# S-RDF: A New RDF Serialization Format for Better Storage Without Losing Human Readability

Irvin Dongo, Richard Chbeir

# S-RDF: A new RDF Serialization Format for better Storage without losing Human Readability

Irvin Dongo[1,2] and Richard Chbeir[3]

[1] Univ. Bordeaux, ESTIA, Bidart, France
[2] Universidad Católica San Pablo, Arequipa, Peru
i.dongoescalante@estia.fr
[3] Univ. Pau & Pays Adour, E2S/UPPA, LIUPPA, EA3000, Anglet, France
richard.chbeir@univ-pau.fr

**Abstract.** Nowadays, RDF data becomes more and more popular on the Web due to the advances of the Semantic Web and the Linked Open Data initiatives. Several works are focused on transforming relational databases to RDF by storing related data in N-Triple serialization format. However, these approaches do not take into account the existing normalization of their databases since N-Triple format allows data redundancy and does not control any normalization by itself. Moreover, the mostly used and recommended serialization formats, such as RDF/XML, Turtle, and HDT, have either high human-readability but waste storage capacity, or focus further on storage capacities while providing low human-readability. To overcome these limitations, we propose here a new serialization format, called *S-RDF*. By considering the structure (graph) and values of the RDF data separately, S-RDF reduces the duplicity of values by using unique identifiers. Results show an important improvement over the existing serialization formats in terms of storage (up to 71,66% w.r.t. N-Triples) and human readability.

**Keywords:** Serialization Format, Semantic Web, Data Representation, RDF

## 1 Introduction

For the Semantic Web, RDF is the *common format* to describe resources, which are abstractions of entities (documents, abstract concepts, persons, companies, etc.) of the real world. It was developed by Ora Lassila and Ralph Swick in 1998 [15]. RDF uses triples in the form of $\langle$`subject`, `predicate`, `object`$\rangle$ expressions, also named statements, to provide relationships among resources.

Currently, RDF data available on the Web is increasing rapidly due to the promotion of the Semantic Web and the Linked Open Data (LOD) initiatives [20]. Governments, organizations and research communities are part of the LOD initiatives, providing their data to have a more flexible data integration, increasing the data quality and providing new services [10]. Since RDF does not restrict how

data is converted, several RDF serializations are available in the literature [11]. For instance, RDF/XML is historically the first W3C standard which serializes the RDF graph ($\langle$`subject`, `predicate`, `object`$\rangle$) into XML. Other serializations, such as Turtle and N3, are also highly recommended [18]. In the literature, several works have been proposed to convert different datasets to RDF/OWL. The works in [12,14,17,23,26] propose to convert XML data into RDF using XPath expressions, XSD Schemas, DTD[4], etc. Other works provided in [3,4,9,13,19,21,22,24] address RDF conversion of relational database models to publish huge quantity of information and linked to the Web. However, current adopted serialization formats are mainly focusing on document-centric view to increase human readability, while requiring important storage space and bandwidth resources [11]. In essence, these formats do not control the redundancy of data by definition which also affects the conceptual model. The authors in [25] address the syntactic redundancy of the data by applying a normalization methodology. Other authors as in [11] propose a binary representation format called HDT, reducing the redundancy of data, but decreasing the human readability of the information.

To overcome these limitations, we propose here a new serialization format called *S-RDF*, which represents the RDF graph structure and the values separately for a better human readability. This serialization is available to manage medium-large datasets by reusing identifiers (keys) extracted from several ones. Moreover, the storage is reduced and some graph properties (e.g., degree centrality measure[5]) can be easily analyzed. We validated our serialization format through several experiments. Results show an improvement over the existing serialization formats in terms of storage (up to 71.66% with respect to N-Triples) and human readability.

The rest of this paper is organized as follows. In Section 2, we present a motivating scenario to illustrate better the needs. Section 3 surveys the related literature. Terminologies and definitions are presented in Section 4. Section 5 describes our serialization format. In Section 6, we present the experiments conducted to evaluate the compression rate and the human-readability. Finally, we present conclusions in Section 7.

## 2 Motivating Scenario

As mentioned previously, RDF data can be represented in different ways (serializations), i.e., stored in a file system through several formats. In order to illustrate the limitations of existing serialization formats, we consider a scenario in which the information of Listing 1 is shared on the Web. This listing shows four `Schools` entities: `S0991, S0992, S0993`, and `S0994`, which have information such as `rdf:type, ins:name, ins:postalCode,` and `ins:established`.

[4] Document Type Definition (DTD) defines the structure and the legal elements and attributes of an XML document.

[5] Centrality identifies the most related nodes within a graph, which have a high number of relations.

```
@Prefix ins:http://institutions.com/0.2/
ins:S0991;
ins:name "Lycee du Parc"^^xsd:string ;
ins:postalCode "64600"^^xsd:string ;
ins:established 1985-05-19^^xsd:date ;
rdf:type http://www.w3.org/2002/07/owl#Thing.

ins:S0992;
ins:name "Napoleon Business"^^xsd:string ;
ins:postalCode "64100"^^xsd:string ;
ins:established 1986-12-19^^xsd:date .
rdf:type http://www.w3.org/2002/07/owl#Thing.

ins:S0993;
ins:name "Ecole national de l'energi"^^xsd:string ;
ins:postalCode "64500"^^xsd:string ;
ins:established 1984-11-21^^xsd:date .
rdf:type http://www.w3.org/2002/07/owl#Thing.

ins:S0994;
ins:name "Grande Ville School"^^xsd:string ;
ins:postalCode "64600"^^xsd:string ;
ins:established 1977-08-22^^xsd:date .
rdf:type http://www.w3.org/2002/07/owl#Thing.
```

**Listing 1.** School Information

Table 1 shows the serialization formats defined by the W3C (RDF/XML, Turtle, N-Triple, and N3). These formats are document-centric view since their data can be read and understood by humans; however, for a data that generates a graph with a considerable depth (more than three), the readability is reduced. For instance, according to our motivating scenario, one can easily observe the properties of the entity `S0991` (`ins:name`, `ins:postalCode`, `ins:established`) and its respective values of the RDF/XML, Turtle , N-Triple and N3 serialization formats, since the depth of the generated graph is 2. If some blank nodes are added between the entity and the properties, the readability decreases by finding the properties in another part of the document, using the entity and blank nodes as references to search the values.

The RDFa, microdata and JSON-LD serialization formats are adopted as recommendation by the W3C. Table 2 shows and describes the three aforementioned formats. These formats are also document centric view as the previous ones; therefore, the same limitation is found. Moreover, since all serialization formats are document centric view, the storage is not taken into account by any of them. For small datasets, it is not a need, but for medium and large datasets, especially the ones obtained from relational databases, the storage represents a critical issue and has an impact on exchanging data.

In general, the first RDF serialization formats were proposed as document-centric view (RDF/XML, Turtle), since RDF data describes mainly Web Pages as resources (e.g, DBpedia from Wikipedia) and the number of properties to described them is limited (About: Eiffel Tower is describe by 156 triples); however, as the resources can be linked on the Web, the number of triples increases exponentially by considering datasets that use several resources. Therefore, a format able to describe a resource or a set of resources is needed considering the storage as a main requirement for medium-large datasets.

**Table 1.** Serialization formats defined by the W3C

| S. Format | Description | Example of Listing 1 |
|---|---|---|
| RDF/XML [18] | It is the first serialization format adopted by the W3C. This format serializes the RDF and XML files, where nodes and edges of the RDF document are represented using XML syntax. Their current media type is application/rdf+xml. | `<?xml version="1.0"?>`<br>`<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"`<br>`xmlns:dc="http://purl.org/dc/elements/1.1/"`<br>`xmlns:ins="http://institutions.com/0.2/">`<br>`<rdf:Description rdf:about="http://institutions.com/0.2/S0991">`<br>`<ins:name rdf:datatype="xsd:string">Lycee du Parc</ins:name>`<br>`<ins:postalCode rdf:datatype="xsd:string">64600</ins:postalCode>`<br>`<ins:established rdf:datatype="xsd:date">1985-05-19</ins:established>`<br>`<rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Thing" />`<br>`</rdf:Description>`<br>`...`<br>`</rdf:RDF>` |
| Turtle (Terse RDF Triple Language) [18] | It is a textual serialization format to encode RDF documents in a compact form and also readable for humans. Their current media type is application/x-turtle. | `@prefix ns0: <http://institutions.com/0.2/> .`<br>`@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .`<br><br>`<http://institutions.com/0.2/S0991>`<br>`ns0:name "Lycee du Parc"^^xsd:string ;`<br>`ns0:postalCode "64600"^^xsd:string ;`<br>`ns0:established "1985-05-19"^^xsd:date . ...` |
| N-Triple (Notation of Triples) [18] | It is simple serialization of RDF but not as compact as Turtle format. Their current media type is text/-plain. | `<http://institutions.com/0.2/S0991> <http://institutions.com/0.2/-`<br>`postalCode> "64600"^^<xsd:string> .`<br>`<http://institutions.com/0.2/S0991> <http://institutions.com/0.2/-`<br>`name> "Lycee du Parc"^^<xsd:string> .`<br>`<http://institutions.com/0.2/S0991> <http://institutions.com/0.2/-`<br>`established> "1985-05-19"^^<xsd:date> . ...` |
| N3 (Notation 3) [18] | It is an extension format of turtle language expressing a superset of RDF and has been designed with human readability in mind. Their current media type is text/rdf+n3. | `@prefix dc: <http://purl.org/dc/elements/1.1/> .`<br>`@prefix ins: <http://institutions.com/0.2/> .`<br>`@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .`<br>`@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .`<br>`@prefix xml: <http://www.w3.org/XML/1998/namespace> .`<br>`@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .`<br><br>`ins:S0991 ins:established "1985-05-19"^^<xsd:date> ;`<br>`ins:name "Lycee du Parc"^^<xsd:string> ;`<br>`ins:postalCode "64600"^^<xsd:string> . ...` |

By regarding the limitations of existing serialization formats, we have identified three main requirements according to the challenges and objectives of this work:

- A high-human readability for easy understanding of data;
- A high radio compression for minimizing the storage space and reducing exchanging delays; and
- A format oriented to describe medium-large datasets.

The following section describes and compares the related work by using the identified requirements.

## 3 Related Work

To the best of our knowledge, several serialization formats have been also proposed in the literature other than the ones adopted or recommended by the W3C. The authors in [8] present a binary RDF representation for large datasets. They

**Table 2.** Serialization Formats recommended by the W3C

| S. Format | Description | Example |
|---|---|---|
| RDFa (Resource Description Framework in Attributes) | It is a serialization format that adds structured data to HTML or XHTML documents by extending the attributes of elements. | `<div xmlns="http://www.w3.org/1999/xhtml"`<br>`  prefix="`<br>`  owl: http://www.w3.org/2002/07/owl#`<br>`  ns1: xsd:`<br>`  rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#`<br>`  ins: http://institutions.com/0.2/`<br>`  rdfs: http://www.w3.org/2000/01/rdf-schema#"`<br>`  >`<br>`  <div typeof="owl:Thing" about="http://institutions.com/0.2/S0991">`<br>`  <div property="ins:name" datatype="ns1:string" content=`<br>`  "Lycee du Parc"></div>`<br>`  <div property="ins:postalCode" datatype="ns1:string" content=`<br>`  "64600"></div>`<br>`  <div property="ins:established" datatype="ns1:date" content=`<br>`  "1985-05-19"></div>`<br>`  </div>  ...`<br>`</div>` |
| Microdata [1] | It is a serialization format that describe a simpler way of annotating HTML elements with machine-readable tags. | `<div>`<br>`  <div itemtype="http://www.w3.org/2002/07/owl#Thing"`<br>`  itemid="http://institutions.com/0.2/S0991" itemscope>`<br>`  <meta itemprop="http://institutions.com/0.2/postalCode"`<br>`  content="64600" />`<br>`  <meta itemprop="http://institutions.com/0.2/established"`<br>`  content="1985-05-19" />`<br>`  <meta itemprop="http://institutions.com/0.2/name"`<br>`  content="Lycee du Parc" />`<br>`  </div>  ...`<br>`</div>` |
| JSON-LD [16] | It is a concrete syntax format that extends the RDF data model to optionally allow JSON-LD to serialize Generalized RDF Datasets. | `{`<br>`  "@context": {`<br>`  "ins": "http://institutions.com/0.2/",`<br>`  "owl": "http://www.w3.org/2002/07/owl#",`<br>`  "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",`<br>`  "rdfs": "http://www.w3.org/2000/01/rdf-schema#",`<br>`  "xsd": "http://www.w3.org/2001/XMLSchema#"`<br>`  },`<br>`  "@id": "ins:S0991",`<br>`  "@type": "owl:Thing",`<br>`  "ins:established": {`<br>`  "@type": "xsd:date",`<br>`  "@value": "1985-05-19"`<br>`  },`<br>`  "ins:name": {`<br>`  "@type": "xsd:string",`<br>`  "@value": "Lycee du Parc"`<br>`  },  ...`<br>`}` |

represent the RDF graph in three logical components: (i) Header, (ii) Dictionary, and (iii) Triples. The size of the datasets is reduced, improving the data sharing and the querying and indexing performance. In [11], the authors improve their previous work up to 2 times for more structured datasets, and a significant improvement for semi-structured datasets as DBpedia. Other works, as in [5], have focused on compressed representation for RDF Querying. The authors highlight that the improvement is around 50% to 60% of the original HDT. This format is proposed for the use of GPU.

Table 3 shows our related work classification. RDF/XML, Turtle, N3 and JSON-LD focus on human readability since their formats can be easily read by humans. HDT, HDT++ and TripleID-C have been designed to improve the

**Table 3.** Related Work Classification

| Serialization Format | Human Readability | Storage (Compression) | Non-Redundancy | Large-Medium Dataset | Media Type |
|---|---|---|---|---|---|
| RDF/XML | Low+ | Medium | Low | Low | RDF+XML |
| Turtle | High | Medium | Medium | Low | - |
| N-Triple | Low | Low | Low | Low | - |
| N3 | High | Medium | Medium | Low | - |
| RDFa | Low | Low | Low | Low | HTML/XHTML |
| Microdata | Low | Low | Low | Low | HTML |
| JSON-LD | Medium | Medium | Medium | Low | JSON |
| HDT [8] | Zero | High | High | High | Binary |
| HDT++ [11] | Zero | High+ | High | High | Binary |
| TripleID-C [5] | Low | High | High | Medium | GPU |
| **RDF Sequence** | **High** | **Medium+** | **High** | **Medium+** | – |

storage, affecting the human readability. Note that none of the works satisfies all the defined requirements; thus, a new RDF serialization format is required.

Before describing our serialization format, the following section introduces some common terminologies and definitions in the context of RDF.

## 4 RDF Terminologies and Definitions

RDF commonly uses triples in the form of ⟨`subject`, `predicate`, `object`⟩ expressions/statements, to provide relationships among resources. The RDF triples can be composed of the following elements:

– An **IRI**, which is an extension of the Uniform Resource Identifier (URI) scheme to a much wider repertoire of characters from the Universal Character Set (Unicode/ISO 10646), including Chinese, Japanese, and Korean character sets [7].

– A **Blank Node**, representing a local identifier used in some concrete RDF syntaxes or RDF store implementations. A blank node can be associated with an identifier (`rdf:nodeID`) to be referenced in the local document, which is generated manually or automatically

– A **Literal Node**, representing values as strings, numbers, and dates. According to the definition in [6], it consists of two or three parts:

- A **lexical form**, being a Unicode string, which should be in Normal Form C[6] to assure that equivalent strings have a unique binary representation
- A datatype **IRI**, being an IRI identifying a **datatype** that determines how the lexical form maps to an object value
- A **non-empty language tag** as defined by "Tags for Identifying Langua-

[6] It is one of the four normalization forms, which consists on a Canonical Decomposition, followed by a Canonical Composition -http://www.unicode.org/reports/tr15/

ges"[2], if and only if the datatype IRI is http://www.w3.org/1999/02/22-rdf-syntax-ns#langString.

Table 4 shows the sets of RDF's elements that we use in our formal approach description.

**Table 4.** Description of sets

| Set | Description |
|---|---|
| $I$ | A set of IRIs is defined as: I = $\{i_1, i_2, ..., i_n\}$ \| $\forall i_i \in I$, $i_i$ is an IRI. |
| $L$ | A set of literal nodes is defined as: L= $\{l_1, l_2, ..., l_n\}$ \| $\forall l_i \in L$, $l_i$ is a literal node. |
| $BN$ | A set of blank nodes is defined as: $BN = \{bn_1, bn_2, ..., bn_n\}$ \| $\forall bn_i \in BN$, $bn_i$ is a Blank Node. |

After the definition of sets of RDF'elements, we formally describe a triple in Def 1.

**Definition 1. Triple (*t*)**: *A Triple, denoted as* ***t****, is defined as an atomic structure consisting of a 3-tuple with a Subject (****s****), a Predicate (****p****), and Object (****o****), denoted as* $t :< s, p, o >$*, where:*
- $s \in I \cup BN$ *represents the subject to be described;*
- *p is a predicate defined as an IRI in the form* ***namespace_prefix:predicate_name****, where namespace_prefix is a local identifier of the IRI, in which the predicate (predicate_name) is defined. The predicate (p) is also known as the* ***property*** *of the triple;*
- $o \in I \cup BN \cup L$ *describes the object.*

♦

From Listing 1, one can observe the following triples with different RDF resources, properties, and literals:
- $t_3$: `<genid:S0991,rdf:type, http://www.w3.org/2002/07/owl#Thing>`
- $t_4$: `<genid:S0991,ins:name,"Lycée de la Plage">`
- $t_5$: `<genid:S0991,ins:established,1985-05-19>`

In this study, we also consider two types of properties (predicates):
- **Entity Property (*ep*):** A predicate is an entity property when it is related to an IRI or a blank node. It is also known as Object property. For example, the property `eni:locates` is an entity property since it is related to a blank node.
- **Value Property (*vp*):** A predicate is a value property when it is related to a literal node. It is also known as Datatype property. For example, the property `ins:established` is a value property since it is related to a literal node.

An RDF document is defined as an encoding of a set of triples, using a predefined serialization format complying with an RDF W3C standards, such as RDF/XML, Turtle, N3, etc. Additionally, we use the term *entity*, formally described in Def.2, to identify an RDF resource (blank node and IRI).

**Definition 2. Entity ($e$)**: *An entity in an RDF document, denoted as e, is represented as an IRI or a blank node (e.g., School, Power Plant).* ♦

For example, from Listing 1, the triple `<S0991,ins:name,"Lycée de la Plage">` has the entity `S0991`.

In Def. 3, Def. 4, Def. 5, and Def. 6, we formally describe the respective sets of entities, entity properties, value properties, and literal values of an RDF document.

**Definition 3. Entity Set ($E$)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$, *the entities of each* $t_i$ *define the set of all entities, denoted as* $E = \bigcup_{i=1}^{n} t_i.s \cup t_i.o \iff t_i.o \in I \cup BN$, *where n is the number of triples.* ♦

The entity set according to Def. 3 of Listing 1 is: E={http://institutions.com-/0.2/S0991, http://institutions.com.com/0.2/S0992, http://institutions.com/0.2-/S0993, http://institutions.com/0.2/S0994, http://www.w3.org/2002/07/owl#-Thing}.

**Definition 4. Entity Properties ($EP$)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$, *the predicates of all* $t_i$ *that are entity properties, define the set of entity properties, denoted as:* $EP = \bigcup_{i=1}^{n} t_i.p \iff t_i.o \in I \cup BN$, *where n is the number of triples.* ♦

The entity properties from Listing 1 are: EP={rdf:type} or EP={http://www-.w3.org/1999/02/22-rdf-syntax-ns#type}.

**Definition 5. Value Properties ($VP$)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$, *the predicate of all* $t_i$ *that are value properties, define the set of value properties, denoted as:* $VP = \bigcup_{i=1}^{n} t_i.p \iff t_i.o \in L$, *where n is the number of triples.* ♦

According to Def. 5, the value properties obtained from the triples of Listing 1 are: VP={http://institutions.com/0.2/name, http://institutions.com/0.2/postal-Code, http://institutions.com/0.2/established}.

**Definition 6. Literal Values ($LV$)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$, *the literals of all* $t_i$ *define the set of literal values, denoted as:* $LV = \bigcup_{i=1}^{n} t_i.o \iff t_i.o \in L$, *where n is the number of triples.* ♦

According to Def. 6, the literal values from Listing 1 are: VA={"Lycée de la Plage", "64600", "1985-05-19", "Napoleon Business", "64100", "1986-12-19", "École National de l'energie", "64500", "1984-11-21", "Grande Ville School", "64200", "1977-08-22"}.

Table 5 summaries the sets of entities, entity and value properties, and literal values of an RDF document.

The following section presents and describes our new serialization format S-RDF.

**Table 5.** Description of sets of data of an RDF document

| Set | Description |
|---|---|
| $E$ | A set of entities is defined as: $E = \bigcup_{i=1}^{n} \{t_i.s\} \cup t_i.o \iff t_i.o \in I \cup BN.$ |
| $EP$ | A set of entity properties is defined as: $EP = \bigcup_{i=1}^{n} t_i.p \iff t_i.o \in I \cup BN.$ |
| $VP$ | A set of value properties is defined as: $VP = \bigcup_{i=1}^{n} t_i.p \iff t_i.o \in L.$ |
| $LV$ | A set of literal values is defined as: $LV = \bigcup_{i=1}^{n} t_i.o \iff t_i.o \in L.$ |

## 5 S-RDF: Our Proposal

Our proposal mainly relies on a three step process: (i) *Extraction of RDF elements*, where the input, an RDF document in any format, is analyzed in order to extract the set of entities (E), entity properties (EP), value properties (VP), and literal values (LV); (ii) *RDF Sequence-Value generation* where entities, properties, and literal values are represented by unique identifiers (e.g., primary keys); and (iii) *RDF Sequence-Structure generation* where relations among entities, which define the RDF graph structure, are expressed using the *Sequence-Value Representation*. Thus, our serialization format (*S-RDF*) consists in two parts: (i) Value Representation and (ii) Structure. Fig. 1 shows the framework of our proposal composed by three modules that materialize the three respective phases.

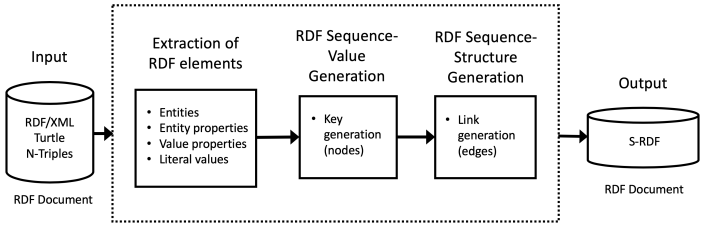


**Fig. 1.** Framework of our serialization format "S-RDF"

In Def. 7, we formally describe the Value Representation part of our RDF sequence, called RDF Sequence–Value. This representation associates to each entity, entity and value property, and literal value a unique identifier to be used in the structure representation of the sequence. We propose four different identifiers to easily recognize the type of data in the second part of our sequence. The entities are represented by numbers of the decimal numeral system (base 10), starting from 1. In the case of entity and value properties, both identifiers correspond to the hexavigesimal numeral system (base 26), with a domain of lowercase and uppercase alphabet letters, respectively. For the literal values, the identifiers belong to the decimal numeral system as the ones of entities, but a symbol "_" is added as a prefix. For instance, the $28^{th}$ element of the entities is represented as "28", "AB" for entity properties, 'ab" for value properties , while for literal values is "_28".

**Definition 7. RDF Sequence–Value (*S-RDF-V*)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$*, its RDF Sequence-Value is defined as a 4-tuple of:*

$S - RDF - V_T = <$
$Entities = \{\bigcup_{i=1}^{m} < pk_i, e_i, type_i >\},$
$Entity\_properties = \{\bigcup_{j=1}^{n} < pk_j, ep_j >\},$
$Value\_properties = \{\bigcup_{k=1}^{o} < pk_k, vp_k, datatype_k >\},$
$Literal\_values = \{\bigcup_{l=1}^{p} < \_pk_l, lv_l >\} >$

*where:*

- *Entities is a set of 3-tuples, where:*
  - $m \in Z^+$*, is the size of E.*
  - $pk_i \in Z^+$*, is a key that represents* $e_i$*.*
  - $e_i \in E$ *is an entity.*
  - $type_i \in \{1, 2\}$*, is the type of the entity* $e_i$ *(1=IRI, 2=blank node).*
- *Entity_properties is a set of 2-tuples, where:*
  - $n \in Z^+$*, is the size of EP.*
  - $pk_j \in \{A...Z\}$*, is a key that represents* $ep_j$*.*
  - $ep_j \in EP$ *is an entity property.*
- *Value_properties is a set of 3-tuples, where:*
  - $o \in Z^+$*, is the size of VP.*
  - $pk_k \in \{a...z\}$*, is a key that represents* $vp_k$*.*
  - $vp_k \in VP$ *is a value property.*
  - $datatype_k$ *is the datatype of the property.*
- *Literal_values is a set of 2-tuples, where:*
  - $p \in Z^+$*, is the size of LV.*
  - $\_pk_l$ *is a key that represents* $lv_l$ *and* $pk_l \in Z^+$*.*
  - $lv_l \in LV$ *is a literal value.* ♦

Tables 6-9 represent the Entities, Entity_properties, Value_properties, and Literal_values of Listing 1. The first element of the S-RDF-V is composed by the entities of Table 6. As only one relation among entities is shown in Listing 1, the second element (entity properties) of the 4-tuple is: {<A, http://www.w3-.org/1999/02/22-rdf-syntax-ns#type>}. The third and fourth elements are composed by the information in Table 8 and Table 9, respectively. The set of triples (T), obtained from Listing 1, has the following RDF Sequence–Value:

S-RDF-V(T) = <
**Entities** = {
$< 1, http://institutions.com/0.2/S0991, 1 >,$
$< 2, http://institutions.com/0.2/S0992, 1 >,$
$< 3, http://institutions.com/0.2/S0993, 1 >,$
$< 4, http://institutions.com/0.2/S0994, 1 >,$
$< 5, http://www.w3.org/2002/07/owl\#Thing, 1 >\},$
**Entity_properties** = {
$< A, http://www.w3.org/1999/02/22 - rdf - syntax - ns\#type >\},$
**Value_properties** = {

**Table 6.** Entities

| **Key** ($pk_i$) | **Entity** ($e_i$) | **Type** ($type_i$) |
|---|---|---|
| 1 | http://institutions.com/0.2/S0991 | 1 |
| 2 | http://institutions.com/0.2/S0992 | 1 |
| 3 | http://institutions.com/0.2/S0993 | 1 |
| 4 | http://institutions.com/0.2/S0994 | 1 |
| 5 | http://www.w3.org/2002/07/owl#Thing | 1 |

**Table 7.** Entity Properties

| **Key** ($pk_j$) | **Entity Property** ($ep_j$) |
|---|---|
| A | http://www.w3.org/1999/02/22-rdf-syntax-ns#type |

**Table 8.** Value Properties

| **Key** ($pk_k$) | **Value Property** ($vp_k$) | **Datatype** ($datatype_k$) |
|---|---|---|
| a | http://institutions.com/0.2/name | xsd:string |
| b | http://institutions.com/0.2/postalCode | xsd:string |
| c | http://institutions.com/0.2/established | xsd:date |

**Table 9.** Literal Values

| **Key** ($pk_l$) | **Value** ($lv_l$) |
|---|---|
| _1 | Lycée de la Plage |
| _2 | 64600 |
| _3 | 1985-05-19 |
| _4 | Napoleon Business |
| _5 | 64100 |
| _6 | 1986-12-19 |
| _7 | École National de l'energie |
| _8 | 64500 |
| _9 | 1984-11-21 |
| _10 | Grande Ville School |
| _11 | 64200 |
| _12 | 1977-08-22 |

$< a, http://institutions.com/0.2/name, xsd:string >,$
$< b, http://institutions.com/0.2/postalCode, xsd:string >,$
$< c, http://institutions.com/0.2/established, xsd:date >\},$
**Literal_values** $= \{$
$< \_1, "Lycée\ de\ la\ Plage" >, < \_2, 64600 >, < \_3, 1985-05-19 >,$
$< \_4, "Napoleon\ Business" >, < \_5, 64100 >, < \_6, 1986-12-19 >,$
$< \_7, "École\ National\ de\ l'energie" >, < \_8, 64500 >, < \_9, 1984-11-21 >,$
$< \_10, "Grande\ Ville\ School" >, < \_11, 64200 >, < \_12, 1977-08-22 >\}$
$>$

The S-RDF-V represents the entities, properties, and values of an RDF document, but a document also has information about the relations among entities and literal values (node-edge-node); thus, the second part of our serialization is dedicated to represent the RDF graph structure, called RDF Sequence–Structure. It consists of a 3-tuple, where the first element is composed of an entity; the second element has all entities, which are related to the first element, preceded by its respective entity property; and the last element is used to represent value properties and its respective literal values. The RDF Sequence–Structure is defined in Def. 8.

**Definition 8. RDF Sequence-Structure (*S-RDF-S*)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$*, its RDF Sequence-Structure is defined as a set of 3-tuples:*

$$\begin{aligned} \textit{S-RDF-S(T)} = \{\textstyle\bigcup_{i=1}^{n} < \\ & \textit{entity} = e_i.pk, \\ & \textit{entity\_property-entity} = \{ep_j.pk, e_k.pk\}, \\ & \textit{value\_property-value} = \{vp_l.pk, lv_m.pk\} >\}. \end{aligned}$$

*where:*

- *entity*
  - $e_i$ *is an entity.*
- *entity_property-entity*
  - $ep_j, e_k$ *represents* $t :< e_i, ep_j, e_k >$*, such that* $ep_j$ *is an EntityProperty where* $e_i$ *is the subject and* $e_k$ *the object.*
- *value_property-value*
  - $vp_i, lv_j$ *represents* $t :< e_i, vp_l, lv_m >$*, such that* $vp_l$ *is a PropertyValue where* $e_i$ *is the subject and* $lv_m$ *the object.* ♦

For example, the set of triples (T) obtained from Listing 1, has the following RDF Sequence–Structure:

S-RDF-S(T) ={

$< 1, \{(A, 5)\}, \{(a, \_1), (b, \_2), (c, \_3)\} >$,
$< 2, \{(A, 5)\}, \{(a, \_4), (b, \_5), (c, \_6)\} >$,
$< 3, \{(A, 5)\}, \{(a, \_7), (b, \_8), (c, \_9)\} >$,
$< 4, \{(A, 5)\}, \{(a, \_10), (b, \_11), (c, \_12)\} >$,
$< 5, \{\}, \{\} >\}$,

representing: entity "1" (`http://institutions.com/0.2/S0991` according to Table 6), has an entity property " A" (`http://www.w3.org/1999/02/22-rdf-syntax-ns#type` according to Table 7), related to the entity "5" (`http://www.w3.org/2002/07/owl#Thing` according to Table 6). It also has a property value "a" (`http://institutions.com/0.2/name` according to Table 8) with a literal value "_1" (`"Lycée de la Plage"` according to Table 9), and so on.

Once values and structure of the RDF data are defined, we formalize the whole RDF Sequence in Def. 9.

**Definition 9. RDF Sequence (*S-RDF*)**: *Given a set of triples* $T = \{t_i \mid t_i :< s, p, o >\}$*, its RDF Sequence is a 2-tuple consisting of two parts, defined as:*

*S-RDF(T) =<S-RDF-V(T), S-RDF-S(T)>*

*where:*

- *S-RDF-V(T) is the set of values of T defined in Def. 7.*
- *S-RDF-S(T) is the structure of T defined in Def. 8.* ♦

The S-RDF is built to represent triples considering the structure and values separately. Thus, an analysis over either the data or structure can be easily performed. Another benefit of this serialization format is the easy detection of some graph properties as the number of relationships (e.g., degree centrality measure) with respect to other serialization formats. Moreover, the storage space is reduced, since an IRI, which appears several times in an RDF document as a resource or property, is represented as a unique short key (e.g., `key:1` represents value: `http://institutions.com/0.2/S0991` or `key:A` represents value:

`http://www.w3.org/2002/07/owl#Thing`, respectively). This new serialization format can be consider as part of RDF partition strategies where the models improve the storage and the querying; however, when the repository is exported/outsourced, the format is still the same (e.g., RDF/XML, Turtle). Our serialization is a new way to represent data to be shared on the Web, improving the storage without losing the readability.

In the following section, we evaluate our S-RDF with respect to the current serialization formats.

## 6 Experimental Evaluation

### 6.1 Experimental Environment and Datasets

In order to evaluate and validate our serialization format, we developed a desktop and online[7] prototype system based on Java and Jena[8] to manage the RDF data. Experiments were undertaken on a MacBook Pro, 2.2 GHz Intel Core(TM) i7 with 16.00GB, running a MacOS Mojave and using a Sun JDK 1.7 programming environment.

Our prototype was used to perform several experiments to evaluate the viability and the compression rate of our approach in comparison with the works proposed in the literature. To do so, we considered two datasets:

- **Data 1**: the *DBpedia person data*[9] with 16,842,176 triples; and
- **Data 2**: the *DBpedia geo coordinates*[10] with 151,205 triples.

Note that some of the serialization formats (e.g., RDFa, HDT++) described in the related work section were not evaluated since there are no tools available that can manage huge quantity of triples. They are mainly document oriented converters (e.g., Easy-RDF[11], RDF-Translator[12]). For our readability test, HDT and HDT+ formats were analyzed since they have a binary representation and cannot be read by humans.

We describe as follows the tests performed to evaluate our proposal.

### 6.2 Evaluation

**Test 1:** We chose randomly 50,000 triples from **Data 1** in order to measure the compression rate of the data with respect to the size of the input (6,102,029

[7] S-RDF: `http://rdf-sequence.sigappfr.org`

[8] Jena is a Java framework for building Semantic Web applications. It provides a extensive Java libraries for helping developers develop code that handles RDF, RDFS, RDFa, OWL and SPARQL in line with published W3C recommendations - https://jena.apache.org/about_jena/about.html

[9] Information about persons extracted from the English and Germany Wikipedia, represented by the FOAF vocabulary - `http://wiki.dbpedia.org/Downloads2015-10`.

[10] Geographic coordinates extracted from Wikipedia - `https://wiki.dbpedia.org/downloads-2016-10`.

[11] Easy-Converte: http://www.easyrdf.org/converter

[12] RDF-Translator: https://rdf-translator.appspot.com

**Table 10.** Related Work Comparison for **Data 1**

| Serialization Format | Triples | Size | Compression Rate (%) |
|---|---|---|---|
| RDF/XML | 50,000 | 3,828,810 | 37.2535 |
| Turtle | 50,000 | 4,650,993 | 23.7796 |
| N-Triple | 50,000 | 6,151,004 | -0.8026 |
| N3 | 50,000 | 4,650,993 | 23.7796 |
| JSON-LD | 50,000 | 3,720,552 | 39.0276 |
| HDT | 50,000 | 944,196 (HDT)<br>130,151 (Index) | 82.3936 |
| **S-RDF** | **50,000** | **1,729,533** | **71.6564** |

bytes). Table 10 shows the results obtained for this test. HDT serialization format clearly overcomes the other ones (82.3936%), since it was created to minimize the storage. However, our serialization has also a good result (71.6564%) without losing the human readability criterion as the binary representation of HDT does. JSON-LD serialization has the biggest compression rate (39.0276%) among the W3C recommendation formats.

For **Data 2**, we also chose 50,000 triples from this dataset, having a size of 7,356,637 bytes. Table 11 shows similar results as the ones of *Data 1*. HDT obtained the best result with 75.6508%, while for our serialization format was 70.7767%. The JSON-LD serialization format has a 59.6130% of compression rate with respect to the input size.

**Table 11.** Related Work Comparison for **Data 2**

| Serialization Format | Triples | Size | Compression Rate (%) |
|---|---|---|---|
| RDF/XML | 50,000 | 4,338,226 | 41.0298 |
| Turtle | 50,000 | 5,908,228 | 19.6885 |
| N-Triple | 50,000 | 7,356,638 | -0.0001 |
| N3 | 50,000 | 5,908,228 | 19.6885 |
| JSON-LD | 50,000 | 2,971,124 | 59.6130 |
| HDT | 50,000 | 1,665,119 (HDT)<br>126,163 (Index) | 75.6508 |
| **S-RDF** | **50,000** | **2,149,852** | **70.7767** |

**Test 2:** Since there is no benchmark model for readability available in the literature to compare the existing serialization formats, we propose three questions which are related to several aspects of the RDF structure. (i) The first question is about relations, which can help to the end-user to recognize some important nodes according to the context, (ii) the second one is related to the terminal nodes, and (iii) the third one to literal values. The questions are presented as follows:

1. Is the resource X the most related one of the data?
2. Is the resource Y a terminal node in the data?
3. How many literal values has the resource Z?

where X, Y, and Z are resources that belong to the set of triples used to evaluate this test (see Listing 2).

```
@prefix ns0: <http://institutions.com/0.2/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
<http://institutions.com/0.2/S0991>
  ns0:validated ns0:N_1 ;
  ns0:invalidated ns0:N_2 ;
  ns0:expired ns0:N_3 .
ns0:N_1
  ns0:name "Lycee du Parc"^^xsd:string ;
  ns0:postalCode "64600"^^xsd:string ;
  ns0:established "1985-05-19"^^xsd:date ;
  a owl:Thing .
ns0:N_2
  ns0:name "Ecole du Parc"^^xsd:string ;
  ns0:postalCode "64100"^^xsd:string ;
  a owl:Thing .
ns0:N_3
  ns0:name "Universite du Parc"^^xsd:string ;
  ns0:postalCode "64200"^^xsd:string ;
  a owl:Thing .
```

**Listing 2.** List of triples used for **Test 2**, serialized in Turtle format

In this test, we evaluated our human readability criterion by surveying 40 people that have under- and post-graduate degrees in computer science[13]. The participants evaluated the serialization formats through the three previous questions, choosing an option to answer them: *Yes*, *No*, and *I do not know* for the two first questions, and a value among *1* to *5* and *"I do not know"* option for the third one.

To evaluate the results, we calculated the F-measure, based on the Recall (R) and Precision (PR). These criteria are commonly adopted in information retrieval and are calculated as follows:

$$\mathbf{PR} = \frac{A}{A+B} \in [0,1] \qquad \mathbf{R} = \frac{A}{A+C} \in [0,1] \qquad \mathbf{F\text{-}measure} = \frac{2 \times PR \times R}{PR+R} \in [0,1]$$

where A is the number of correct answers; B is the number of wrong answers; and C is the number of *"I do not know"* options selected by the participants.

Table 12 shows the results obtained for this evaluation. For *Question 1*, the N3 serialization format obtained the best Precision (84.62%), while the one for our serialization format was 84.00%. RDF/XML and JSON-LD obtained the lowest Precision (22.73% and 36.36%, respectively). By regarding the F-measure, we can observe that Turtle, N3, and our proposal (S-RDF) help user to identify some graph properties as the centrality measure, since they obtained a high result (over 68.00%). For *Question 2*, which is related to identify terminal nodes, most of the serialization formats obtained a similar F-measure (≈61.00%), but for the RDF/XML format, the F-measure was 43.14% due to the low Recall (35.48%). A low Recall can be interpreted as the serialization format is not easy-readable for the user. For *Question 3*, Turtle obtained the best F-measure (73.02%), while for S-RDF the value was 68.85%. By analyzing the answers, we noticed that some

[13] The form is available here: `https://forms.gle/DNMfsp5LL3nw1hW9A`

people confused the entity property and its respective value as a literal value since they only counted the number of elements associated to the entity.

**Table 12.** Number of correct, incorrect, and ambiguous values of each question per serialization format

| Serialization Format | Question 1 | | | | | | Question 2 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **C** | **I** | **A** | **Precision** | **Recall** | **F measure** | **C** | **I** | **A** | **Precision** | **Recall** | **F measure** |
| RDF/XML | 5 | 17 | 18 | 22.73% | 21.74% | 22.22% | 11 | 9 | 20 | 55.00% | 35.48% | 43.14% |
| Turtle | 24 | 6 | 10 | 80.00% | 70.59% | 75.00% | 17 | 5 | 18 | 77.27% | 48.57% | 59.65% |
| N-Triple | 10 | 6 | 24 | 62.50% | 29.41% | 40.00% | 15 | 6 | 19 | 71.43% | 44.11% | 54.55% |
| N3 | 22 | 4 | 14 | 84.62% | 61.11% | 70.97% | 21 | 2 | 17 | 91.30% | 55.26% | 68.85% |
| JSON-LD | 8 | 14 | 18 | 36.36% | 30.77% | 33.33% | 18 | 9 | 13 | 66.67% | 58.06% | 62.07% |
| **S-RDF** | 21 | 4 | 15 | 84.00% | 58.33% | 68.85% | 19 | 8 | 13 | 70.37% | 59.38% | 64.41% |

| Serialization Format | Question 3 | | | | | |
|---|---|---|---|---|---|---|
| | **C** | **I** | **A** | **Precision** | **Recall** | **F measure** |
| RDF/XML | 15 | 6 | 19 | 71.43% | 62.50% | 66,67% |
| Turtle | 23 | 14 | 3 | 62.12% | 88.46% | 73.02% |
| N-Triple | 8 | 10 | 22 | 44.44% | 26.67% | 33.33% |
| N3 | 19 | 10 | 11 | 65.52% | 63.33% | 64.41% |
| JSON-LD | 6 | 23 | 11 | 20.69% | 35.29% | 26.09% |
| **S-RDF** | 21 | 12 | 7 | 63.63% | 75.00% | 68.85% |

C = Correct
I = Incorrect
A = I do not know option (ambiguous).

Table 13 shows the global results of this test. In this table, we can identify two groups: G1: RDF/XML, N-Triples, and JSON-LD with a F-measure around 43.00%, and G2: Turtle, N3, and S-RDF with a value around 68.00%. One of the reasons of the low F-measure obtained by G1, is that these formats were created to keep the interoperability among system, using XML and JSON formats for example. The results demonstrate that our serialization format (S-RDF) can improve the storage without losing the human-readability criterion.

**Table 13.** Total number of correct, incorrect and ambiguous values per serialization format

| Serialization Format | C | I | A | Precision | Recall | F-measure |
|---|---|---|---|---|---|---|
| RDF/XML | 31 | 32 | 47 | 49.72% | 39.91% | 44.01% |
| Turtle | 64 | 25 | 31 | 73.14% | 69.20% | 69.22% |
| N-Triple | 33 | 22 | 65 | 59.46% | 33.40% | 42.63% |
| N3 | 62 | 16 | 42 | 80.48% | 59.90% | 68.08% |
| JSON-LD | 32 | 46 | 42 | 41.24% | 41.38% | 40.50% |
| **S-RDF** | 61 | 24 | 35 | 72.67% | 64.24% | 67.37% |

C = Correct, I = Incorrect, A = I do not know option (ambiguous).

## 7 Conclusion

In this paper, we propose a new serialization format, called *S-RDF*, which represents the RDF graph structure and values, separately. This format is focused on human readability, storage, and data redundancy to represent medium and large datasets. We evaluated our serialization format in terms of compression rate and human readability with respect to the state of the art. Results show a high compression without losing human readability, which is an advantage over the serialization formats created to minimize storage. According to the survey evaluation, our S-RDF allows identify easily the resources with more relations in the RDF graph (degree centrality measure) by identifying the entity with the bigger number of entity properties.

We are currently working on normalization methods over the S-RDF in order to provide a unique and deterministic output for similar inputs.

## References

1. Microdata to RDF - Second Edition - Transformation from HTML+Microdata to RDF. `https://www.w3.org/TR/microdata-rdf/`, 2014. Online; accessed 2019-07-01.
2. M. D. A. Phillips. Tags for Identifying Languages. `https://tools.ietf.org/html/bcp47`. Online; accessed 2019-07-01.
3. M. A. Bornea, J. Dolby, A. Kementsietsidis, K. Srinivas, P. Dantressangle, O. Udrea, and B. Bhattacharjee. Building an efficient rdf store over a relational database. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 121–132, New York, NY, USA, 2013. ACM.
4. G. Būmans and K. Čerāns. Rdb2owl: A practical approach for transforming rdb data into rdf/owl. In *Proceedings of the 6th International Conference on Semantic Systems*, I-SEMANTICS '10, pages 25:1–25:3, New York, NY, USA, 2010. ACM.
5. C. Chantrapornchai and P. Makpaisit. Tripleid-c: Low cost compressed representation for rdf query processing in gpus. In *Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region*, HPC Asia 2018, pages 261–270, New York, NY, USA, 2018. ACM.
6. R. Cyganiak, D. Wood, and M. Lanthaler. RDF 1.1 Concepts and Abstract Syntax. Technical report, 2014. Online; accessed 2016-12-06.
7. M. Duerst and M. Suignard. Internationalized Resource Identifiers (IRIs). Technical report, Microsoft Corporation, 2004.
8. J. D. Fernández. Binary rdf for scalable publishing, exchanging and consumption in the web of data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, pages 133–138, New York, NY, USA, 2012. ACM.
9. F. Goasdoué, I. Manolescu, and A. Roatiş. Getting more rdf support from relational databases. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, pages 515–516, New York, NY, USA, 2012. ACM.
10. M. Hausenblas, L. Ding, and V. Peristeras. Linked open government data. *IEEE Intelligent Systems*, 27:11–15, 2012.

11. A. Hernndez-Illera, M. A. Martnez-Prieto, and J. D. Fernndez. Serializing rdf in compressed space. In *2015 Data Compression Conference*, pages 363–372, April 2015.
12. J.-Y. Huang, C. Lange, and S. Auer. Streaming transformation of xml to rdf using xpath-based mappings. In *Proceedings of the 11th International Conference on Semantic Systems*, SEMANTICS '15, pages 129–136, New York, NY, USA, 2015. ACM.
13. N. Konstantinou, D. Kouis, and N. Mitrou. Incremental export of relational database contents into rdf graphs. In *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*, WIMS '14, pages 33:1–33:8, New York, NY, USA, 2014. ACM.
14. D. Lacoste, K. P. Sawant, and S. Roy. An efficient xml to owl converter. In *Proceedings of the 4th India Software Engineering Conference*, ISEC '11, pages 145–154, New York, NY, USA, 2011. ACM.
15. O. Lassila, R. R. Swick, W. Wide, and W. Consortium. Resource description framework (rdf) model and syntax specification, 1998.
16. G. K.-M. L. N. L. Manu Sporny, Dave Longley. JSON-LD 1.0, A JSON-based Serialization for Linked Data, W3C Recommendation 16 January 2014. `https://www.w3.org/TR/json-ld/`, 2014. Online; accessed 2017-10-27.
17. M. J. O'Connor and A. Das. Acquiring owl ontologies from xml documents. In *Proceedings of the Sixth International Conference on Knowledge Capture*, K-CAP '11, pages 17–24, New York, NY, USA, 2011. ACM.
18. P. F. P.-S. Patrick J. Hayes. RDF 1.1 Semantics, W3C Recommendation 25 February 2014. `https://www.w3.org/TR/rdf11-mt/#literals-and-datatypes`, 2014. Online; accessed 2019-07-01.
19. P. E. Salas, E. Marx, A. Mera, and J. Viterbo. Rdb2rdf plugin: Relational databases to rdf plugin for eclipse. In *Proceedings of the 1st Workshop on Developing Tools As Plug-ins*, TOPI '11, pages 28–31, New York, NY, USA, 2011. ACM.
20. P. A. Sandro Hawke, Ivan Herman. W3C Semantic Web Activity. `https://www.w3c.org/2001/sw/`, 2001. Online; accessed 2018-12-06.
21. J. F. Sequeda, M. Arenas, and D. P. Miranker. On directly mapping relational databases to rdf and owl. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 649–658, New York, NY, USA, 2012. ACM.
22. S. Stefanova and T. Risch. Scalable reconstruction of rdf-archived relational databases. In *Proceedings of the Fifth Workshop on Semantic Web Information Management*, SWIM '13, pages 5:1–5:4, New York, NY, USA, 2013. ACM.
23. P. T. T. Thuy, Y.-K. Lee, and S. Lee. Dtd2owl: Automatic transforming xml documents into owl ontology. In *Proceedings of the 2Nd International Conference on Interaction Sciences: Information Technology, Culture and Human*, ICIS '09, pages 125–131, New York, NY, USA, 2009. ACM.
24. P. T. T. Thuy, N. D. Thuan, Y. Han, K. Park, and Y.-K. Lee. Rdb2rdf: Completed transformation from relational database into rdf ontology. In *Proceedings of the 8th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '14, pages 88:1–88:7, New York, NY, USA, 2014. ACM.
25. R. Ticona-Herrera, J. Tekli, R. Chbeir, S. Laborie, I. Dongo, and R. Guzman. *Toward RDF Normalization*, pages 261–275. Springer International Publishing, Cham, 2015.
26. J.-Y. Vion-Dury. Using rdfs/owl to ease semantic integration of structured documents. In *Proceedings of the 2013 ACM Symposium on Document Engineering*, DocEng '13, pages 189–192, New York, NY, USA, 2013. ACM.