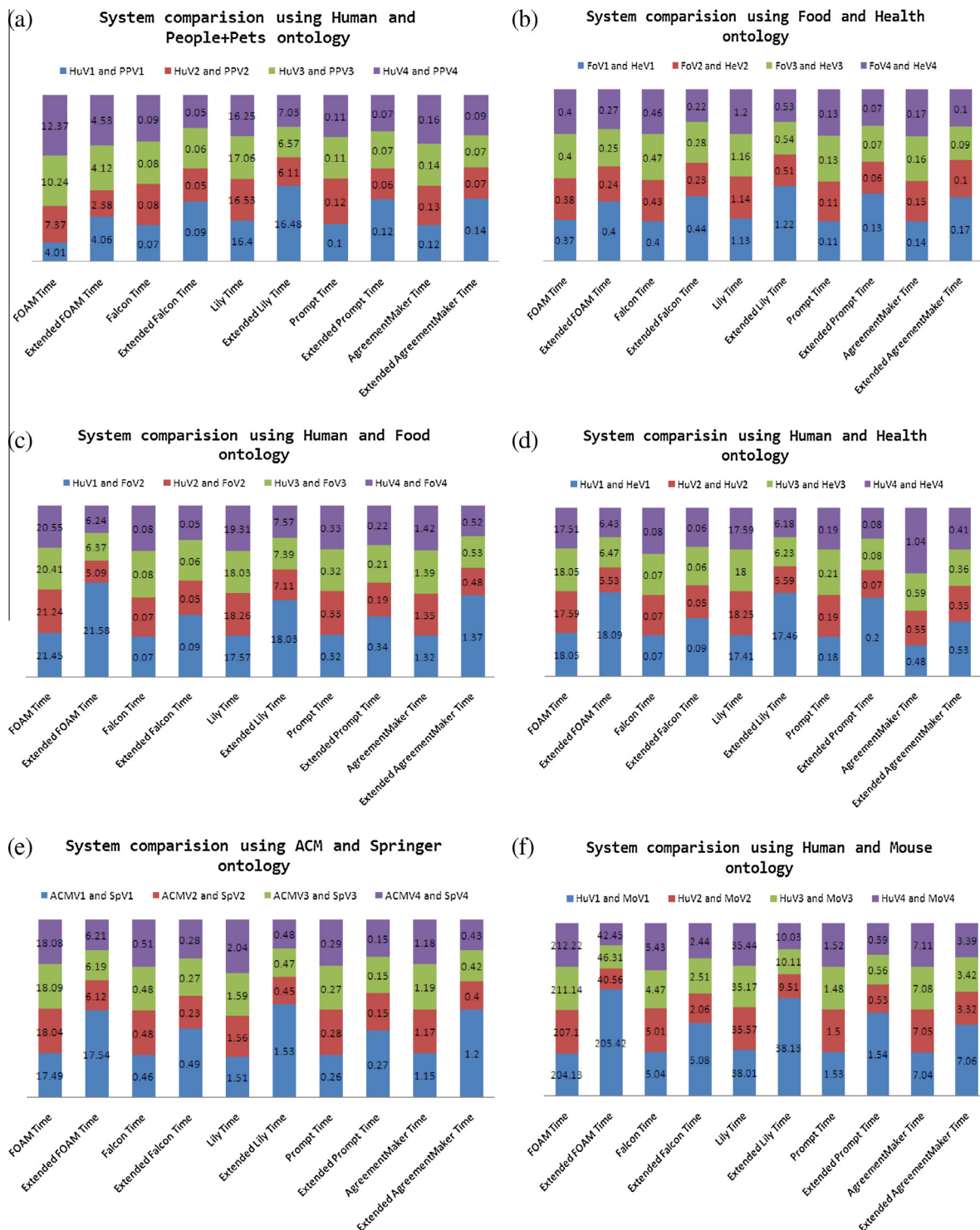


Fig. 6. Detailed comparison of the proposed extensions against Falcon [32], FOAM [18], Lily [63], AgreementMaker [13], and Prompt [49] using a combination of seven different data sets. This figure shows the results for Case 2. Each graph shows the results of the existing systems compared to the proposed extensions. Each graph consists of five pairs, yielding ten bars in total. Each alternative pair is the result of comparing the proposed system against the existing system. The 1st bar of each pair shows the execution time (y-axis shows the execution time) of the existing systems, while the 2nd bar shows the execution time for the proposed extensions. Each packet of every bar (stacked column) in the graphs with different colors shows the execution time consumed by the existing systems and the proposed extensions to reconcile mappings between various versions of the ontology. In these graphs Hu = Human, Mo = Mouse, Fo = Food, He = Health, ACM = ACM, Sp = Springer, and PP = People + Pets are used as abbreviations for the ontology names where V represents the version number of the ontology. For example, HuV2 represents the second version of the Human ontology.

centage of the colored segments occupying each bar represents the percentage of time consumed for that particular execution of the mapping and remapping procedure compared with the other one in the pair, whereas the numeric value inside represents the exact amount of time in minutes consumed in each experiment. The detailed experimental results shown in Fig. 6 validated our hypothesis. This facilitates the process of interoperability and information exchange between web services. Thus, the services are not suspended for longer durations due to evolving ontologies.

3.4. Effects of change type

The time it takes to reconcile the mappings between ontologies depends on the types of changes that are made. A single change may have a cascading effect on the existing resources or may result in several induced changes [21]. Our approach depends on the number of these changes. As the number of changes in an ontology increases, the mapping time will increase when using our approach. However, it is important to note that this mapping time is still less than those of the original algorithms. The cascading effects and induced changes are due to the changes that occur at higher levels of hierarchy and are less frequent once a domain ontology becomes more mature [21,27]. One such case is also visible in Fig. 7 (x-axis = no of tests, y-axis = minutes) in which the third bar can be compared between Fig. 7a and b. Fig. 7a shows the results for complex changes, whereas Fig. 7b shows the results for atomic changes.

The first bars in Fig. 7a and b are the original times for all of the algorithms to establish the mappings between the Human and Mouse ontologies, whereas the remaining bars represent the amount of time for mapping reconciliation when using our proposed extensions using the CHL. In Fig. 7a, a set of 25 random changes (complex) are introduced to each version of the ontology. In Fig. 7b, the changes (atomic) listed in Table 3 are introduced to each version of the ontology. In the 3rd bar combination of Fig. 7a and b, the cascading effects cause the reconciliation procedure to take longer than the other reconciliation tests with the proposed extensions. Nevertheless, even with the cascading effects and induced changes, our proposed approach requires less mapping computational time than the original algorithms.

3.5. Reconciled mapping accuracy

Although reconciled mapping accuracy is not the focus of our research, the accuracy of generated mappings is an important issue. The proposed extensions reduce the amount of time required for mapping reconciliation; however, it is also important to test the effects of the proposed method on the accuracy of reconciled mappings. In this section, the detailed results related to the reconciled mapping accuracy are provided (see Table 4) based on atomic level changes. The details of these atomic changes are given in Table 3. The results in Table 4 show the percentage (rounded percentages are given) of the overall mappings found after the reconciliation procedure. The mappings found by the original mapping systems are considered as the total possible mappings, whereas the mappings found with the proposed extensions are compared against the results of the original mapping systems. The details of the data sets and the mapping systems that have been used for the experiments are also provided in Table 4. During the logging process, every ontology change is logged in the CHL, and this also results in establishing/reconciling redundant mappings (that already exist in the original mappings),

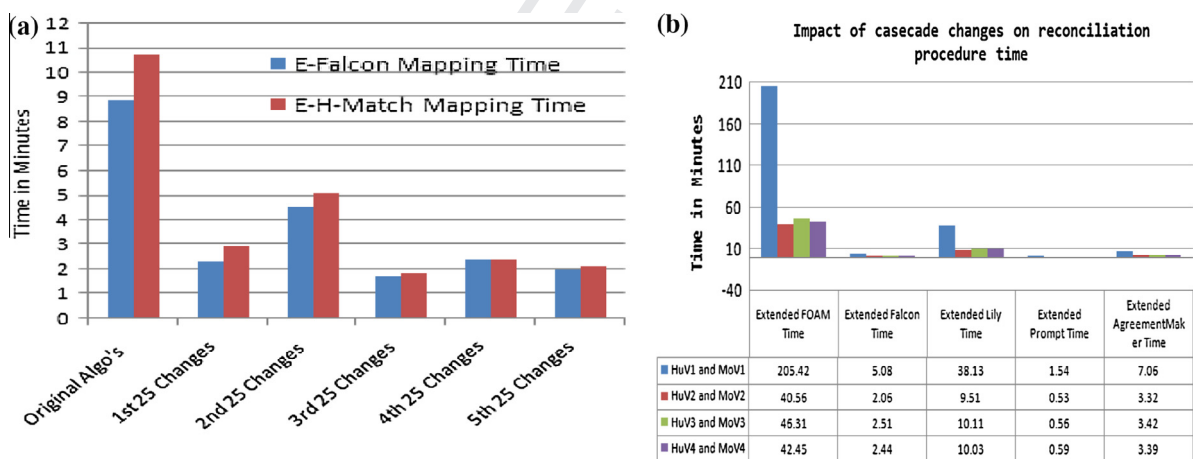


Fig. 7. (a) The mapping and recreation of the mapping results for the Mouse and Human ontologies with complex changes. The first bar combination is the result of the original Falcon [32] and H-Match [8], while the remaining bar combinations are the results of our proposed extension. The 3rd bar shows the time increase due to the cascading effects of changes. (b) The mapping and recreation of mapping results for the Mouse and Human ontologies with atomic changes. The first bar combination is the result of the original Falcon [32], FOAM [18], Lily [63], AgreementMaker [13], and Prompt [49], while the remaining bar combinations are the results of our proposed extension. The 3rd bar combination shows the time increase due to the cascading effects of changes. The same effects are also visible in the 3rd row of the tabular view of the results, which can be seen here by comparing it against the 2nd and 4th rows.

Table 4

The mapping accuracy results of the proposed extensions to the mapping systems against the original mapping systems. The results in this table are provided for Human, Mouse, Health, Food, ACM, and Springer ontologies, and FOAM [18], Falcon [32], AgreementMaker [13], and Lily [63] were used to represent the original mapping systems. The results do not show that the proposed extensions achieved 100% mapping accuracy; however, these results do show that the percentage of results (accuracy) achieved by the proposed extensions is higher compared to the original systems that require recreation of the complete mappings from scratch.

Ontology 1	Ontology 2	Changes	Ext-FOAM (%)	Ext-Falcon (%)	Ext-AgrMaker (%)	Ext-Lily (%)
Human V1	Mouse V1	Original	100	100	100	100
Human V2	Mouse V2	283 vs. 116	96.50	96.50	96.00	96.00
Human V3	Mouse V3	112 vs. 201	93.50	95.00	94.00	95.00
Human V4	Mouse V4	123 vs. 198	97.00	97.00	97.00	97.00
Human V1	Health V1	Original	100	100	100	100
Human V2	Health V2	283 vs. 169	97.00	98.50	98.00	99.00
Human V3	Health V3	112 vs. 153	96.50	96.00	97.00	97.00
Human V4	Health V4	123 vs. 145	99.00	100	100	100
Health V1	Food V1	Original	100	100	100	100
Health V2	Food V2	169 vs. 122	98.50	100	100	100
Health V3	Food V3	153 vs. 161	97.00	98.50	99.00	99.00
Health V4	Food V4	145 vs. 114	98.50	99.50	99.00	100
ACM V1	Springer V1	Original	100	100	100	100
ACM V	Springer V2	109 vs. 176	99.00	100	100	100
ACM V	Springer V3	133 vs. 114	98.50	99.50	99.00	99.50
ACM V	Springer V4	141 vs. 106	99.50	100	100	100

which are then removed from the final list of reconciled mappings. The formula used to calculate the percentage is simple and is provided below:

$$\text{Percentage accuracy of reconciled mappings} = (\text{No of reconciled mappings} / \text{No of original system mappings}) * 100$$

Most of the mapping systems that have been developed mainly focus on the accuracy of the mappings, which is more critical when the services or information systems deal with information related to the healthcare domain. To investigate the accuracy of the reconciled mappings, the healthcare domain ontologies, i.e., *HL7 Classes ontology* and *openEHR Classes ontology*, have been used to compare the two different versions for mapping and mapping reconciliation purposes. These ontologies have been tested using FOAM [18], Falcon [32], AgreementMaker [13], and Lily [63], and their results were compared with results from the proposed extensions to these systems (see Table 5). The changes used in these tests are also atomic changes, and the numbers of changes introduced in the different versions of the ontology are listed in Table 5. Similar to Tables 4, 5 also shows that there are fewer mappings after the reconciliation procedure than those that were identified by the original systems. In addition, these tools were found to have some deficiencies in finding exact matches with regard to the concepts of the ontology. For instance, the HL7 Classes ontology used SNOMED CT (O_1) as a base line that was then matched with another HL7 Classes ontology using HL7 RIM (O_2) as its base model, and the *Event* concept from O_2 is mapped with the *Event* concept of O_1 . However, both of them had different semantics. Similarly, the *Event* concept from O_2 has the same semantics as the *Clinical_Findings* in O_1 ; however, they are not matched when using the existing matching systems.

Table 5

Mapping accuracy results of the proposed extensions to the mapping systems against the original mapping systems using HL7 Classes ontology and openEHR Classes ontology. In these tests, only two versions of the said ontologies are used. FOAM [18], Falcon [32], AgreementMaker [13], and Lily [63] were used to represent the existing mapping process.

Mapping systems	Ontology 1	Ontology 2	Ontology changes	Mapping time (min)	Number of mappings found
FOAM	HL7 V1	openEHR V1	Original	18.23	16
Ext-FOAM	HL7 V1	openEHR V1	Original	18.57	16
FOAM	HL7 V2	openEHR V2	103 vs. 166	19.30	19
Ext-FOAM	HL7 V2	openEHR V2	103 vs. 166	4.03	17
Falcon	HL7 V1	openEHR V1	Original	0.58	18
Ext-Falcon	HL7 V1	openEHR V1	Original	1.01	18
Falcon	HL7 V2	openEHR V2	103 vs. 166	1.18	20
Ext-Falcon	HL7 V2	openEHR V2	103 vs. 166	0.26	19
AgrMaker	HL7 V1	openEHR V1	Original	1.17	18
Ext-AgrMaker	HL7 V1	openEHR V1	Original	1.24	18
AgrMaker	HL7 V2	openEHR V2	103 vs. 166	1.49	20
Ext-AgrMaker	HL7 V2	openEHR V2	103 vs. 166	0.41	18
Lily	HL7 V1	openEHR V1	Original	1.45	17
Ext-Lily	HL7 V1	openEHR V1	Original	1.49	17
Lily	HL7 V2	openEHR V2	103 vs. 166	2.09	19
Ext-Lily	HL7 V2	openEHR V2	103 vs. 166	0.49	18

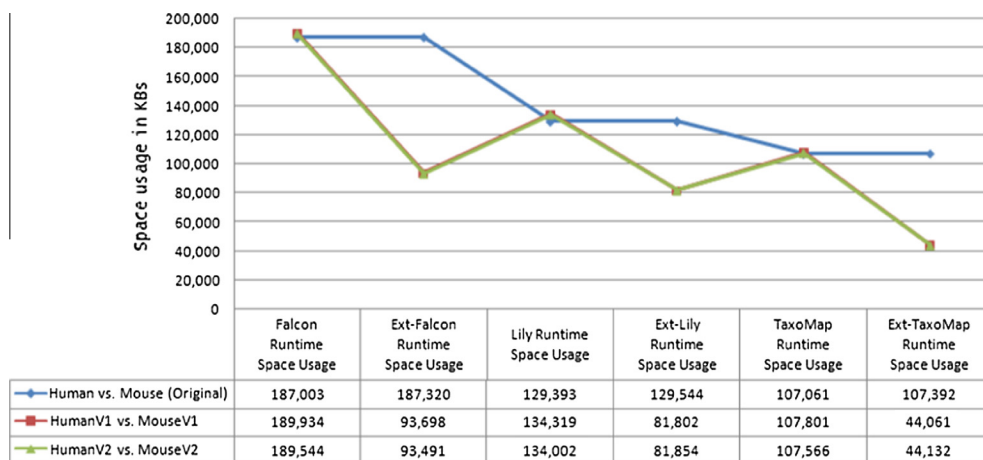


Fig. 8. Space consumption analysis of Falcon [52], H-Match [15], Lily [63], and TaxoMap [49] against the systems with the proposed extensions using CHL during the mapping reconciliation procedure. Memory size is represented in KBs, and the memory usage values are defined as the peak memory usage values recorded during the execution of the respective systems.

To overcome the decrease in accuracy of the reconciled mappings, two points have been addressed. The first point is that there has been an increase in the level of information with the changes. With every class change (except class deletion), extra information is provided during the reconciliation procedure, such as when dealing with a super class and its sub classes. Similarly, additional domain and range information is provided for every change in property (excluding property deletion). Improvements were found in the accuracy of reconciled mappings; however, this additional information also increased the time it took to complete the mapping reconciliation process. The second point to consider is the semantic conflicts that cannot be resolved without expert intervention as discussed above, such as the example regarding the HL7 Classes ontology. Currently, the focus is on identifying the missing mappings and the reasons for these missing mappings, which will also help to optimize the proposed system with regard to mapping accuracy.

3.6. Memory usage

In addition to time efficiency, the proposed extensions are also shown to be space efficient. After the mapping reconciliation procedure, when compared against the existing systems, at any particular instance of time, the ontology occurs from one side, and the changes occur from another side. The changes are far smaller in size than those of the original ontology and this is why we compared the proposed system's runtime memory usage with that of the existing systems. The results (see Fig. 8) show that the memory consumption of the proposed system is less than that of the existing systems. Moreover, as the proposed system needs less time than the existing systems to process, memory consumption also occurs for a shorter interval of time with the proposed system. Fig. 8 shows the results of the memory consumption of the proposed extensions to the existing systems against the traditional approach of the existing systems. The original and changed versions of the Human and Mouse ontologies shown in Table 3 were used to demonstrate the effect of the proposed system on memory usage.

4. Related work

The progressive emergence of information and communication technology has enabled the continuous flow and storage of information in diverse (i.e., heterogeneous) nature. This information is sometimes overlapping or to some extent connected and needs to be integrated. The problem is how to overcome the syntactic and semantic heterogeneity of the information and merge/integrate it [26,58]. The scheme of matching and mapping is used to overcome this issue of heterogeneity. To formalize the process ontology based matching, schemes are proposed that provide basic vocabulary to resolve the syntactic and semantic heterogeneities [20,21,26,58]. Ontology is currently being used by convergent technologies, such as Context-aware Search Engines [36], Software Agents [9,35], Data Integration [23,64], Semantic Grid [53], Cloud Computing [7,60], and Semantic Web Services [45,51]. Many research groups are working on ontology matching/mapping and have developed different systems that facilitate interoperability between collaborative convergent technologies which demonstrate that research in this area is active. This section lists the existing systems in the field of ontology and schema matching and the systems extended in the proposed approach to achieve the overall objectives. From this prospective, Falcon [32] is an overall infrastructure for Semantic Web ontology learning, matching, mapping, and aligning and is extensively used by the semantic web community for applications, such as providing well designed technologies for finding, aligning, and learning ontologies and ultimately for knowledge discovery. Falcon-AO is one of the prominent components of Falcon and is an automatic ontology matching component that enables interoperability among Semantic Web applications by using related ontol-

ogies. Falcon-AO is one of the most practical and popular tools for web ontology matching and mappings that are expressed in RDF(S) and/or OWL. This tool consists of five main components: (1) *Repository*, (2) *Model Pool*, (3) *Alignment Set*, (4) *Matcher Library*, and (5) *Central Controller*. All of these components collectively perform the tasks that are submitted to the Falcon-AO. Falcon-AO considers ontology at both the element level and the structure level while matching two ontologies. Similarly, the COMA system also generates different segments for the two schemas and then finds similarities among different generated segments of the ontologies [16]. However, automatically generating appropriate segments of schemas is another research challenge. COMA [16] initializes the matching process with partial input, which is manually constructed. On the other hand [30] uses statistical values of different schemas to find the originating or basic building model for the schemas and also finds similarities amongst compared schemas.

AgreementMaker [12,13] provides a wide range of matching methods that address the different levels of granularity (i.e., concept and structure level) of ontology. This algorithm also facilitates user intervention for semantic conflict resolution and is a flexible system in that it can facilitate integration and performance tuning of different matching methods. Prompt [49] is an ontology merging, difference, and alignment tool with a sophisticated scheme of matching terms and is an open source system built using Java. It handles ontologies expressed in OWL and RDFS and produces the alignments/mappings between two ontology inputs. Prompt is already being developed as the Protégé plug-in. However, it is important to note that additional modifications are required on the user side in order to use the source in a user's own application.

H-Match [8] is an ontology matching system that takes ontologies as inputs and produces results as associations among the related resources between different ontologies. The associations are then used to create the mappings between these ontologies. H-Match is capable of dynamically configuring its adaptation to the semantic complexities of the ontologies that are to be matched, where the number and types of ontology features are not known in advance. H-Match enforces these dynamic adoption capabilities with the help of syntactic and semantic techniques for ontology matching, and it also incorporates a set of four matching models, i.e., *surface*, *shallow*, *deep*, and *intensive*. H-Match is used for knowledge discovery in the *Helios* framework, which is a peer-based system [9]. Similarly, in [64] a semantic approach is followed to find element and attribute level similarities in XML sources using an Object-Relationship-Attribute model to integrate the sources. The focus is to uncover the implicit semantics for the elements and attributes which can participate in the process of matching and integration.

Lily [63] is a combination of textual and structural techniques that is used to determine the alignments between ontologies. Lily builds semantic descriptions for each entity of the ontology and then uses lexical similarity and similarity flooding of the ontology structure. Its uniqueness lies in the use of web search engines to overcome semantic heterogeneity. Post processing is conducted in order to remove inconsistencies and to increase the accuracy of the results. The framework of FOAM [18] is based on heuristically calculated similarities of each resource that is available in the ontology. Its focus is on the efficiency of the alignment that is generated that also distinguishes it from the other systems. Like most of the other systems, FOAM also uses the structure of the ontology to determine the relatedness among the entities of the participating ontologies.

MAFRA [44] is an Ontology Mapping FRamework that was mainly developed to distribute ontologies in the Semantic Web. MAFRA provides a conceptual framework with a generic view of the complete distributed mapping process among the distributed ontologies. Due to the decentralized nature of the Semantic Web, there is a significant amount of information redundancy, and consistent evolution of the ontologies occurs in order to accommodate the domain knowledge. Thus, the changing nature of ontology also needs to be re-established with regard to the mapping among other ontologies, and the developers of MAFRA are considering this issue for future versions.

The mapping systems that are discussed above are those known for their outstanding performances according to the *Ontology Alignment Evaluation Initiative*¹¹. Among the discussed systems, AgreementMaker [12,13] and Lily [63] are the most efficient and widely used tools for ontology matching and mapping with relatively better accuracy. Also, when the alignment is constructed from scratch, the accuracy is generally better than that of the other existing algorithms [8,44,49]. However, like every other system, both AgreementMaker and Lily take a considerable amount of time to establish alignments and have no support for the mapping reconciliation process (i.e., unreliable mappings).

All of the above systems reinitiate the process of mapping between ontologies after updating. This consumes a lot of time as the changes are usually very few and simple in type [21,27]. Systems, such as those in [2,17,44,55,63,65], are designed to support mapping evolution; however, some have a different focus, and a few are not mature enough in their approach. The systems discussed in [17,65] mainly focus on the schema-based mapping evolution that supports Local as View and Global as View approaches [41], which support query reformulation [55] in data integration applications. The system proposed in [65] focuses on mapping evolution that is based on the incremental adoption of changed mappings. The incremental adoption technique makes it hard to cope with drastic schema evolution situations. The system discussed in [17] is based on the composition and inversion technique. This technique restricts the schema evolution to a set of defined states that are based on mapping evolution options, which do not occur in real world situations [21]. The approach discussed in [55] formalized a unified schema for managing the mappings generated between schemas after the matching process. However, our proposed approach is different from those used [17,30,55,65] as the schema and the ontologies are fundamentally different [2,48]. In [2], the authors proposed a mapping evolution algorithm for mappings between a schema and the schema's annotations. The focus of the algorithm is to maintain consistency with regard to the mapping between the schemas and their corresponding

¹¹ <http://ontologymatching.org/evaluation.html>.

annotations. Both of the systems discussed in [17,65,2] are different from our proposed system, as [17,65] focus on schema level mapping evolution, and [2] focuses on the mapping evolution between schema and annotations (Meta data) for the schema. MAFRA [44] and Lily [63] are the two mapping systems that, in addition to mapping the generation between the two ontology versions, also focus on the evolution of the mappings when at least one of the mapped ontologies evolves from one state to another. However, neither MAFRA [44] nor Lily [63] has a concrete methodology in place that can support the mapping of evolving ontologies. Thus, for the testing and discussion of the proposed system, the authors made extensions to the existing systems in order to support the mapping reconciliation procedure instead of redeveloping a completely new mapping system.

5. Conclusions and future directions

Information exchange and interoperability are key research issues for many research groups and financial organizations. Mapping between two information sources (i.e., ontologies) is the key for information sharing and achieving interoperability. Systems exist that generate mappings between ontologies to support the exchange of information and interoperability; however, these are time consuming when we consider dynamic ontologies from the participating organizations that evolve over the passage of time. The dynamic nature of ontologies makes the existing mappings unreliable and stale, thus these mappings need to be reconciled to keep the services functioning optimally in order to exchange information. The proposed scheme uses the concept of CHL to log all the changes in an evolving ontology. These logged changes are later used with our proposed extensions to the existing mapping systems during the reconciliation of the mapping process. The process of mapping reconciliation is executed and tested on both cases i.e., evolution of only one ontology and evolution of both ontologies, and the proposed extensions to the existing systems have shown good results. The proposed scheme has drastically reduced the amount of time required to reconcile ontology mappings among dynamic ontologies when compared to the existing systems that recreate the process from scratch. The claims for our proposed extensions are validated by the results from seven (7) different mapping systems and thirteen (13) different data sets. Our future work will focus on the variable mapping accuracy of the proposed technique in order to find an optimized technique that is not only time efficient but also provides the same level of mapping accuracy by incorporating meta information with the logged changes.

6. Uncited reference

[34].

Acknowledgements

This work was supported by the Industrial Core Technology Development Program (10049079, Development of Mining core technology exploiting personal big data) funded by the Ministry of Trade, Industry and Energy (MOTIE, Korea). Also, this work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2014R1A2A2A01003914).

References

- [1] C.E. Alchourrn, P. Grdenfors, D. Makinson, On the logic of theory change: partial meet contraction and revision functions, *J. Symbolic Logic* 50 (1) (1985).
- [2] Y. An, T. Topaloglou, Maintaining Semantic Mappings between Database Schemas and Ontologies, *Semantic Web, Ontologies and Databases*, Lecture Notes in Computer Science, 2008, pp. 138–152.
- [3] P.B. Bhat, C.S. Raghavendra, V.K. Prasanna, Efficient collective communication in distributed heterogeneous systems, in: *Proc. 19th IEEE International Conference on Distributed Computing Systems (ICDCS)*, Austin, TX, 1999, pp. 15–24.
- [4] V. Bicer, O. Kilic, A. Dogac, G.B. Laleci, Archetype-based semantic interoperability of web service messages in the health care domain, *Int. J. Semantic Web Inf. Syst. (IJSWIS)* 1 (4) (2005) 1–23.
- [5] M. Bilal, R. Batool, W.A. Khan, E.N. Huh, S.Y. Lee, SPHeRe: a performance initiative towards ontology matching by implementing parallelism over cloud platform, *J. Supercomput.* 68 (1) (2014) 274–301.
- [6] E. Brynjolfsson, H. Mendelson, Information systems and the organization of modern enterprise, *J. organ. Comput.* 3 (3) (1993) 245–255.
- [7] R. Buyya, S. Yeo, S. Venugopal, J. Broberg, I. Brandic, Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility, *Future Gener. Comput. Syst.* (2009).
- [8] S. Castano, A. Ferrara, S. Montanelli, Matching ontologies in open networked systems, *Tech. Appl. J. Data Semantics (JoDS)* V (2006) 25–63.
- [9] H. Chen, T. Finin, A. Joshi, An ontology for context-aware pervasive computing environments, *Special Issue Ontol. Distrib. Syst. Knowl. Eng. Rev.* 18 (3) (2004) 197–207.
- [10] F. Chen, Z. Zhang, L. Li, J. Kang, H. Yang, Service identification via ontology mapping, in: *Proceedings of the 33rd Annual IEEE International Computer Software and Applications Conference, USA, 2009*, pp. 486–491.
- [11] N. Choi, I. Song, H. Han, A survey on ontology mapping, *SIGMOD Rec.* 35 (3) (2006) 34–41.
- [12] I.F. Cruz, F.P. Antonelli, C. Stroe, AgreementMaker: efficient matching for large real-world schemas and ontologies, *PVLDB* 2 (2) (2009) 1586–1589.
- [13] I.F. Cruz, W. Sunna, N. Makar, S. Bathala, A visual tool for ontology alignment to enable geospatial interoperability, *J. Visual Lang. Comput.* 18 (3) (2007) 230–254.
- [14] S. Dietze, H. Benn, J. Domingue, A. Conconi, F. Cattaneo, Two-fold service matchmaking – applying ontology mapping for semantic web service discovery, in: *Proceedings of Asian Semantic Web Conference, China, 2009*, pp. 246–260.
- [15] S. Dietze, A. Gugliotta, J. Domingue, H.Q. Yu, M. Mrissa, An automated approach to semantic web services mediation, *J. Serv. Orient. Comput. Appl.* 4 (4) (2010) 261–2754.

- [16] H. Do, E. Rahm, COMA – a system for flexible combination of schema matching approaches, in: Proceedings of the International Conference on Very Large Databases (VLDB 2002), Hong Kong, China, 2002, pp. 610–621.
- [17] R. Fagin, P. Kolaitis, L. Popa, G. Tan, Schema mapping evolution through composition and inversion, in: Schema Matching and Mapping, Data-Centric Systems and Applications, 2011, pp. 191–222.
- [18] M. Ehrig, Y. Sure, Foam – framework for ontology alignment and mapping; results of the ontology alignment initiative. in: Proc. of the Workshop on Integrating Ontologies, 2005.
- [19] R. Elmasri, S.B. Navathe, Fundamentals of Database Systems, fourth ed., Addison Wesley, 2003.
- [20] A. Ferrara, A. Nikolov, J. Noessner, F. Scharffe, Evaluation of instance matching tools: the experience of OAEI, Web Semantics: Sci. Serv. Agents World Wide Web 21 (2013) 49–60.
- [21] G. Flouris, D. Manakanatas, H. Kondylakis, D. Plexousakis, G. Antoniou, Ontology change: classification and survey, Knowl. Eng. Rev. (KER) 23 (2) (2008) 117–152.
- [22] A. Gangemi, Ontology design patterns for semantic web content, 4th Intl Semantic Web Conf (ISWC), vol. 3729, Springer, Ireland, 2005.
- [23] P.R. Gianolli, J. Mylopoulos, A semantic approach to XML-based data integration, in: Proceedings of the International Conference on Conceptual Modelling (ER'01), Yokohama, Japan, 2001, pp. 117–132.
- [24] J. Gray, A. Reuter, Transaction Processing: Concepts and Techniques, Morgan Kaufmann Publishers, Inc., San Francisco, USA, 1993.
- [25] A. Groß, M. Hartung, A. Thor, E. Rahm, How do ontology mappings change in the life sciences?, in: Selected Poster @ Intl. Conference on Data Integration in the Life Sciences (DILS), June 2012.
- [26] M. Gulić, B. Vrdoljak, CroMatcher-results for OAEI 2013, in: Proceedings of the 8th International Workshop on Ontology Matching, October 2013, pp. 117–222.
- [27] A.Y. Halevy, Z.G. Ives, M. Jayant, P. Mork, D. Suciu, I. Tatarinov, The Piazza peer data management system, IEEE Trans. Knowl. Data Eng. 16 (2004) 787–798.
- [28] M. Hartung, T. Kirsten, E. Rahm, Analyzing the evolution of life science ontologies and mappings, in: Proc. of 5th Int. Workshop on Data Integration in the Life Sciences (DILS), LNCS, vol. 5109, Springer, 2008.
- [29] M. Hartung, A. Groß, E. Rahm, COnto-Diff: generation of complex evolution mappings for life science ontologies, J. Biomed. Inform. 46 (1) (2013) 15–32.
- [30] B. He, K.C.C. Chang, Statistical schema matching across web query interfaces, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD 2003), San Diego, California, United States, 2003, pp. 217–228.
- [31] S.M. Huff, R.A. Rocha, B.E. Bray, H.R. Warner, P.J. Haug, An event model of medical information representation, JAMIA 2 (1995) 116–134.
- [32] W. Hu, Y. Qu, Falcon-AO: a practical ontology matching system, J. Web Semantics 6 (3) (2008) 237–239.
- [33] D. Jones, T. Bench-Capon, P. Visser, Methodologies for ontology development, in: J. Cuenca (Ed.), IFIP XV IT & KNOWS, Hungary, 1998.
- [34] J. Kang, J.F. Naughton, On schema matching with opaque column names and data values, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD 2003), San Diego, California, United States, 2003, pp. 205–216.
- [35] W.A. Khan, M. Bilal, A.M. Khattak, M. Hussain, M. Afzal, S.Y. Lee, E.S. Kim, Object oriented and ontology alignment patterns based expressive mediation bridge ontology (MBO), J. Inf. Sci. (2014) (in press).
- [36] A.M. Khattak, J. Mustafa, N. Ahmed, K. Latif, S. Khan, Intelligent search in digital documents, in: IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Australia, vol. 1, 2008, pp. 558–561.
- [37] A.M. Khattak, K. Latif, S. Lee, Y.K. Lee, Ontology evolution: a survey and future challenges, in: The 2nd International Conference on u- and e-Service, Science and Technology (UNESST 09), Korea, 2009.
- [38] A.M. Khattak, K. Latif, S.Y. Lee, Change management in evolving web ontologies, Knowl.-Based Syst. 37 (2013) 1–18.
- [39] M. Klein, Change Management for Distributed Ontologies, PhD Thesis, Vrije University, Netherlands, 2004.
- [40] T.B. Lee, J. Hendler, O. Lassila, The semantic web: a new form of web content that is meaningful to computers will unleash a new revolution of possibilities, Sci. Am. 5 (1) (2001).
- [41] M. Lenzerini, Data integration: a theoretical perspective, in: Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, New York, USA, 2002, pp. 233–246.
- [42] Y.D. Liang, Enabling Active Ontology Change Management Within Semantic Web-Based Applications. Mini PhD Thesis, University of Southampton, 2006.
- [43] J. Madhavan, P. Bernstein, E. Rahm, Generic schema matching with cupid, in: 27th International Conference on Very Large Data Bases, 2001.
- [44] A. Maedche, B. Motik, N. Silva, R. Volz, MAFRA – a Mapping FRamework for distributed ontologies, in: Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, London, 2002, pp. 235–250.
- [45] M. Martin, S. Paolucci, M. McIlraith, D. Burstein, D. McDermott, B. McGuinness, T. Parsia, M. Payne, M. Sabou, N. Solanki, Srinivasan, K. Sycara, Bringing semantics to web services: the OWL-S approach, in: Presented at the First Int. Workshop on Semantic Web Services and Web Process Composition (SWSWPC2004), San Diego, CA, 2004.
- [46] I. Mathur, N. Joshi, H. Darbari, A. Kumar, Shiva: A Framework for Graph Based Ontology Matching, 2014. arXiv preprint arXiv: 1403.7465.
- [47] M. Nagarajan, K. Verma, A.P. Sheth, J. Miller, J. Lathem, Semantic interoperability of web services: challenges and experiences, in: Proc. 4th IEEE International Conference on Web Services, IEEE CS Press, 2006, pp. 373–382.
- [48] N. Noy, M. Klein, Ontology evolution: not the same as schema evolution, J. Knowl. Inf. Syst. 6 (4) (2004) 428–440.
- [49] N. Noy, M. Musen, The PROMPT suite: interactive tools for ontology merging and mapping, Int. J. Hum. Comput. Stud. 59 (6) (2003) 983–1024.
- [50] D. Oberle, R. Volz, B. Motik, S. Staab, An extensible ontology software environment, in: Handbook on Ontologies, Series of International Handbooks on Information Systems, Springer, 2004, pp. 311–333.
- [51] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara, Semantic matching of web services capabilities, in: Proceedings of International Semantic Web Conference (ISWC-2002), 2002.
- [52] M. Paolucci, N. Srinivasan, K. Sycara, Expressing WSMO mediators in OWL-S, in: Proceedings of Semantic Web Services (ISWC), 2004.
- [53] D.D. Roure, N.R. Jennings, N.R. Shadbolt, The semantic grid: past, present and future, in: Proceedings of the IEEE 93, vol. 3, 2005, pp. 669–681.
- [54] D. Rogozan, G. Paquette, Managing ontology changes on the semantic web, in: IEEE/WIC/ACM International Conference on Web Intelligence, 2005.
- [55] R.D. Santos Mello, S. Castano, C.A. Heuser, A method for the unification of XML schemata, Inf. Softw. Technol. 44 (4) (2002) 241–249.
- [56] A.P. Sheth, J.A. Larson, Federated database systems for managing distributed, heterogeneous, and autonomous databases, ACM Comput. Surv. (1990).
- [57] P. Shvaiko, J. Euzenat, Ten challenges for ontology matching, in: Proceedings of the 7th International Conference on Ontologies, Databases, and Applications of Semantics (ODBASE), 2008.
- [58] P. Shvaiko, J. Euzenat, Ontology matching: state of the art and future challenges, IEEE Trans. Knowl. Data Eng. 25 (1) (2013) 158–176.
- [59] P.S. Tan, A.E.S. Goh, S.S.G. Lee, An ontology to support context-aware B2B services, in: IEEE International Conference on Services Computing (SCC), USA, 2010, pp. 586–593.
- [60] F. Tao, Y. Cheng, L.D. Xu, L. Zhang, B.H. Li, CCloudT-CMfg: Cloud Computing and Internet of Things Based Cloud Manufacturing Service System, 1-1, 2014.
- [61] S. Tunnice, I. Davis, Changelog, 2005. <<http://vocab.org/changeset/schema.html>>.
- [62] M. Uschold, Building ontologies: towards a unified methodology, in: 16th Annual Conference of the British Computer Society Specialist Group on Expert Systems, Cambridge, UK, 1996.
- [63] P. Wang, B. Xu, Lily: ontology alignment results for oaei 2009. Ontology Matching (OM), 2009.
- [64] X. Yang, M.L. Lee, T.W. Ling, Resolving structural conflicts in the integration of XML schemas: a semantic approach, in: Proceedings of the International Conference on Conceptual Modeling (ER 2003), Chicago, Illinois, United States, 2003.
- [65] V. Yannits, J.M. Renée, L. Popa, Preserving mapping consistency under schema changes, VLDB J. 13 (3) (2004) 274–293.