Received: March 9, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Known in Advance, What Ontologies to Integrate? For Effective Ontology Merging Using K-means Clustering

Ashwin Makwana[1]* **Amit Ganatra**[1]

[1]*U & P U. Patel Department of Computer Engineering, CSPIT, Charotar University of Science and Technology, India*
* Corresponding author's Email: ashwinmakwana.ce@charusat.ac.in

Abstract: A significant aspect of any system is how we present the knowledge. Ontology is one of the methods to present shared knowledge of the particular domain. Ontology can be designed and developed on specific domain or subdomain by many groups and researchers, which will create heterogeneity. To solve this problem ontology integration is necessary. To integrate shared knowledge, we require calculating the similarity between two ontologies, selected from a corpus using ontology matching techniques. Here we define a procedure to create a group of ontologies. Ontology comparison can be made using tools to find similar classes. As a similarity measure, Jaccard Similarity Index (JSI) is used create a group of ontology by an algorithm named k means. For each cluster, we generate buckets of ontologies, and from the bucket, all ontologies are grouped in one ontology reducing the attempt of exploring in enquiring understanding in multiple ontologies. Here we have check performance of ontology matching; clustering and merging algorithms script using standard benchmarking techniques of agriculture domain and conference track. Also, compare response time performance of SPARQL (SPARQL Protocol and RDF Query Language) query on merged ontology using this method. For experimentation, we have selected ontology corpus from agriculture domain and conference tack domain of OAEI (Ontology Alignment Evaluation Initiative). At the end of experimentation through our proposed approach we could achieve improvement in average loading and match time in ontology matching process compare to an existing tool. Also, we could achieve significance result in ontology merging process though benchmark parameter of coverage, compactness, and average merge time of two ontologies. Finally, in SPARQL query experimentation, we got success in improvement in reducing search space and response time of the query.

Keywords: Ontology, Knowledge reuse, Knowledge merging, Ontology matching, Semantic web, Clustering.

1. Introduction

Ontologies help to understand the similar domains in semantic web-based applications [1, 2]. In last decades ontologies is widely used as a knowledge representation in various domain and applications like semantic web, education, agriculture, healthcare, biological sciences, etc. There is a requirement of query answering machine to process complex queries of the user on a domain like agriculture, university and research conferences. Usually, this application is using SPARQL [1] query by the user. SPARQL (SPARQL Protocol and RDF Query Language) is an RDF query language, that is, a semantic query language for databases, able to retrieve and manipulate data stored in Resource Description Framework (RDF) format. To search knowledge from this multiple ontologies using SPARQL [2] queries required much time by sequentially applying these queries one by one on each ontology. For reduction in response time of the query, it is required to merge similar domain ontology into one through a reduction in search space.

An ontology [3, 4] is sharing the perception of knowledge depiction. Ontologies express a solid domain knowledge and provide a simple understanding of the same. With regards to the usage and situations, individuals or group of people develop and depict Ontologies, which may create heterogeneity in knowledge [5] represented. Added to that in the different domain also there are few concepts which are common, those concepts create heterogeneity in showcasing abstract ideas. This demand for ontology matching [5], which generates a total number of concepts matched among those ontologies. Many researchers [6, 7] have developed techniques for ontology matching and ontology alignment [8]. These techniques are required to merge knowledge present in the ontology. Ontology alignment means a representation of how different ontologies are related to each other. A set of correspondences or set of mappings [9] between two ontologies is defined as Ontology Alignment.

According to a survey recently done by [10] on ontology matching, many ontologies matching systems, approaches, and techniques developed by researchers. By ontology matching, we can reduce syntactic, terminological and conceptual heterogeneity [11, 12] present between multiple ontologies. Two Ontologies expressed in different ontology language represents syntactic heterogeneity. Terminological heterogeneity means when same entities in two different ontologies results generate variation in names. It is possible to reduce few types of heterogeneity by matching ontologies. Ontology matching described in [6, 10, and 13] using different matching algorithms. They are called as matchers. They assign a numerical value to each mapping [12]. This value represents the similarity between terms. These matchers also include element level and structure level.

Challenges to do heterogeneous knowledge integration [14] using ontology merging or ontology integration [15] motivate to develop a technique for ontology clustering based on ontology matching result. Local ontologies store information and data in their respective different formats. The problem of querying multiple local ontologies can be resolved by generating global ontologies, which can provide uniform query interface knowledge merging using ontology reuse. Ontology reuse [16] is research problem in the ontology field which consists of processes like merging and integration.

A standard platform is needed for the merging of one or more ontologies instead of unarranged merging. A well organized and structured level of merging is preferable here, and for that, we can use ontology matching where two different ontologies can be compared using similarity measures [17, 18]. It is very crucial for the understanding of the ontology.

Comparison between two ontologies can be made through Jaccard Similarity Index [5, 8] and we can produce a congregate of ontologies.

Researchers for ontology matching and alignment have developed many techniques. These techniques are required to merge knowledge present in the ontology. Out of many approaches, it is advisable to select, widely use an open source tool and techniques which gives better precision, recall, and F-measure [19]. Ontology Alignment Evaluation Initiative (OAEI) [20] is an international initiative which is doing a review of the ontology matching systems for observing and measuring the performance of the various system using empirical experiment. Many of ontology matching performance benchmark based on it is 2004–2017 campaigns of OAEI. OAEI give various tasks related to ontology matching to participant system after that results are evaluated using evaluation measures such as recall, precision, and F-measure. Out of available standard techniques and tools, Agreement Marker Light (AML) tool [21] can be adjusted [22] to find global similarities of two ontologies and thus can congregate ontologies together. OAEI also provide ontologies for experimentation and evaluation of the system. Conference track contains 16 ontologies which are suitable for ontology matching and of the same domain.

Using these techniques, we can find relevant ontologies to merge with similar or nearer ontology present in the corpus. For this, we need to develop techniques which can create clusters of similar ontology from a corpus of ontologies and merge it in single ontology cluster wise.

Now, theoretically many research groups have developed ontology matching and merging system, but for practical applications very few have developed an integrated system. For any domain, specific user it is vital to have a handy script to execute multiple ontology matching pairs at a time. Here, we try to automate ontology matching, clustering and ontology merging process in one go. For same we have developed two algorithm script and modified an existing tool. We have used Agreement Maker Light (AML) and GROM [23] tools for ontology matching and ontology merging respectively. Problem with both tools is they could do matching and merging of one ontology pair at a time. As an application developer or knowledge worker, it is required to do bulk ontology pair matching and merging. Here we have tried to overcome this problem of bulk ontology merging and generating knowledge at one single ontology, cluster wise.

The first contribution of this paper is to develop the technique is to understand the level of global similarity between two ontologies. Here, similarity measure considers ontology as a whole instead of element level similarities. The second contribution is in ontology matching, instead of matching ontology pairwise, we have developed algorithm script for matching multiple ontologies in one go and consolidate result of the matching process in single data set. Furthermore, develop a technique to do clustering using k-means which used to create buckets of ontologies based on a global similarity measure. Finally merge real-world ontologies of a specific domain, where instead of merging ontology pairwise, we have to develop algorithm script to do the merging of ontologies from given specific bucket. We check the performance of our methods algorithm by using standard benchmarking techniques suggested by literature for ontology matching [20, 24], ontology merging [25] and SPARQL query answering [26] one merged ontology.

The paper is structured as follows: Section 2 introduces the background of the work. It presents concepts and understanding of schema clustering, ontology matching, ontology mapping and ontology merging process. Section 3 presents related work on this domain of ontology clustering, ontology matching, and ontology merging. Section 4 presents our proposed approach to process flow and pseudo code description. Section 5 present experimentation setup, dataset, analysis, and result. Finally, a conclusion and future work.

2. Background and related work

2.1 XML document clustering

Clustering is a technique that involves the grouping of data points. Given a set of data points, we can use a clustering algorithm to classify each data point into a specific group. The objective of clustering is to determine the intrinsic group in a set of unlabelled data.

Ontology in semantic web represented by RDF or OWL (Ontology Web Language) [1, 2] which is one type of XML document. Many researchers have worked on document clustering, especially on in [27] XML document clustering method. XClust [27] is presenting works for clustering of XML Schemas for integration; Here they presented an approach to finding similarities between Document Type Definition (DTD) and then generate clusters of DTDs using hierarchical clustering based on similarity matrix. DTDs are grouped into clusters which facilitates integration and merging process to produce newly integrated schema. Integration is used to keep only the common DTD elements. However, this work is limited to DTD as XML format and not focus generic knowledge representation schemas.

Torres et al. have work related to similarity measures for clustering of XML documents, which represent similarity measures, but not focus on clustering of schema and knowledge integration. In [28], S. Ahmed et al., present how clustering can be useful for ontology partitioning. They have applied clustering on an individual element, i.e., concepts present in particular ontology. They used k-means clustering using various similarity measures like Jaccard, Cosine, Euclidean, Dice, Wu and Palmer Measure and Dennai Measure. Here, they focus only on the individual concept and not on ontology as a whole.

2.2 Ontology matching

Many good matching systems [5-7] were developed in the past decade; here we describe some of the famous matchers. According to past literature, using ontology matching, we can solve the problem of semantic heterogeneity. Semiautomatic tools are used for Ontology matching, or it can also be done manually. There are different approaches to match the ontologies; the primary ontology matching system makes different lexicons by using lexical matchers. Different tokens are separated from the whole ontology, and these separated tokens are matched with each other to get the similarity value. As per literature found in [5-10], this matching may be word based, string-based or structure based matching. Out of all these techniques have not shown an application of ontology matching process and also not work much on global similarities of ontology.

There has been an increase in a number of the matchers developed for ontology matching in recent times [6]. Recently L. Otero et al. [10] have done an extensive survey on the current state of the art ontology matching approaches and the application of such approaches to real-life. They have summarized that majority of researcher have done theoretical work, but very few practical, real-life application have been developed which includes DSSim, Agreement Maker, RiMOM, Falcon, etc.

Faria et al. [21] have developed ontology matching system named as Agreement Maker. AML (Agreement Maker Light) is upgraded version of Agreement Maker and is an automated ontology matching system which is extensible and efficient.

They have provided open source tool with sample ontology matching result and customizable approach by selecting multiple level ontology matching steps. The final result is shown in RDF file, but they have not provided multiple ontology support. For matching more than one ontology pairs we required to do multiple executions of the tool, we have used this tool to implement our proposed approach.

Essayeh et al. [11] proposed an automatic system for comparing heterogeneous ontologies by using various techniques of comparing entities of ontology. The structural working of an ontology can be studied using Similarity flooding algorithm. This approach matching using only structure similarity instead of customized similarity measure between two ontologies. Mecca et al. [12] have done work on mapping process of ontology. They have developed an algorithm that translates writes a mapping again, from the source schema to the target ontology and also from the source ontology to the target databases using equivalence mapping. Harmony Search based Ontology Mapping (HSOMap) has been proposed by Forsati et al. [29] which is a useful ontology mapping technique where the similarities of the ontology entities are specified using many rating functions, and also they compared HSOMap algorithm performance with other method using benchmark datasets. A Compact Interactive Memetic Algorithm (CIMA) based Xue has proposed collaborative ontology matching technology. Et al. [30] which simplifies shared ontology matching. Ceron et al. [31] present the classification model to work with instances from various ontologies.

COGOM [32] is a reasoning based ontology matching system which is quite adaptive. In this, OAEI 2015 datasets are utilized and using the precision and recall metrics; the overall effect can be enhanced. However, still, the primary focus is not on query answering system and also not able to do multiple ontology pair matching.

Ngo et al. [33] offered better basic matcher and framework in a tool named as YAM++. The latest version of YAM++ obtained great matching results in comparison to OAEI datasets. YAM++ is matcher producing a better result which uses standard algorithms for matching, which consolidate those algorithm to match ontologies. This matcher provides self-configurable and flexible in user preferences by the customized matching approach. However, this tool is not the fully open source and not providing permission to change the source code for customization. It supports one pair ontology at a time.

CroMatcher [34] is an automatic ontology matching system which denotes the resemblance or correlation between the entities of two different ontologies. It analyses the arrangement delivered by the matchers and highlights the one with the unique and particular arrangement. CroMatcher is not doing ontology merging and processing queries, its only focus on ontology matching system.

Ruiz and Grau [35] have developed LogMap, which have used reasoning using logic based semantics for better alignments. The LogMap can be scaled, and it is an ontology system based on logic, which participates in OAEI since past seven years all tracks and giving top performance. The main disadvantage of this system is if ontology is lexically disparate or missing lexical information then it will not give better performance as they use similarities between vocabularies for ontology matching.

2.3 Ontology merging

Ontology-based data integration includes the use of ontology to combine data or information from multiple heterogeneous sources effectively [14, 15]. It is one of the multiple data integration approaches, and its effectiveness is closely related to the stability and delivery of the ontology used in the integration process. Hitzler et al., [36] illuminate that how the converging of ontologies by the pushout development from class. Hence, we see class theory as a comprehensive "meta specific vernacular" that engages us to decide properties of ontological associations and advancements in a way that does not depend upon a particular execution. This can be refined since the central objects of gather in class speculation are the associations between various ontological conclusions, not the internal structure of a lone learning depiction. The procedures of ontology arrangement and consolidating are typically taken care of manually and regularly constitute an expansive segment of the sharing procedure. Noy et al., present [37] PROMPT algorithm is used which can semi-automatically merge and align ontologies. PROMPT plays out a few assignments consequently and guides the client in performing different tasks. PROMPT likewise decides irregularities in the condition of the ontologies, which result from the client's activities, also, recommends approaches to correct these irregularities. PROMPT can be used across various platforms as it is based on a very generic model. Techniques that PROMPT uses to direct a client to the following conceivable purpose of blending or arrangement, to recommend what activities ought to be performed there, and to perform specific activities naturally.

Nováček and Smrž [38] presents a novel portrayal of indeterminate learning in the space of programmed ontology securing that contains help for Semantic Web applications. The primary target of the ontology procurement stage OLE is to execute a framework that can consequently make and refreshing space specific ontologies for a given area of the scientific learning. The procurement procedure is incrementally helped by the learning as of now put away in the ontology. They introduced the ANUIC system that arrangements with unverifiable learning in ontologies. The theoretical foundation of the fuzzy sets system permits to build up suitable math and successively fabricate new deduction devices to reason among the ideas put away in expressive ANUIC organize efficiently.

In this paper, Mezzi and Nadji [39] acquaint another route with combine OWL ontologies by the semantic change of initial ontologies. To add a semantic estimation to the merger, their approach in perspective of semantic progression of starting ontologies. This process is refined by propelling starting ontologies by a course of action of initial ontologies that clarify their thoughts with proportional words for each thought. They have used methods for the use of WordNet, or semantic headway of an expert, by then it delivers a thesaurus for each close-by reasoning to build the overall thesaurus. Our procedure fixates on enrolling semantic similarity between thoughts of ontologies, and in light of a weighted blend of enlisting resemblance systems, they use syntactic, lexical, assistant and semantic techniques, for making the correspondence arrange; from this last, we deliver the united ontology.

Ontology merging can be performed with the use of typed graph grammars done using the techniques SPO (simple Push Out) and GROM (Graph Rewriting for Ontologies Merge) [23]. The AGG (Algebraic Graph Grammar) is an application of algebraic outlook to graph transformation. It includes the two approaches OWLToGraph and GraphToOWL. Type Graph Grammar is used here in the ontology merging process contains three significant steps including searching the correspondences between the nodes from the ontologies, Merging the ontology's structure with the use of SPO and using semantic relations to enhance the combined ontology.

Fu e. al. [41] has provided a semi-automated approach for ontology merging which assimilates information from the uncertain data. Specific Investigations were made on this approach, and it was found out that this approach offers a practical approach to coordinate information and for active learning of the information. Maree et al. [42] have introduced a wholly automated approach for domain-specific ontology merging which can be grouped into three classes further: Single technique based methodologies, procedure-based methodologies and semantic assets using methodologies. Fahad et al. [43] show a system of commonly identifying semantic irregularities in the prior stages of ontology merging. The DKP-AOM process is introduced here which improves the knowledge and consolidation of data.

The ATOM approach automatically merges the source ontology into a target ontology proposed by Raunich et al. [44]. This is a target driven approach that integrates source scientific classification into the objective classification. The ATOM approach could be efficiently linked to substantial genuine scientific categorizations from various areas.

2.4 Benchmarking techniques

Yingjie Li et al. [24] paper, proposed a multi-ontology guideline for the Semantic Web frameworks. This benchmark takes a two-level customization display including the web profile and the ontology profile as its sources of info and creates client redid ontologies. In this paper, proposed a multi-ontology benchmark for the Semantic Web frameworks. One critical utilize case for the Semantic Web is the reconciliation of information crosswise over numerous heterogeneous ontologies. This work is not providing how to match and merge multiple ontologies, but multi ontology benchmark helps us to define our proposed work performance issues.

Raunich et al. present in [25] ontology merging benchmarking approaches. They present that the critical issue is that, for a merge task either is performed symmetrically or asymmetrically. They have presented two metrics, the compactness of the merge result and the degree of redundancy. For reduction in the amount of redundancy, it is required to improve symmetric merge approach whereas asymmetric merge approaches fully preserve only one of the input ontologies which are useful for many applications. The main disadvantage is this approach not include coverage of ontologies and also not focused on query processing on individual vs. merged ontologies.

In [45] E. Daskalaki et al. survey to present the benchmarking techniques, for instance, matching for Linked Data by discussing its principles, dimensions, characteristics and providing a survey of benchmarks and generator of the benchmark. They consider the presented benchmarks from the standpoint of the systems to identify the appropriate benchmark for a given setting. This work considers benchmarking, for instance, matching only and not considering ontology as a whole as global ontology similarity measure. Duchateau et al. [46] undertaken to study and evaluation of data integration as a Schema Integration. Two benchmarks have been presented, minimality and completeness. These measurements are collected to assess the closeness between two outlines. They have assessed the tools which are used for schema matching COMA++ and Similarity Flooding, more than ten datasets. However, this work does not cover the coverage of two ontologies and generic for schema matching.

Madhfoudh et al. [47] introduced a measure for merging of ontologies that specify different ontologies types: scientific classifications, lightweight ontologies, heavyweight ontologies and multilingual ontologies. They propose a benchmark for ontologies merging, which contains different ontologies sorts. They additionally indicate how the GROM instrument (Graph Rewriting for Ontology Merging) can address the combining procedure and assess it given scope, repetition and rationality measurements. We have used this approach of benchmark in our proposed work to check the quality of merged ontology, but the limitation of GROM approach is it can only give merging benchmark for one pair of ontology merging at a time, and not supporting multiples ontology merging.

2.5 Use of SPARQL query in ontology alignment

Alessandro et al. in [26] present about a novel OAEI track for ontology alignment for query answering (OAQQA) in 2014 and 2015 OAEI Campaign From the initial assessment the first limits of the conventional assessment were clearly identified. They used Conference track ontology dataset with a set of queries. This work is used only for benchmarking alignment and matching process rather than global merged ontology query, but it is beneficial for our work to decide benchmark query performance comparison on conference track.

Thiéblin, Élodie, et al. presented in [48] presented technique for rewriting SPARQL queries on complex correspondences which in format 1:n between overlapping ontologies are used for SELECT SPARQL queries which can be presented in the Linked Open Data through different RDF datasets. Two datasets have been used to assess the approach out of which one is from the agriculture domain, and another is built on a reduced set which involves the ontologies from the OAEI conference track. Thiéblin, Elodie, et al. [49] present that how to process to query on lined open data dataset from a known ontology using composite alignments. This method as applied to Agriculture domain ontologies. They presented the use of composite alignments for a SPARQL rewriting approach used in agronomic LOD (Link Open Data) sources. In both papers, they have focused only on rewriting query but not response time improvement for single vs. multiple merged global ontology. However, we have used in our work their data set and query for performance check.

Xavier et al. in [50] paper concentrate on an arrangement strategy for these ontologies given Formal Concept Analysis, an information examination procedure established on cross-section hypothesis, and a technique for handling user query. They have focused on various. Agriculture domain ontologies and subdomain as well. The outcomes demonstrated that merging two ontologies in partial ordering, given by a cross-section of terms. Given this examination, one can assess and choose which set of items is the best response for a given user query. This work does not provide merging multiple ontologies at a time using global measures and also not able to give performance check on query processing.

3. Proposed system

In this section, we have presented a new approach to do merging and integration of ontology knowledge using global similarity measure found from ontology matching process.

3.1 Steps for knowledge integration in ontologies

Knowledge integration requires the following process which integrates knowledge presented in different ontologies. The above-mentioned steps to be performed on a specific domain of ontology collection. (see Fig. 1).

First, from the ontology Corpus, a pair of ontology, i.e., source ontology and target ontology are separated. This two pair of ontologies are then matched using ontology matching algorithm, and multiple RDF files are generated. In the next step, the ontology mapping RDF file is merged into single CSV File. Further, the Jaccard similarity Index is found out using the Jaccard Similarity Calculation. The ontologies are then clustered or collected using the K means clustering technique. From the cluster, an individual pair of ontologies are separated and collected into buckets, and individual merged

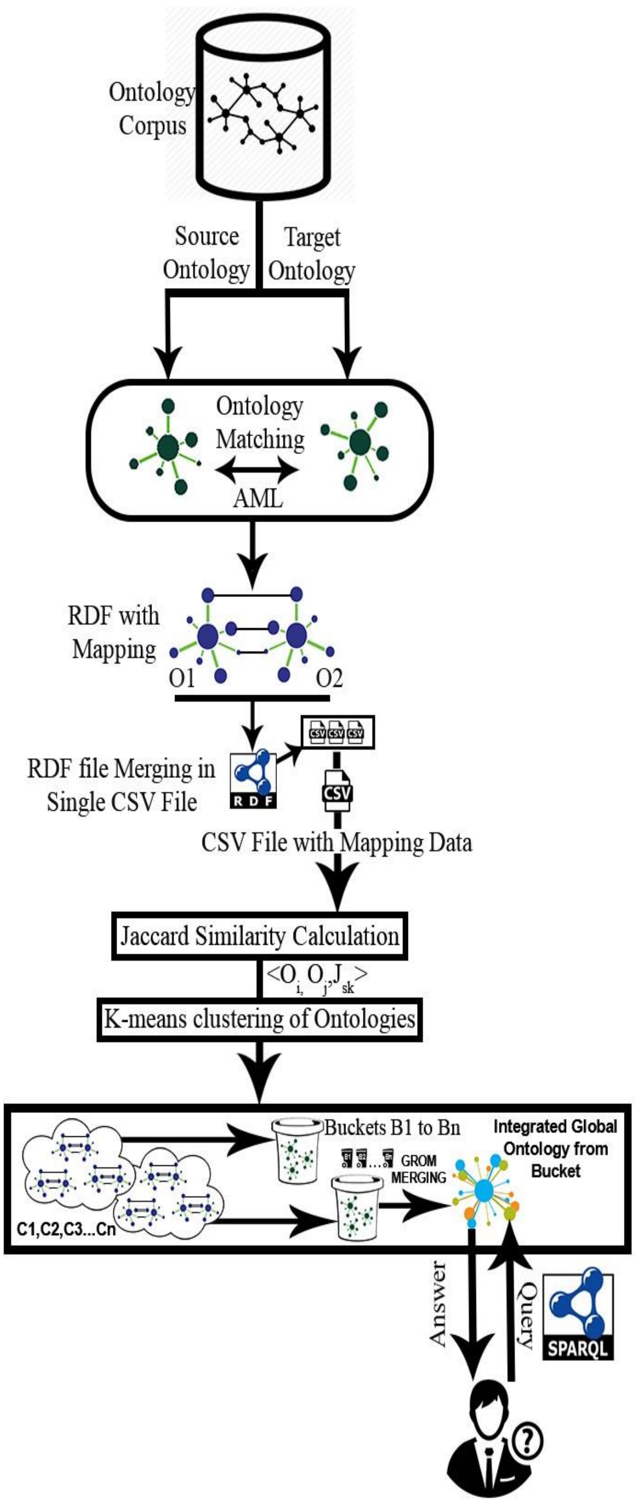
Ontology Corpus
Source Ontology
Target Ontology
Ontology Matching
AML
RDF with Mapping
O1
O2
RDF file Merging in Single CSV File
CSV File with Mapping Data
Jaccard Similarity Calculation
<Oi, Oj, Jsk>
K-means clustering of Ontologies
Buckets B1 to Bn
GROM MERGING
Integrated Global Ontology from Bucket
C1,C2,C3...Cn
Answer
Query
SPARQL

Figure. 1 Flow diagram of the proposed system

ontology is extracted from their respective bucket. The user queries these merged ontologies.

3.2 Global similarity measure on ontology clustering

For generating a cluster of ontologies, we require a pool of ontologies know as Ontology Corpus, which can be defined as OC [O_1.O_N]. For the dataset, we have used the domain-specific ontologies used by OAEI portal.

The Agreement Maker Light (AML) is an ontology alignment tool for finding the Jaccard Similarity Index. From AML tool, we get mappings between any two ontologies from the corpus. From the ontology corpus from the same domain in one particular set C to match each ontology O with all ontologies C-O*i*, the work carried out on some ontologies (O_1 *to* O_N).

For matching two ontologies we can use the Global similarity index as Jaccard Similarity index [5, 8], we can calculate this index using the following equation-2. We present similar (matched) mapping using *(o_1*o_2)*. |*x*| and |*y*| is a total number of classes, properties and individual in O_1 and O_2 respectively, which can be identifying here with |o_1| and |o_2| respectively.

$$JS\ (O_1, O_2) = \frac{(o1 \times o2)}{|o1|+|o2|-(o1\times o2)} \tag{1}$$

For Jaccard similarity index, we generate few numerical values from source ontology, which is a number of classes of source ontology-C_s, number of properties of source ontology-P_s, and number of individual of source ontology-I_s. We required to the summation of all these three numbers of source ontology, i.e., $C_s+P_s+I_s$. Moreover, also from target ontology that is classes of target ontology-C_t, some properties of target ontology-P_t, and some individual of target ontology-I_t. We required to the summation of all these three numbers of target ontology, i.e., $C_t+P_t+I_t$.

This bucket B_1 *to* B_m corresponds to clusters C_1 *to* C_m respectively. In these buckets, we will insert SO and TO from respective cluster. $<O_{im}, O_{jm}, S_{km}>$ which denotes which connect SO and TO in m^{th} Cluster C_m *(m= 1 to M).*

From ontology matching tool, we find the total number of mapping that is *M*, which is equal to the summation of the number of class map *Cm*, a number of properties map *Pm* and number of individual map *Im*; between source ontology and target ontology.

From this, we can rewrite Eq. (1) as below:

$$JS_k\ (O_i, O_j) = \frac{(Cm+Pm+Im)}{(Cs+Ps+Is+Ct+Pt+It)-(Cm+Pm+Im)}$$
$$\text{Where } 0< JS_k < 1;\ i\neq j\ (i,j = 1,2 \ldots,n),\ k= 1,2\ldots P \tag{2}$$

Eq. (2) will be used to find Jaccard similarity index between two ontologies *O1* as source ontology and

O2 as target ontology. The value of JS_k is between 0 and 1. If there is no mapping between two ontologies, then the value of JS_k will be 0. If all class, properties, and individual are similar and map then the value of JS_k will be 1.Otherwise, for rest of another case, JSk is between, 0 to 1 depending upon some mapping between to ontologies *O1* and *O2*.Above Eq.-2 gives triplets $<Oi, Oj, S_k>$ where S_k represent k^{th} Jaccard Similarity of particular ontology *Oi* and *Oj*.

The k-means clustering problem can be solved using The Lloyd's algorithm (k-means algorithm), K-means is the most straightforward algorithm and easy to implement, which uses unsupervised learning method. It works well with large datasets. Its result is easy to interpret for clustering. K-means algorithm is fast and efficient regarding computational cost for one-dimensional data is ordered. The complexity of k-means is $O\ (n\times k\times i)$.

Using Jaccard similarity index, we create a cluster of an ontology using K-means algorithm. Using orange tool we have to provide a dataset of our ontology Jaccard similarity index values of various Ontology pairs from the corpus.

Using Orange tool, we provide a dataset of our ontology Jaccard similarity index values of various ontology pairs from the corpus. Generate clusters of ontology based on Jaccard similarity index field using K-means algorithm. After generating *M* number of clusters, we can identify corresponding *SO* and *TO* pair from which we have selected from a corpus of Ontology. Finally, we can identify buckets of ontologies, which we create from these pairs of *SO* and *TO* from respective Cluster.

$$B_m = \{ O_{im} \mid (O_{iml}, O_{jmr})\text{- unique pair, } l = 1 \text{ to } L, r = 1 \text{ to } R\} \text{ Where, } |B_m| \le L+R < N \quad (3)$$

L=Number of Source ontologies and R=Number of target ontologies in above triplets.

Finally, a bucket of similar ontologies merged in the single ontology using Protégé tool. After merging of two or more ontologies then we can check ontology pairs having zero or less Jaccard Index, then some axioms, classes, and elements in individual ontology total are equal to merged new ontology. While in case of Jaccard index is more than zero or high then some axioms, classes, and elements in individual ontology total are more significant than merged new ontology.

$CO_1 + CO_2 = CO_m$ *(If Jaccard Index =0)*
$CO_1 + CO_2 > CO_m$ *(If Jaccard Index>0)*

Here, CO_1, CO_2 is some classes in ontology from the unique pair, CO_m is some classes in merged ontology from CO_1 *and* CO_2.

However, ontology presented in buckets is likely to be similar in nature, so we can use these buckets to do any type of research on query answering through ontology integration.

3.3 Algorithms

3.3.1. Main algorithm

```
Integration_Knowledge (OC [O_1--- O_N])
Input: Ontology Corpus- OC (domain specific
ontology corpus of O_1.....O_N and Ontology features
(#class, #properties, #individuals) of unique pair
source and target ontologies.
Output: Integrated Ontologies, SPARQL Results,
Matching Results, Merging Results
1.  For each i & j=1 to N
2.  OM[4]=AML_OMCH(<Oi,Oj>)
3.  OM[0]=FSs, OM[1]= FSt, OM[2]= FSm
4.  Feature_struct FSx= {*Cx[ ],*Px[ ],*Ix[ ], |Cx|,
    |Px|, |Ix| }
5.  where x=s / t / m
6.  For each p=1 to P
7.  $JS_p (O_i, O_j) = \frac{(|Cm|+|Pm|+|Im|)}{(|Cs|+|Ps|+|Is|+|Ct|+|Pt|+|It|)-(|Cm|+|Pm|+|Im|)}$
8.  (Oi[p]=Oi, Oj[p]=Oj, $JS_p$ [p]=JSp)
9.  CLSTR[$C_1..C_z$]=Generate_cluster_Kmeans(Oi[
    p], Oj[p], JSk[[p])
10. Generate TABLE-
    1[CLSTR[],Oi,Oj,OM[1…3], $(|Cm|+|Pm|+|Im|)$,$JS_p$[p])]
11. End For
12. End For
13. For each x=1 to z //where z is number of
    clusters
14. BKT[x]←CLSTR[x]
     where BKT[$B_1...B_z$] , $B_z$= { $O_{iz}$ / ($O_{izl}$, $O_{jzr}$) is
    unique pair, l= 1 to L, r=1 to R}
     Where, z=|BKT[]| ≤ L+R < N
15. Generate TABLE-2[BKT[$B_x$],CLSTR[$C_x$], $O_{iz}$,
    |$O_{iz}$|, |($O_{izl}$, $O_{jzr}$)|]
16. $O_{IM}$[iz],T_$MRG_{iz}$=GROM_OMRG($O_{izl}$[], $O_{jzr}$[])
17. End For
18. TABLE-6[Qi(X),Time_exec(Qi,Oi)] =
    Exec_Query(Qi,Oi)
19. TABLE-6=+TABLE-6
    [$Q_{IM}$(X),Time_exec($Q_{IM}$,$O_{IM}$)],Prc,Rcal,Fm]
```

3.3.2. Algorithm 1

```
AML_OMCH(<Oi,Oj>)
Input: Oi, Oj
```

```
Output:OM={FSt, FSs, FSm}
1. S=Oi, T=Oj, Feature_struct
   FSx{*Cx[ ],*Px[ ],*Ix[ ], |Cx|, |Px|, |Ix| }
2. FSs ← {* Cs[ ],* Ps[ ],* Is[ ], |Cs|, |Ps|, |Is|} ← Parse(S)
3. FSt ← {* Ct[ ],* Pt[ ],* It[ ], |Ct|, |Pt|, |It|} ← Parse(T)
4. FSm ← Lexical_Matcher(FSs,FSt)
5. FSm ← FSm + Mediating_Matcher(FSs,FSt)
6. FSm ← FSm + Word_Matcher(FSs,FSt)
7. Where finally, FSm={* Cm[ ],* Pm[ ],* Im[ ], |Cm|, |Pm|, |Im| }
8. T_MCH ← Execution Time of Matching process from Step 1 to 6.
9. Return OM[4]={FSt, FSs, FSm, T_MCH}
```

3.3.3. Algorithm 2 (Generating clusters)

```
Generate_cluster_Kmeans(Oi[ ], Oj[ ], $JS_p$ [ ])

1. Clusters the JSp[] into k groups
2. Consider k points at random as cluster centres
3. Assign JSp[] element to their closest cluster centre by the Euclidean distance.
4. Generate centroid of all elements in each Ci cluster.
5. Repeat steps 2, 3 and 4 until the same points are assigned to each cluster in consecutive rounds.
6. Return (CLSTR[$C_1$...$C_k$]).
```

3.3.4. Algorithm 3 (GROM_OMRG)

```
GROM_OMRG (Oi[], Oj[])
1. CM[ ], EQ[ ], SY[ ], SB[ ]
   CM-commons, EQ- syntactically equivalent, SY-synonyms, SB-share subsumption relation
2. For each i=1 to z //z is number of ontology pair
3. For each j=1 to z
4. For each N in EQ
5. O'1 ← SPO_Ren_Ent (O1, EQ{Oi}, EQ{Oj} );
6. End For
7. CM ← CM U EQ{Oi} ;
8. CMNO ← Generate common ontology;
9. $GLBO_{ij}$ ← SPO_Mrg_Grph (Oi, CMNO, Oj);
10. For each N in SY
11. $GLBO_{ij}$ ← SPO_AdEq_Ent (GLBO, SY {Oi}, SY {Oj});
12. End For
13. For each N in SB
14. $GLBO_{ij}$ ← SPO_AdSb_Cls (GLBO, SB{Oi}, SB{Oj} );
15. End For
16. $T\_MRG_{ij}$ ← Calculate time of Merging Two Ontologies Oi,Oj
17. $O_{IM}$[i] ← $GLBO_{ij}$,T_MRG[i] ← $T\_MRG_{ij}$
18. End For
19. End For
20. Return ($O_{IM}$[],T_MRG[])
```

3.3.4. Algorithm 4 (SPARQL query)

```
Exec_Query(Qi,Oi)

1. Open Ontology (Oi)
2. Execute SPARQL query engine for Qi query on Oi
3. Get result set Qi(x)
4. T_Exe(Qi, Oi) ← Calculate time of execution of Qi on Oi
5. Return [Qi(x), T_Exe(Qi,Oi)]
```

3.4 Performance analysis and comparatives methods

For particular domain, it is difficult to find single ontology consisting of all relevant information and knowledge, especially for query answering. Many people [18 - 20] have developed different ontologies in same domain or subdomain, and it is also possible that one person or group have developed different ontologies for same domain or subdomain. For query answering searching information and knowledge in these multiple ontologies is a strenuous and time-consuming task. To resolve this issue many researchers, have work on ontology matching and ontology merging techniques. Some of them have also suggested benchmarking techniques to check the performance of both processes. Here, we use similar techniques to check the performance of our proposed approach, we have used standard ontology matching and merging benchmark. In addition to that to check performance, we have fire SPARQL query one by one on various ontologies of a particular domain and compare it with our merged ontology CO_m.

3.4.1. Comparative of ontology matching

Ideally, for ontology matching performance evaluation [45] precision, recall and F-measures are used as performance parameters. Instead of that we have focused here on loading time and matching time. Ontology matching results are compared with existing state of the art matching system regarding some ontologies match. Ideally, any ontology matching system can match at a time sing pair of ontology and generate a result. Using our approach, we could do more than one pair of ontology. For experimentation, we have use case study example of 20 ontologies of agriculture domain and 17 ontologies from OAEI conference track. Using our approach we could reduce the average loading time of source and target ontology and also drastically reduce average matching time. We have also provided the consolidated result of multiple ontology pairs regarding mapping between each pair.

3.4.2. Comparative of ontology merging

Similar to ontology matching, we could also enhance the performance of merging system for merging multiple pairs of ontology at a time, instead of the individual merging of ontology pair. That will help users to merge multiple ontologies on single click, using this average approach time of merging two ontology. We have used benchmarking technique present in [24, 25, 46, 47].

There are few metrics to assess ontology merger quality which includes coverage (completeness), compactness and redundancy (minimality). Equations related to all these three metrics are explained in the paper [25, 47].Coverage means the degree to which source and target ontology concepts are preserved in merge result ontology which is between 0 and 1. Coverage 1 means cover all concepts of input ontology, and 0 means no input ontology concepts covered. Compactness is to check the size of the created merge ontology which is related to its understanding ability because merge solution should not be that large. Absolute result size refers to the number of concepts in the resulted ontology. Relative result size is a ration of input source ontology plus target ontology versus full merge result. Redundancy checks that no redundant concept appears in the integrated merge ontology. Reduce the degree of redundancy or semantic overlap in an integrated ontology for improving understandability. A value is of redundancy higher than one means a redundant path for merge concepts, which is not good for understandability. If the value is one, then it means avoidance of redundancy.

We have discussed earlier that one of the applications of ontology matching and merging process is query answering system. For same, we required using SPARQL query on OWL ontology to find the answer from knowledge-based of a particular domain. We have used SPARQL query performance evaluation method suggested in [26, 48-50] for performance evaluation of ontology merging. We specifically want to showcase that using our approach search space for multiple ontologies is reduced and because of that querying in multiple ontologies, because of that processing time and searching time will be reduced in any query answering system.

In OAEI evaluation metrics used for the Ontology Alignment for Query Answering (OA4QA) track [26] are not only on classical methods like recall, precision, and f-measure but also on the result set of the query assessment. To generate a reference and a model set for the results of the query, we have used the reference alignment of the Conference track ontologies.

4. Experimentation and results

In this section, we describe experiment setup, implementation, and result based on our proposed approach. We have selected ontologies from agriculture domain and conference domain of conference track of OAEI. We have compared existing related work with our proposed approach in a three-way. One is to improve in ontology matching time, second in improvement in ontology merging time, minimality, coverage and compactness of merged ontology. Third in a response time of SPARQL query in multiple individual ontologies with single global ontology created by our approach.

4.1 Experimentation setup

Agreement Maker Light (AML) is an open source tool for Ontology matching. The Machine Configuration used to implement and test the proposed design was a 2.50 GHz Intel Core i5 processor, 16 GB RAM and Windows 10 OS. On this machine we have installed JVM 1.8, JDK 1.8 to use programming language JAVA 8 using IDE NetBeans-8.01 which is used to modify open source tool AML. We have done experimentation of K-means clustering using ORANGE 3.4.1 in which we gave input a CSV file and got excel file as an output of Cluster details. We have used Notepad++ v 6.9, Protégé tool for Ontology editing and visualization. We also used Java to create a script for executing AML tool and generating CSV file as an output.

4.2 Framework for implementation

The AML (Agreement Maker Light), ontology matching system, is an open source tool which can be implemented with NetBeans IDE tool and Java8. It is one of the leading tools for ontology matching [19]. For the ranking of ontology matching tools, there is a guideline called Ontology Alignment Evaluation Initiative (OAEI). It contains different matching algorithms which do automated matching. Different matchers include lexical matcher, structural matcher, string matcher, word matcher, background matcher, property matcher. It also has filters for obsolete, cardinality and coherence filtering. Different sets of ontology belonging to different domains were given as input to the widespread implementation of AML tool. The AML tool takes a pair of ontologies as input to the system and outputs alignment value.

GROM is one more tool which we have used for merging of ontology pair. GROM apparatus which is being actualized here which created a global ontology from given two ontologies and their mapping in a planned way. We have modified this tool using our algorithm for calling merge module of this tool for multiple ontology pair merging. This tool we have applied to a bucket of ontology created by ontology clustering technique. GROM consists of three main steps: i.e., Similarity search, ontologies merging and global ontology adaptation.

4.3 Dataset

For ontology matching and alignment tools benchmarking, there is a standard dataset in OAEI. The Agriculture and conference domain were selected. These ontologies were downloaded OAEI [22], OBO-Foundry [51], Bio-Portal [52] and Agro-Portal [53] which has ontology belonging to different domains. We have selected around 17 small size ontologies of OWL format of Agriculture domain and 20 ontologies of Conference tack of OAEI.

4.4 Steps for implementation

For the implementation of our technique, it is required to do experimentation by some of the tools on a specific domain. Following steps are necessary to create clusters of ontologies:

1. Select a path of ontology corpus directory of the particular domain.
2. Loading source and target ontologies in AML
3. Apply auto-matching on AML of those pairs one by one
4. Save Alignment results, i.e., mapping in one CSV file with details of # of Classes, # of Properties and # of Individuals in each ontology of particular pair with the mapping between them.
5. On this CSV file apply ORANGE data mining tool for clustering of ontologies using k-means clustering method.
6. Divide ontology corpus directory in multiple subdirectories for each bucket, which generated from each cluster of ontologies.
7. For each subdirectory, apply ontology merging tool- GROM to convert into the merged global ontology. For this provides a path of each subdirectory to GROM tool to merge multi-pair ontology.

Finally, apply SPARQL query one by one on each ontology of corpus subdirectory then, check result and response time. Also, apply the same query on global merged ontology and check result and response time.

4.3 Results and discussion

We have shown here sample data and result, as it is difficult to put all dataset and result set.

Table 1 describe how we can generate clusters of ontology. It is based on Jaccard similarity index field using K-means algorithm. Table 1 describes 11 different columns representing stable Data about the task done. The further second column describes the source and the target ontologies. For example for the agriculture domain for N=20 is some sample ontologies from the corpus, then here P = $_NC_2$ = 380. Similarly for the conference domain, for N=17, then P = $_NC_2$ = 136.

Jaccard Similarity Index is used to calculate the Global Similarity measure between two Ontologies. In Table 2, the Jaccard Similarity Index can be calculated. Then K-means algorithm is used to create the cluster of ontologies. Then from each cluster, there is a bucket generated. Table 2 describes the name of all ontologies in particular bucket, number of ontologies pair per bucket and number of different and unique ontologies in the bucket.

The result obtained from the Ontology Integration is shown in Table 3 describing data for Ontology 1 # of Classes, Ontology 2 # of Classes, Common Classes in both ontologies, Total Class in both ontologies, Jaccard Similarity Index and Integrated Ontology #of Classes.

Table 4 is presenting ontology matching time for ontology pair using existing AML method [19 - 21] and our approach which is modified using algorithm script for multiple pairs of ontology. We can see from this table that average ontology matching time and loading time for both domains of an ontology using our algorithm method is less compare to the pairwise matching of AML tool. From last three column of the table, we can observe proposed cluster approach for AML tool give significant improvement in average loading time, i.e., 61% and 57% respectively for agriculture and conference domain. At the same time, we can observe that average matching time is improve using our proposed approach 20% and 28% respectively for agriculture and conference domain.

Table 1. Sample experimentation data for Calculating Jaccard similarity index and clustering

Cluster	Source-Onto	Target-Onto	Cs	Ps	Is	Ct	Pt	It	Mapping	Jaccard Similarity Index
C1	cmt	crs_dr	29	59	0	14	17	0	9	0.082
C1	conference	confOf	59	64	0	38	36	0	15	0.082
C2	edas	openconf	103	50	114	62	45	7	5	0.013
C2	crs_dr	openconf	14	17	0	62	45	7	2	0.014
C2	linklings	openconf	37	43	5	62	45	7	3	0.015
C2	iasted	paperdyne	140	41	4	45	78	0	5	0.017
C2	iasted	pcs	140	41	4	23	38	0	4	0.017
C3	edas	pcs	103	50	114	23	38	0	11	0.035
C3	confOf	edas	38	36	0	103	50	114	12	0.036
C3	openconf	pcs	62	45	7	23	38	0	6	0.036
C3	conference	edas	59	64	0	103	50	114	14	0.037
C4	ekaw	openconf	73	33	0	62	45	7	4	0.019
C4	confOf	iasted	38	36	0	140	41	4	5	0.02
C4	crs_dr	edas	14	17	0	103	50	114	6	0.021
C4	edas	sigkdd	103	50	114	49	28	0	7	0.021
C5	crs_dr	myreview	14	17	0	38	66	2	12	0.096
C5	confOf	ekaw	38	36	0	73	33	0	16	0.098
C6	cmt	iasted	29	59	0	140	41	4	6	0.022
C6	confOf	openconf	38	36	0	62	45	7	4	0.022
C6	edas	micro	103	50	114	31	26	4	7	0.022
C7	crs_dr	sigkdd	14	17	0	49	28	0	6	0.059
C7	ekaw	myreview	73	33	0	38	66	2	12	0.06
C7	iasted	sigkdd	140	41	4	49	28	0	15	0.061
C7	linklings	myreview	37	43	5	38	66	2	11	0.061
C7	micro	pcs	31	26	4	23	38	0	7	0.061
C7	conference	crs_dr	59	64	0	14	17	0	9	0.062

Table 2. Summary of buckets generated from clusters

Bucket	Cluster	Ontologies	Number of ontology pairs	Unique ontologies in Bucket
B1	C1	Cmt,crs_dr,conference,confOf	2	4
B2	C2	Edas,openconf,crs_dr,linklings ,iasted,paper dyne,pcs	5	4
B3	C3	pcs,confOf,openconf,conference,edas	4	5
B4	C4	Ekaw,openconf,confOf,iasted ,crs_dr,edas , sigkdd	4	7
B5	C5	crs_dr myreview confOf,ekaw	2	4
B6	C6	cmt,iasted,confOf,openconf,edas,micro	3	6
B7	C7	sigkdd,ekaw,myreview,iasted,linklings,micr o,pcs,conference,crs_dr	6	9

Table 5 is presenting ontology merging result using standard benchmarking parameter [25, 47] like completeness, compactness, and redundancy. Apart from that we also measure average merging time of ontology pair for existing GROM tool vs. our algorithm applied to GROM tool for merging multiple ontologies. From last two columns, we can observe that in average merging time there is 33% and 31% improvement using our proposed approach in agriculture and conference track respectively. We have compared the result of GROM [25] with our work especially benchmark parameters like coverage, compactness, and redundancy. Coverage is as good as pure GROM tool, but it somewhat reduces because to avoid redundancy. Compactness is also reduced, that is around 20% that means it improve understandability of merged result. Redundancy we could not improve using our approach, but there is no loss of concept.

Table 3. Sample integrated ontology results

Cluster	Source-Onto	Target-Onto	Ontology 1 # of Classes Cs	Ontology 2 # of Classes Ct	Common Class	Total Class	Jaccard Similarity Index	Integrated Ontology # of Classes
C1	cmt	crs_dr	29	14	9	43	0.082	34
C1	conference	confOf	59	38	15	97	0.082	82
C2	edas	openconf	103	62	5	165	0.013	160
C2	crs_dr	openconf	14	62	2	76	0.014	74
C2	linklings	openconf	37	62	3	99	0.015	96
C2	iasted	paperdyne	140	45	5	185	0.017	180
C2	iasted	pcs	140	23	4	163	0.017	159
C3	edas	pcs	103	23	11	126	0.035	115
C3	confOf	edas	38	103	12	141	0.036	129
C3	openconf	pcs	62	23	6	85	0.036	79
C3	conference	edas	59	103	14	162	0.037	148
C4	ekaw	openconf	73	62	4	135	0.019	131
C4	confOf	iasted	38	140	5	178	0.02	173
C4	crs_dr	edas	14	103	6	117	0.021	111
C4	edas	sigkdd	103	49	7	152	0.021	145
C5	crs_dr	myreview	14	38	12	52	0.096	40
C5	confOf	ekaw	38	73	16	111	0.098	95
C6	cmt	iasted	29	140	6	169	0.022	163
C6	confOf	openconf	38	62	4	100	0.022	96
C6	edas	micro	103	31	7	134	0.022	127
C7	crs_dr	sigkdd	14	49	6	63	0.059	57
C7	ekaw	myreview	73	38	12	111	0.06	99
C7	iasted	sigkdd	140	49	15	189	0.061	174
C7	linklings	myreview	37	38	11	75	0.061	64
C7	micro	pcs	31	23	7	54	0.061	47
C7	conference	crs_dr	59	14	9	73	0.062	34

Table 4. Ontology matching time result

Performance Parameter	Domain	Approaches		
		Simple AML	AML-Cluster Script	% improvement
Average Match Time / Pair (ms)	Agriculture	443	356	20%
	Conference	234	168	28%
Average Load Time / Ontology (ms)	Agriculture	99	38	61%
	Conference	67	29	57%

Table 5. Ontology merge results

Parameter	Ontology Domain	Approaches	
		Simple GROM	GROM-Cluster Script
Average Merge Time (ms)	Agriculture	345	232 (33% improve)
	Conference	157	108 (31% improve)
Coverage	Agriculture	0.95	0.9
	Conference	0.91	0.9
Compactness	Agriculture	1.2	0.8
	Conference	1.0	0.85
Redundancy	Agriculture	0.2	0
	Conference	0	0

Table 6. Ontology SPARQL query results

Query	#q(x) Reference	#q(x) Merge onto	Total Response time Input Reference ontologies	Parameters			Response time – Cluster Merge ontology	% improvement in Response time
				Precision	Recall	F-measure		
Q1	98	145	796 ms	1	1	1	234 ms	70%
Q2	53	87	967 ms	0.	1	0.83	322 ms	67%

Table 6 is describing SPARQL query result, which shows some records given by particular query and query response time for a particular query of on reference ontology pair and merged global ontology using clustering. We have also shown standard parameter used in OAEI like precision, recall, and F-measure. Using our proposed method we could improve the response time of Query [48] Q1 and Q2 respectively by 70% and 67% respectively for conference domain. We have compared this result and approach of query evaluation done by existing work done by in OA4QA tack in OAEI [26, 48-50].

5. Conclusion and future work

In this paper, we have proposed a better approach for ontology integration. Several different ontologies can be merged into one from the same or different domain. From particular domain we can integrate ontologies one by one, it is essential to match similarities between two ontology. Here we have used ontology clustering using ontology matching tool and merge knowledge shared by different ontologies using ontology integration. For clustering, as a similarity measure, we have used global similarity measure Jaccard similarity index. Result describe how ontology clustering is performed and finally relevant similar ontologies are integrated into merged knowledge.

From this result, we could show that there is a reduction in average matching and merge time of ontology per pair using our new approach of matching and merging multiple pair ontologies. We could also show search space and response time reduction in SPARQL query processing. We could improve average 59% in loading time during ontology matching process because of the automated script was written to select a pair of ontologies from corpus one by one. We could also improve average matching time up to 24% compare to AML tool. In case of GROM tool also we could improve average merging time up to 32%. Query processing time also we could improve, compare to individual ontology by 68%. Our proposed merging approach also improves compactness and coverage benchmark parameter by 20%.

In future work, we could expand this approach for more number of ontologies and also on multiple domains. We could also extend our work in future for SPARQL query performance measurement benchmark using more number of query empirically by increasing number of experimentation.

References

[1] P. Hitzler, M. Krotzsch, and S. Rudolph, Foundations of semantic web technologies, CRC Press. 2011.

[2] D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, "Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential", *The MIT Press, Cambridge, London, England,* 2005.

[3] T. Gruber, "What is an Ontology," *http://www-ksl.stanford.edu/kst/whatis-an-ontology.html*, 1993.

[4] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins, "What are ontologies, and why do we need them?", *IEEE Intelligent Systems and their applications*, Vol. 14, No. 1, pp. 20-26, 1999.

[5] J. Euzenat and P. Shvaiko, *Ontology matching*, Vol. 18, Springer, Heidelberg, 2007.

[6] P. Shvaiko and J. Euzenat, "Ontology matching: state of the art and future challenges", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 1, pp. 158-176, 2013.

[7] P. Shvaiko and J. Euzenat, "Ten challenges for ontology matching," In: *Proc. of International Conferences on the Move to Meaningful Internet Systems*, pp.1164-1182, 2008.

[8] A. Zimmermann, M. Krotzsch, J. Euzenat, and P. Hitzler, "Formalizing Ontology Alignment and its Operations with Category Theory", In: *Proc. of the 4th International conference on Formal ontology in information systems*, 2006.

[9] J. Zhu, "A survey on ontology mapping," Elsevier publication, *Physics Procedia*, 242012. pp. 1857-1862

[10] L. Otero-Cerdeira, F. J. Rodríguez-Martínez, and A. Gómez-Rodríguez, "Ontology matching: a literature review", *Expert System Application*, Vol. 42, No. 2, pp. 949-971, 2015.

[11] A. Essayeh and M. Abed, "Towards ontology matching based system through terminological, structural and semantic level", *Procedia Computer Science*, Vol. 60, pp. 403-412, 2015.

[12] G. Mecca, G. Rull, D. Santoro, and E. Teniente, "Ontology-based mappings", *Data & Knowledge Engineering,* Vol. 98, pp.8-29, 2015.

[13] L. Otero-Cerdeira, F. J. Rodrıguez-Martınez, and A. Gómez-Rodrıguez, "Ontology Matching: Current trends among practitioners", *Expert Systems with Applications: An International Journal archive*, Vol. 42, No. 2, pp. 949-971, 2015.

[14] H. S. Pinto, A. Gómez-Pérez, and J. P. Martins, “Some issues on ontology integration”, In: *Proc. of the International Joint Conference on Artificial Intelligence*, 1999.

[15] H. S. Pinto and J. P. Martins, “A methodology for ontology integration”, In: *Proc. of the 1st international conference on Knowledge capture*, pp. 131-138, 2001.

[16] C. Ochs, Y. Perl, J. Geller, S. Arabandi, T. Tudorache, and M. A. Musen, “An Empirical Analysis of Ontology Reuse in BioPortal”, *Journal of Biomedical Informatics*, Vol. 71, pp.165-177, 2017.

[17] S. S. Choi, S. H. Cha, and C. C. Tappert, "A Survey of Binary Similarity and Distance Measures", *Journal of Systemics, Cybernetics and Informatics*, Vol. 8, No. 1, pp. 43-48, 2010.

[18] M. J. Lesot, M. Rifqi, and H. Benhadda, "Similarity measures for binary and numerical data: a survey", *International J. Knowledge Engineering and Soft Data Paradigms*, Vol. 1, No. 1, pp. 63-84, 2009.

[19] D. Faria, C. Pesquita, E. Santos, M. Palmonari, I.F.Cruz, and F. M. Couto, “Agreement maker light results for OAEI 2013”, In: *Proc. of the 8th International Conference on Ontology Matching*, pp. 101-108, 2013.

[20] OAEI : http://oaei.ontolgotymatching.org/2017

[21] D. Faria, C. Pesquita, E. Santos, M. Palmonari, I. F. Cruz, and F. M. Couto, “The agreementmakerlight ontology matching system”, In: *Proc. of the International Conferences on the Move to Meaningful Internet Systems*, pp. 527-541, 2013.

[22] Agreement Maker Light source https://github.com/AgreementMakerLight/

[23] M. Mahfoudh, L. Thiry, G. Forestier, and M. Hassenforder, “Algebraic graph transformations for merging ontologies", In: *Proc. of the International Conference on Model and Data Engineering*, pp.154-168, 2014.

[24] Y. Li, Y. Yu, and J. Heflin, "A Multi-ontology synthetic benchmark for the semantic web”, *IWEST@ ISWC*, 2010.

[25] S. Raunich and E. Rahm, “Towards a benchmark for ontology merging”, In: *Proc. of OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*, pp. 124-133, 2012.

[26] A. Solimando, E. Jiménez-Ruiz, and C. Pinkel, “Evaluating ontology alignment systems in query answering tasks”, In: *Proc. of the 2014 International Conference on Posters Demonstrations*, 2014.

[27] G. Guerrini, M. Mesiti, and I. Sanz, “An overview of similarity measures for clustering XML documents*”, Web Data Management Practices: Emerging Techniques and Technologies*, Idea Group Publishing, Hershey, London, Melbourne, Singapore, pp 56-78, 2007.

[28] S. S. Ahmed, M. Malki, and S. M. Benslimane, “Ontology Partitioning: Clustering Based Approach”, *International Journal of Information Technology and Computer Science*, Vol. 7, No. 6, pp. 1-11, 2015.

[29] E. Daskalaki, G. Flouris, I. Fundulaki, and T. Saveta, “Instance matching benchmarks in the era of linked data”, *Web Semantics: Science, Services, and Agents on the World Wide Web*, Vol. 39, pp.1–14, 2016.

[30] F. Rana and M. Shamsfard, "Symbiosis of evolutionary and combinatorial ontology mapping approaches", *Information Sciences*, Vol. 342, pp. 53-80, 2016.

[31] X. Xingsi and J. Liu, “Collaborative ontology matching based on a compact, interactive evolutionary algorithm", *Knowledge-Based Systems*, Vol. 137, pp.94-103, 2017.

[32] S. Cerón-Figueroa, I. López-Yáñez, W. Alhalabi, O. Camacho-Nieto, Y. Villuendas-Rey, M. Aldape-Pérez, and C. Yáñez-Márquez, “Instance-based ontology matching for e-learning material using an associative pattern classifier”, *Computers in Human Behavior*, Vol. 69, pp. 218-225, 2017.

[33] K. Saruladha and S. Ranjini, "COGOM: Cognitive Theory-Based Ontology Matching System", *Procedia Computer Science*, Vol. 85, pp. 301-308, 2016.

[34] M. Gulić, B. Vrdoljak, and M. Banek, “CroMatcher: An ontology matching system based on automated weighted aggregation and iterative final alignment”, *Web Semantics: Science, Services, and Agents on the World Wide Web*, Vol. 41, pp.50-71, 2016.

[35] E. Jiménez-Ruiz and B. C. Grau, "Logmap: Logic-based and scalable ontology matching", *International Semantic Web Conference, Lecture Notes in Computer Science*, 2011.

[36] P. Hitzler, M. Krotzsch, M. Ehrig, and Y. Sure, “What is ontology merging?”, *American Association for Artificial Intelligence*, 2005.

[37] N. F. Noy and M. A. Musen, “Algorithm and tool for automated ontology merging and alignment”, In: *Proc. of the 17th National Conference on Artificial Intelligence*, pp. 450-455, 2000.

[38] V. Nováček and P. Smrž, "Empirical merging of the ontologies-a proposal of universal uncertainty representation framework", In: *Proc. of the 3rd Annual European Semantic Web Conference*, pp. 65-79, 2006.

[39] M. Fareh, O. Boussaid, R. Chalal, M. Mezzi, and K. Nadji, "Merging ontology by Semantic enrichment and combining similarity measures", *Metadata, Semantics and Ontologies*, Vol. 8, No. 1, pp. 65-74. 2013.

[40] M. Mariem, G. Forestier, L. Thiry, and M. Hassenforder, "Consistent ontologies evolution using graph grammars", In: *Proc. of the International Conference on Knowledge Science, Engineering and Management*, pp. 64-75, 2013.

[41] F. Gaihua, "FCA based ontology development for data integration", *Information Processing & Management,* Vol. 52, No. 5, pp. 765-782, 2016.

[42] M. Maree and M. Belkhatir, "Addressing semantic heterogeneity through multiple knowledge bases assisted merging of domain-specific ontologies", *Knowledge-Based Systems,* Vol. 73, pp. 199-211, 2015.

[43] F. Muhammad, N. Moalla, and A. Bouras, "Towards ensuring satisfiability of merged ontology", *Procedia Computer Science,* Vol. 4, pp. 2216-2225, 2011.

[44] S. Raunich and E. Rahm, "Target-driven merging of taxonomies with atom", *Information Systems,* Vol. 42, pp.1-14, 2014.

[45] E. Daskalaki, G. Flouris, I. Fundulaki, and T. Saveta, "Instance matching benchmarks in the era of linked data.", *Web Semantics: Science, Services, and Agents on the World Wide Web*, Vol. 39, pp.1–14, 2016.

[46] F. Duchateau and Z. Bellahsene, "Measuring the Quality of an Integrated Schema", *Conceptual Modelling – ER 2010. Lecture Notes in Computer Science*, Vol. 6412, 2010.

[47] M. Mahfoudh, G. Forestier, and M. Hassenforder, "A benchmark for ontologies merging assessment", In: *Proc. of the International Conference on Knowledge Science, Engineering and Management,* 2016.

[48] E. Thiéblin, F. Amarger, O. Haemmerlé, N. Hernandez, and C. T. D. Santos, "Rewriting SELECT SPARQL queries from 1: n complex correspondences", In: *Proc. of the Ontology Matching Workshop 13th International Semantic Web Conference*, pp. 49-60, 2016.

[49] E. Thiéblin, F. Amarger, N. Hernandez, C. Roussey, and T. C. D. Santos, "Cross-Querying LOD Datasets Using Complex Alignments: An Application to Agronomic Taxa", *Metadata and Semantic Research, MTSR 2017, Communications in Computer and Information Science*, Vol. 755, pp. 25-37, 2017.

[50] K. X. Sampaio de Souza and J. Davis, "Using an Aligned Ontology to Process User Queries", *Lecture Notes in Computer Science*, Vol. 3192, 2004.

[51] The OBO Foundry *http://obofoundry.org/*

[52] NCBO BioPortal *http://bioportal.bioontology.org/*

[53] AgroPortal- *http://agroportal.lirmm.fr/*