Towards Social Collaborative Editing of Distributed Linked Data

Hafed Zarzour,[1] Mahmoud Al-Ayyoub,[2] and Yaser Jararweh[2]
[1]University of Souk Ahras, 41000, Souk Ahras, Algeria
[2]Jordan University of Science and Technology, Irbid, Jordan

***Abstract*— Recently, an increased number of organizations and institutions publish and expose the contents of their data as Linked Data where it becomes possible to share, reuse and link data over geographically-dispersed datasets. However, the existing principles of Linked Data do not suffice when collaboratively editing the same distributed Linked Data stores, more precisely, executing concurrent modifications in a social collaborative context. This paper adopts the optimistic replication purpose to develop a new approach that allows distant users to edit the same distributed Linked Data concurrently without using a central synchronization mechanism. The experimental results on real Linked Data stores demonstrate the effectiveness of the proposed approach in terms of execution time.**

Keywords*— collaborative editing; optimistic replication; semantic web; Linked Data; distributed system.

I. INTRODUCTION

In last years, the Linked Data paradigm has been adopted by multiple organizations [1] to support and help in exposing the resources; not only publishing them online but also utilizing this paradigm for enhancing their contents with relevant external data. As a result, several distributed datasets have been interconnected to construct what we call a Web of Data [2] [3] that consists of billions of RDF triples provided by different institutions. Hence, the idea of Linked Data is defined as a collection of best practices that helps at exposing and creating links between structured data via the Web in such a way that they are processable by machines and their meanings are explicitly defined. These best practices have been further reported in [4] by using four principles as follows: (i) utilization of URIs for handling data, (ii) the HTTP-URIs can be dereferenced, (iii) when data are dereferenced, they return useful meanings; RDF data model and the query language SPARQL are used as standards and (iv) by linking additional URLS, more data can be discovered.

Although Linked Data has proven to be useful in various domains such as educational technologies [5] [6], its principles do not suffice when collaboratively editing the same distributed Linked Data stores, more precisely, by executing concurrent modifications in a social collaborative context. Social collaborative editing can be defined as a situation where more than one user work together on the same replicas of data and interact by sharing operations in order to achieve a common goal. Indeed, it is necessary to adopt en efficient method for replication and consistency management [7] [8] [9]. Figure 1 shows the divergence issues of collaborative Linked Data editing after executing concurrent updates. In other words, when two users in two peers execute the same sequence of operations in different orders, they obtain divergent results. This is due to the noncommutativity of concurrent insert and delete operations executed on the same Linked Data store.

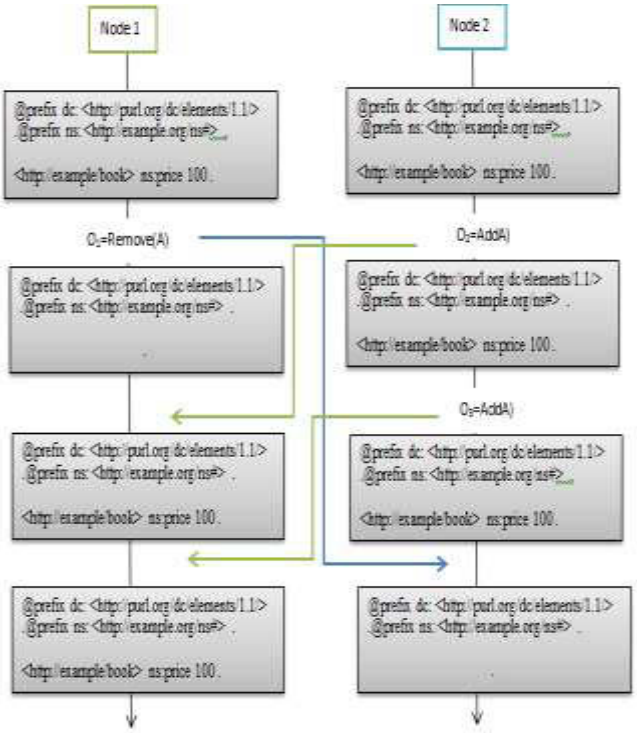

Fig.1. Divergence issues of collaborative Linked Data editing after executing concurrent updates

In distributed systems literature, there are two main classes of replication algorithms [10]: the first one, called pessimistic class, aims to guarantee strong consistency in which any access to replicated data is only ensured after the system blocks the modifications from all other sites. However, the pessimistic replication-based approaches are not well suited for all distributed systems since they do not scale when there are huge number of sites generating an important number of updates [11] [12]. Additionally, it is typically involved with performance reduction when blocking updates [13]. In contrast to the class of pessimistic replication, the second one, called optimistic class, uses weak consistency by allowing access to replicas without a priori synchronization with other sites. The main idea behind it is to focus on eventual consistency which

means that a replica is ensured to converge without blocking any of its updates.

In this paper, we focus on optimistic replication. We develop a new approach, called LD-Set, that allows distant users to edit the same distributed Linked Data concurrently without using a central synchronization mechanism. Our approach was implemented and its performance was assessed in terms of time of execution with real Linked Data stores.

The rest of this paper is structured as follows: in section 2 we review background and related work. In Section 3, we introduce our approach. In Section 4, we describe the effectiveness of our approach through experimental evaluations. Finally, we conclude the paper and provide future research directions.

II. Background and Related Work

With the popularity of the technologies related to Semantic Web, many researchers have proposed new approaches for distributed systems that support the sharing and synchronization of multiple replicas in geographically-dispersed datasets. For instance, RDFGrowth [14] is based on the idea of semantic information to share platforms. The peers in RDFGrowth are allowed only to perform updating or reading operations. To solve the consistency issue, it uses an algorithm of merge. However, RDFGrowth does not take into consideration the collaborative aspect as it simply shares data. RDFSync [15] is another example of RDF synchronization. The data model consists of RDF-graphs in which each one of them is represented as a triple-subsets. For synchronizing these triple-subsets, a function is called in order to cumulate the difference between the target and the source. Unfortunately, RDFSync ignores the effect produced when several operations are performed at the same time on the same replicas.

In last years, a new optimistic class has appeared called Commutative Replicated Data Type (CRDT) [16] [17]. This class is recognized by its simplicity in ensuring eventual consistency for distributed systems without a central synchronization mechanism. CRDT suggests that all concurrent updates can easily commute [17]. Many previous works have been focused on CRDT keys for supporting collaborative editing of text documents such as [17] [18]. In the same way, CRDT has been also designed for maintaining consistency in distributed semantic stores. B-Set [7], C-Set [19] and SU-Set [20] are typical examples that define CRDT for the set structure. However, none of them describe how to use CRDT in the context of Linked Data.

RDF-graph is considered as one of technologies used in Linked Data context to help in storing linked data as well as semantically enriching the interconnected databases. The triples in RDF-graph are expressed by connections between nodes and arcs; For example, an RDF-graph describing the Mona Lisa in terms of triples is expressed with seven statements as follows. This example is extracted from [22] (see Figure 2).

The RDF-graph has different ways to be represented. One of the most well-known is the Turtle format. The RDF-graph introduced in Figure 3 was transformed in Turtle as shown in Figure 4.

1. <http://example.org/bob#me>
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://xmlns.com/foaf/0.1/Person>.
2. <http://example.org/bob#me>
<http://xmlns.com/foaf/0.1/knows>
<http://example.org/alice#me>.
3.<http://example.org/bob#me>
<http://schema.org/birthDate> "1990-07-04"^^
<http://www.w3.org/2001/XMLSchema#date>.
4. <http://example.org/bob#me>
<http://xmlns.com/foaf/0.1/topic_interest>
<http://www.wikidata.org/entity/Q12418>.
5.<http://www.wikidata.org/entity/Q12418>
<http://purl.org/dc/terms/title> "Mona Lisa".
6.<http://www.wikidata.org/entity/Q12418>
<http://purl.org/dc/terms/creator>
<http://dbpedia.org/resource/Leonardo_da_Vinci>.
7.<http://data.europeana.eu/item/04802/243FA8618938F411
7025F17A8B813C5F9AA4D619>
<http://purl.org/dc/terms/subject>
<http://www.wikidata.org/entity/Q12418>.

Fig. 2. An example of an RDF-graph in triples [22]

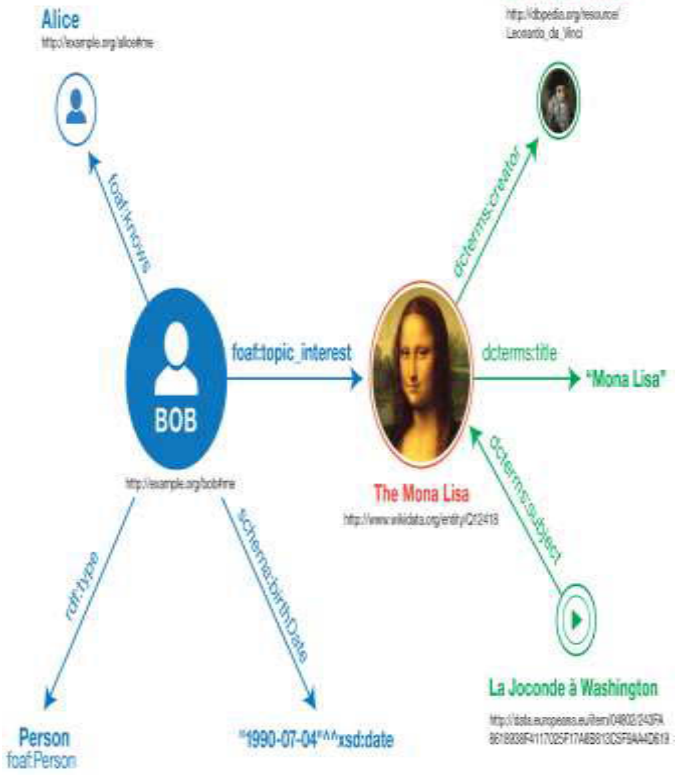

Fig. 3. An RDF-graph example [22]

III. Proposed Approach

Our solution adopts an optimistic replication purpose [10], where the system comprises a set of distributed and interconnected nodes in which all nodes share a single replica of the data all along for higher availability and performance. The crucial idea behind this is that there is a method enabling multiple nodes to update the shared replica content in concurrent way and discover conflicts and resolve them in order to obtain an identical result at the end of collaboration session.

A. Key concepts

In this section, the main concepts used are introduced for developing the proposed collaborative editing of Linked Data.

Social user. A social user is a person who owns a local copy of shared date and has the full access privilege to his/her copy. In other words, he has the ability to generate a sequence of local operations, broadcast them, and receive the remote operations that are sent by other peers. Each social user is identified by unique ID and is associated with one node of the Network.

Collaborative group. A collaborative group is comprised primarily of social users having the same directed relationship. The direct relationship refers to the relationship between two or more social users sharing the same interests and engaging in the same task to achieve the same goal. Moreover, this relationship is a useful propriety since it is able to assist social users in updating and managing their own replicas. The size of a collaborative group is the number of participants it includes. Furthermore, the members of the same collaborative group are connected through the Internet, and communicate using a broadcasting mechanism which ensures that all operations are firstly executed on local node then distributed to all other nodes of the collaborative group. Thus, the messages are disseminated between group peers are at operation level; this is called intra-group dissimilation.

Social network. In our context, a social network consists of a social organization composed of a collection of collaborative groups, which are linked by a specific common goal. To reduce the network traffic and improve network performance, we introduce the notion of patch in the inter-groups dissimilation process within the social network as opposed to intra-group dissimilation. A patch includes a set of operations (at least one operation) created by the same collaborative group. Therefore, the operations made in each collaborative group are placed in patches and are next sent to all other groups to be integrated.

Store of Linked Data. This store is based on RDF statements as RDF is the standard model used for describing and publishing Linked Data via the Web [21]. Such store can provide a data model using generic graph that stores and creates data interlinking to describe digital resources over the Web. More precisely, a definition of the RDF statement is viewed as a triples-set in which each single triple is assigned a unique URI. In addition, each triple consists of three elements and is written as <Sub, Pred, Obj>, where Sub is a subject, Pred is a predicate, Obj is an object. These elements correspond to source, target, and label, respectively. Accordingly, a collection of RDF triples forms what is called "RDF-graph".

To address the problem occurring when executing concurrent operations are performed (whether at the level of collaborative groups, or at the level of all social network), we define a new Linked Data-based CRDT. The main idea consists of guaranteeing eventual consistency, such that all replicas converge to an identical state, without neither complex synchronization nor total order. The best way of doing this is simply by ensuring the commutativity of any couple of operations. To do this, we define a Linked Data store as repository used for storing RDF triples with additional triple associated to each existing triple helping in supporting the commutativity of the concurrent operation. Formally, a Linked Data store S is a pair <X,A> where X is a triple and A is an additional triple. The additional triple is defined as <S,Instance,NB>, where the subject is S, the predicate is Instance and the occurrence of instances is expressed by NB. This corresponds to the insertion operations run for X, i.e., NB is used to give information about how many times X has been added. A sample of the defined structure for Linked Data store written in Turtle description is shown in Figure 5 in which the triple <book, price, 100> has been inserted only one time.

```
BASE   <http://example.org/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX schema: <http://schema.org/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX wd: <http://www.wikidata.org/entity/>

<bob#me>
    a foaf:Person ;
    foaf:knows <alice#me> ;
    schema:birthDate "1990-07-04"^^xsd:date ;
    foaf:topic_interest wd:Q12418 .

wd:Q12418
    dcterms:title "Mona Lisa" ;
    dcterms:creator <http://dbpedia.org/resource/Leonardo_da_Vinci> .

<http://data.europeana.eu/item/04802/243FA8618938F4117025F17A8B813C5F9AA4D6
19>
    dcterms:subject wd:Q12418 .
```

Fig. 4. An RDF-graph written in Turtle format

```
# Default graph
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ns: <http://example.org/ns#> .
@prefix in: <http://example.org/in#> .

<http://example/book> ns:price 100 .
<http://example/instance> in:NB 1 .
```

Fig. 5. An example of the LD-Set data structure

B. *Editing operations*

As any collaborative editing systems, our approach uses two main operations to update the Linked Data stores by performing adding and deleting actions.

The Add (X) adds a triple, given inline in the request, into the Linked Data store.

The Remove (X, n) removes a triple, given inline in the request, from the Linked Data store n times if it already exists.

These operations are invoked by social users and executed as SPARQL/UPDATE operation on RDF stores.

C. *Consistency specification*

In this section we give the specification of the consistency mechanism that enables us to resolve the problem of divergence between replicas sharing the same Linked Data and executing concurrent modifications in different order. The LD-Set is closer to the usual set semantics and does not have counter-intuitive behaviors. The main idea is to associate with each triple of Linked Data unique counter as additional triple which is initialized to 0 and incremented when the same triple is reinserted to the Linked Data store. When removing a triple, an interesting parameter must be declared to determine how many times we need to remove the inserted triple. Figure 4 presents the specification related to the mechanism of consistency of our solution. Payload is the data structure defined above. Lookup and Update are two methods that use the payload to give a result for an argument and perform adding and removing operations, respectively. Figure 6 details this specification. Add(t) inserts a triple at source and then broadcasts it to all other nodes, which add the received element into their private payloads. In this way, if the triple already exists, its counter is only incremented; otherwise, it is inserted with 0 as initial value. In the same way, Remove(t,n) deletes n times the triple t from the local replica and sends the corresponding operation to all other nodes which realizes the same effects in their payload. The manner in which the LD-Set is defined, supports the commutativity between any generated operations, thus, LD-Set is compliant with CRDT. Likewise, according to [16], it guarantees eventual consistency in any case.

```
1  payload set S
2  initial S=∅

3  query lookup (triple t) : boolean b
4        let b =(∃ i): (<t,i>)∈S)

5  update add(t)
6        if ( ∃ j: (t,j)∈S) then
7          let j:=j+1
8          S=S\{(t,j)} ∪ {(t,j:)}
9        else
10         let j =1
11         J=S ∪ {(t,j)}

12 update remove(t,n)
13       pre lookup(t)
14       if ( ∃ j: (t, j) ∈S) then
15        if ( n≤j) then
16          let j:=j-n
17          S=S\{(t,j)} ∪ {(t,j:)}
18        else
19          S=S\{(t,j)}
```

Fig. 6. Specification of LD-Set

D. *Steps of social collaborative editing for Linked Data stores*

To collaboratively edit the Linked Data stores in social context using our method, every node of the social network hosts a replica of all Linked Data contents and autonomously provides a flexible support for updating, reusing and sharing resources over the Web. Hence, this offers an efficient mechanism to take into account the massive collaboration without any complex conflict managing or data availability issues. The following steps illustrates the life cycle when updating Linked Data stores in the social networks.

- Step 1: Generating of the local operation by a social user on one node in a collaborative group.
- Step 2: Broadcasting the corresponding operation to other nodes of the same collaborative group.
- Step 3: Retrieving the remote operation.
- Step 4: Executing the retrieved operation on the remote site as a local operation.
- Step 5: Creating the patch from all executed operations at the level of collaborative group.
- Step 6: Delivering the corresponding patch to all other collaborative groups of the social network.
- Step 7: Receiving the sent patch.
- Step 8: Extracting operations from the received patch and running them.

When LD-Set is used, the eventual consistency is maintained between both nodes, as illustrated in Figure 7, contrary to that shown in Figure 1.

IV. Evaluation

To evaluate the effectiveness of DL-Set, the execution time is measured by implementing the specification, presented in Figure 7 on top of the ARQ. ARQ is a query engine for Jena that implements the SPARQL/Update core according to the W3C standards. In the present study, three different sites were considered where each site performs a random number of patches including a set of operations as follows:

Site 1: Patch1= [Add/Add/Add];
Site 2: Patch2= [Add/Add/Remove];
Site 3: Patch3= [Add/Remove/Add].

In our evaluation, we mainly focus on a dataset consisting of triples extracted from the Wikipedia by using the DBpedia mechanism. DBpedia [23] is a project that aims at extracting data from the Wikipedia then making them structured and available online as Linked Data.

Figure 8 shows a graph of how the execution time of three sites changes as the number of updates grows following the different patches. As shown, the first patch presents considerably improved performance all along the editing session compared with the second and third patches. This is because there is no conflict between the insert operations in the case of distributed Linked Data store. In other words, the execution of the sequence of insertions does not require any particular treatment to achieve the convergence since the same triple inserted by several operations can be only added one time. In the same way, the second patch outperforms the third patch at all update levels. This is due to the fact that both insert operations in the second path are executed as one, while those in the third path are run two time since they are separate by a delete operation. In summary, the existence and the position of delete operation over the insertion have a direct effect on the performance of our solution.

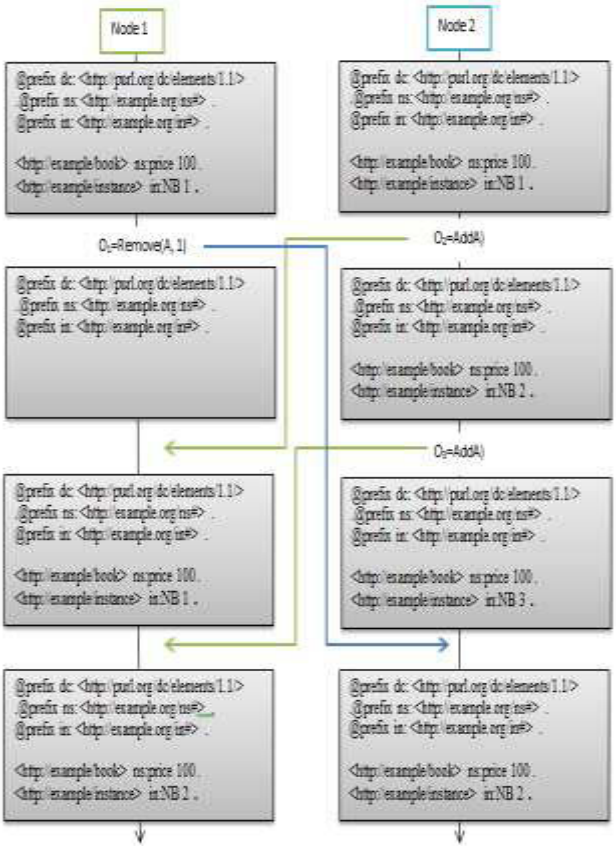

Fig. 7. Convergence after integrating LD-Set

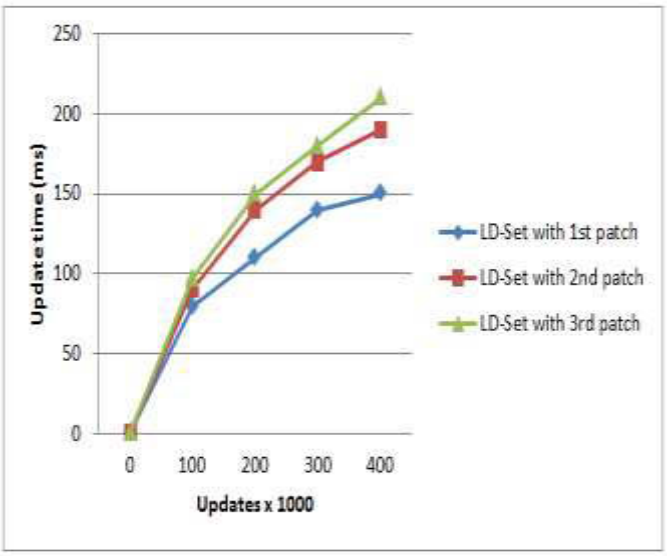

Fig. 8. The execution time for LD-Set with the three patches.

V. CONCLUSION

In this work, we have developed a solution called LD-Set that focuses on an optimistic replication mechanism. It aims to support the social collaborative editing of distributed Linked Data within a virtual community of users from different sites, while maintaining eventual consistency in order to enable concurrent updating. For implementing our approach, we have reused some of the SPARQL cores and developed a new one that takes into account the concurrency aspects when a set of users update the same replicas of Linked Data in a concurrent manner. From the experiment results, we can conclude that the existence of delete operations has an influence on the effectiveness of LD-Set in terms of execution time. This fact is produced only if consisting of the same triple.

Our solution opens several tasks for interesting future work. First, we plan to explore the adoption of the LD-Set to semantically enrich and interlink educational resources. Secondly, we plan also to conduct further evaluations with scalable datasets extracted from DBpedia [23] such as a geographic coordinates and articles categories in different languages.

REFERENCES

[1] Bizer, C., Heath, T., Berners-Lee, T.: Linked Data-the story so far. Semantic Services, Interoperability and Web Applications: Emerging Concepts, p. 205-227 (2009)

[2] Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S.: DBpedia-A crystallization point for the Web of Data. Web Semantics: science, services and agents on the world wide web, vol. 7, no 3, p. 154-165 (2009)

[3] Ristoski, P., Bizer, C., & Paulheim, H.: Mining the web of Linked Data with rapidminer. Web Semantics: Science, Services and Agents on the World Wide Web, vol. 35, p. 142-151 (2015)

[4] Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Sheets, D.: Tabulator: Exploring and analyzing Linked Data on the semantic web. In : Proceedings of the 3rd international semantic web user interaction workshop (2006)

[5] Zarzour, H., Sellami, M. : A Linked Data-based collaborative annotation system for increasing learning achievements. Educational Technology Research and Development. 65(2), 381-397 (2017).

[6] Zarzour, H., Sellami, M. : An investigation into whether learning performance can be improved by CAALDT. Innovations in Education and Teaching International. p. 1-8 (2017) Doi : 10.1080/14703297.2017.1286997

[7] Zarzour, H., Sellami, M. : B-Set: A synchronization method for distributed semantic stores. In Complex Systems (ICCS), 2012 International Conference on. p. 1-6 (2012).

[8] Zarzour, H., Sellami, M. : Using commutative replicated data type for collaborative video annotation. In Multimedia Computing and Systems (ICMCS), 2014 International Conference on p. 523-529 (2014).

[9] Zarzour, H., Sellami, M. : Achieving consistency in collaborative image annotation systems. In Information and Communication Systems (ICICS), 2014 5th International Conference on. p. 1-7 (2014).

[10] Saito, Y., Shapiro, M. : Optimistic replication. ACM Computing Surveys (CSUR), vol. 37, no 1, p. 42-81 (2005)

[11] Yu, H., Vahdat, A. : Minimal replication cost for availability. In : Proceedings of the twenty-first annual symposium on Principles of distributed computing. ACM, p. 98-107 (2002)

[12] Li, P., Gao, D., Reiter, M. K. : Replica placement for availability in the worst case. In : Distributed Computing Systems (ICDCS), 2015 IEEE 35th International Conference on. IEEE, p. 599-608 (2015)

[13] Lipskoch, K., Theel, O. : Relaxing data consistency along different dimensions for increasing operation availabilities. International Journal of Parallel, Emergent and Distributed Systems, vol. 30, no 3, p. 233-261 (2015)

[14] Tummarello, G., Morbidoni, C., Petersson, J., Puliti, P., Piazza, F. : RDFGrowth, a P2P annotation exchange algorithm for scalable Semantic Web applications. P2PKM, vol. 108 (2004).

[15] Quilitz, B., Leser, U. : Querying distributed RDF data sources with SPARQL. Springer Berlin Heidelberg, (2008).

[16] Preguica, N., Marques, J. M., Shapiro, M., Letia, M. : A commutative replicated data type for cooperative editing. In : Distributed Computing Systems, 2009. ICDCS'09. 29th IEEE International Conference on. IEEE, p. 395-403 (2009).

[17] Shapiro, M., Preguiça, N., Baquero, C., & Zawirski, M. : Conflict-free replicated data types. In : Stabilization, Safety, and Security of Distributed Systems. Springer Berlin Heidelberg, p. 386-400 (2011).

[18] Weiss, S., Urso, P., & Molli, P. : Logoot-undo: Distributed collaborative editing system on p2p networks. Parallel and Distributed Systems, IEEE Transactions on, vol. 21, no 8, p. 1162-1174 (2010).

[19] Aslan, K., Molli, P., Skaf-Molli, H., Weiss, S. : C-set: a commutative replicated data type for semantic stores. In RED: Fourth International Workshop on REsource Discovery (2011)

[20] Ibáñez, L. D., Skaf-Molli, H., Molli, P., Corby, O. : Synchronizing semantic stores with commutative replicated data types. In : Proceedings of the 21st international conference companion on World Wide Web. ACM, p. 1091-1096 (2012)

[21] Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S.: Quality assessment for Linked Data: A survey. Semantic Web, vol. 7, no 1, p. 63-93 (2015)

[22] Manola, F., Miller, E., McBride, B.: RDF 1.1 Primer. W3C Working Group Note, vol. 25 (February, 2014)

[23] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Bizer, C. : DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. Semantic Web, vol. 6, no 2, p. 167-195 (2015)