Summarizing Entity Temporal Evolution in Knowledge Graphs

Mayesha Tasnim*
RWTH Aachen
mayesha.tasnim@rwth-aachen.de

Diego Collarana
Fraunhofer IAIS & Univ. of Bonn
collaran@cs.uni-bonn.de

Damien Graux
Fraunhofer IAIS
damien.graux@iais.fraunhofer.de

Fabrizio Orlandi
ADAPT Centre, Trinity College Dublin
orlandif@tcd.ie

Maria-Esther Vidal
TIB, L3S & Leibniz Univ. of Hannover
maria.vidal@tib.eu

ABSTRACT

Knowledge graphs are dynamic in nature, new facts about an entity are added or removed over time. Therefore, multiple versions of the same knowledge graph exist, each of which represents a snapshot of the knowledge graph at some point in time. Entities within the knowledge graph undergo evolution as new facts are added or removed. The problem of automatically generating a summary out of different versions of a knowledge graph is a long-studied problem. However, most of the existing approaches are limited to a pairwise version comparison. This limitation makes it difficult to capture a complete evolution out of several versions of the same knowledge graph. To overcome this limitation, we envision an approach to create a summary graph capturing temporal evolution of entities across different versions of a knowledge graph. The entity summary graphs may then be used for documentation generation, profiling or visualization purposes. First, we take different temporal versions of a knowledge graph and convert them into RDF molecules. Secondly, we perform Formal Concept Analysis on these molecules to generate summary information. Finally, we apply a summary fusion policy in order to generate a compact summary graph which captures the evolution of entities.

KEYWORDS

RDF Knowledge Graph, RDF Molecules, Entity Evolution

ACM Reference Format:
Mayesha Tasnim, Diego Collarana, Damien Graux, Fabrizio Orlandi, and Maria-Esther Vidal. 2019. Summarizing Entity Temporal Evolution in Knowledge Graphs. In *Companion Proceedings of the 2019 World Wide Web Conference (WWW '19 Companion), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3308560.3316521

1 INTRODUCTION

Knowledge graphs evolve over time with the addition of new entities and relationships, or the modification of its existing ones [4]. It is often the case that different versions of an RDF graph are maintained separately. For example, DBpedia releases a new version of its datasets on a yearly basis. Typically most applications based on knowledge graphs are concerned with the latest version available at any point in time. These graphs contain the most updated information, however, they are missing the knowledge about how these entities transform over multiple versions. From the perspective of a single entity, it can often be interesting to observe how it evolves over time. This information can be new knowledge that can add value to the existing knowledge graph. In this work, we propose a technique using RDF molecules and Formal Concept Analysis to summarize the temporal evolution of knowledge graphs.

Consider an RDF data entity representing the same person in different temporal versions of a knowledge graph, as depicted in Figure 1. The updates in the properties of this entity correspond to real-world changes over time. For example, the person might *inter alia* relocate to different cities or change job title. New properties may also be added to this entity, e.g. when this person becomes a parent for the first time. If only *some* versions of this knowledge graph are considered and others are ignored, the information obtained is valid solely in the context of that time period, and the knowledge about how this entity evolved over time is lost. For instance, considering the versions in Figure 1a and Figure 1d (years 2010 and 2016), it can be observed that this person has the same spouse P. But this observation misses the information that the person had a different spouse Q from 2012 to 2014 (see Figures 1b & 1c).

We motivate our work using this problem and propose a technique to summarize the evolution of entities in knowledge graphs. In this technique we identify entities in different versions of a knowledge graph that correspond to the same real-world object and apply formal concept analysis to generate a summary of the evolution of these entities. The generated *summary molecule* contains a compact representation of all the object and data properties the molecule was connected to over time, along with the temporal information indicating the ranges of validity for each property. In order to ensure compactness, the summary molecule contains each distinct object or data property only once, as shown in Figure 1e.

2 RELATED WORK

Knowledge graphs are becoming increasingly dynamic in nature and approaches have been proposed [9] to (i) detect changes during their evolution, (ii) represent change information (using vocabularies) and (iii) propagate changes to replicas or federated systems [2]. Approaches for change detection mainly focus on computing "deltas" (or changesets[1]) between two versions of a knowledge

ACM ISBN 978-1-4503-6675-5/19/05.
https://doi.org/10.1145/3308560.3316521

[1]https://www.w3.org/2009/12/rdf-ws/papers/ws07 (accessed on 22/02/2019)

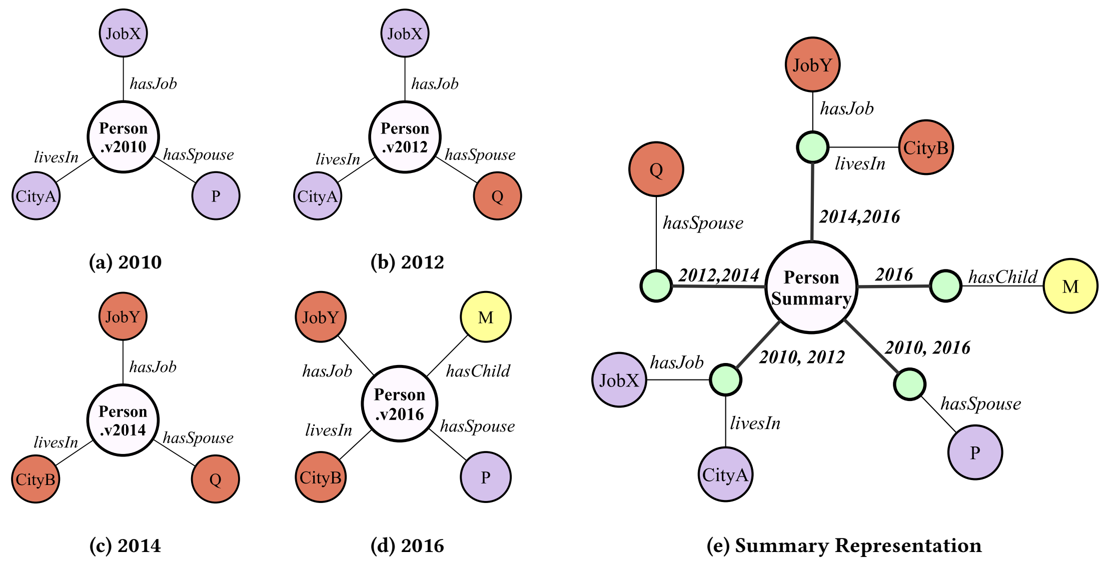

Figure 1: Visual representation of a person info obtained from four yearly molecules.

graph at different granularity levels [9]: dataset, resource and statement level. The Talis Changeset Vocabulary[2] defines a set of terms for describing changes to resource descriptions. In the context of this vocabulary, a resource description is the set of triples that includes a description of a resource. The DELTA-LD framework detects and classifies the changes between two versions of a linked dataset and represents them with the DELTA-LD change model [8]. DSNotify [6] is a change-detection system capable of detecting and fixing broken links between resources in two different versions of a dataset. In [7], in order to study the dynamics of LOD, the authors propose a framework for detecting and analysing changes and the evolution history of LOD datasets. Moreover, changes are automatically categorised according to their level of complexity and are represented using an ontology, hence allowing SPARQL to be used to query data changes. Specific queries need to be constructed in order to extract information about the history of changes for a particular class/entity across different versions. In contrast, our approach allows for automatic extraction and exploration of all changes of a class/entity over time.

Producing a summary of an entity evolution can be also tackled as an integration problem. Integration frameworks can be used to solve the problem of extracting the temporal evolution of an entity. In this regard, a central part of this work is represented by the MINTE approach [1]. We adopt the fusion policies for data integration described by Collarana et al. in order to produce a temporal evolution summary of entities across different versions of a knowledge graph. Similar to MINTE, ODCleanStore [5] is an ETL framework for integrating RDF data. It relies on SILK to perform instance matching and provides custom data fusion modules to merge the data of the discovered matches. The aim of the said integration frameworks is to map different data sources with possibly varying schema, i.e., they perform inter-schema mapping. However, even in these cases, creating a summarized view of an entity could only be supported on a superficial level by processing and filtering query results. Hence, we identify the need for an approach to automatically produce summaries of entities' evolution, as described in the following sections.

[2]http://vocab.org/changeset/schema (accessed on 22/02/2019)

3 THE APPROACH

3.1 Preliminaries

Given different versions of a knowledge graph, e.g., DBpedia 2010, 2012, 2014, and 2016, and an entity type, e.g., Person. Our approach automatically produces a summary of evolution over time of the entities under the specified type. Each entity summary is composed of the evolution of properties and relations among these entities. To better understand our approach, we define the main concepts, i.e., RDF Molecule, Formal Concept Analysis and Fusion Policies, respectively.

Definition 3.1 (RDF Molecule [3]). If G is a given RDF Graph, we define an RDF Molecule M as a sub-graph of G such that,

$$M = \{t_1, \ldots, t_n\}, \forall (i,j) \in \{1, \ldots, n\}^2 \left(subject(t_i) = subject(t_j)\right)$$

where $t_1, \ldots, t_n$ denote the triples in M. In other words, an RDF Molecule M consists of triples which have the same subject. The RDF molecules are used as unit to produce the entity summary.

Definition 3.2 (Formal Concept Analysis [10]). aims at describing the relationships between objects and their attributes by considering binary data tables. In our approach, we apply the algorithm proposed by V. Vychodil [10], transforming RDF molecules into the binary data table it requires. Formal concepts are defined as conceptual clusters found within entity-property data tables. These data tables have rows corresponding to entities, and columns corresponding to the properties of those entities. Formal concepts are a set of $< A, B >$ pairs where A is the set of entities, B is a set of

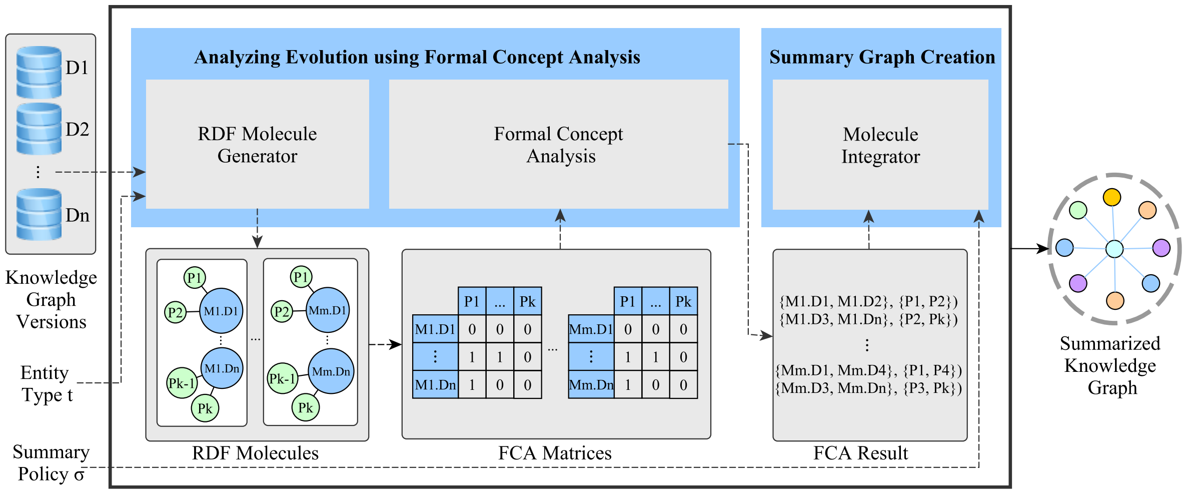

Figure 2: The approach receives a set of knowledge graph versions, an entity type, and a summary policy. The output is the entity summaries describing the evolution of knowledge among the different versions of the knowledge graph.

properties, and all the entities in A contain all the properties in B. A is known as *extent* and B is known as *intent*.

Definition 3.3 (Fusion Policy [1]). To produce a temporal evolution summary of entities spread over different versions of a knowledge graph, we resort to the concept of fusion policies defined by Collarana et al. [1]. A fusion policy is a set of rules operating on the triple level, which are triggered by a certain combination of predicates and objects. Fusion policies resort to an ontology O to resolve possible conflicts and inequalities on the levels of resources, predicates, objects and literals.

In this paper, we employ RDF molecules, Formal Concept Analysis and Fusion Policies to address the following problem: given a set of versions of RDF knowledge graphs, build an evolution summary of specific entities. The following section introduces and describes the architecture of our approach.

3.2 Architecture

Grounded on the semantic integration technique proposed by Collarana et al. [1], we propose a pipeline able to automatically create summaries of RDF entity evolution. Thus, a solution to *summarizing entity temporal evolution* out of different versions of an RDF knowledge graph is provided.

We propose a three-fold approach in order to identify equivalent entities in different versions of a knowledge graph and summarizing the evolution of those entities, thus providing a solution to the problem of *summarizing entity temporal evolution* in RDF knowledge graphs. This approach has three essential components. First, the pipeline takes input a set of knowledge graphs representing different temporal versions of the same knowledge graph. These graphs are then converted into a set of RDF molecules, from which molecules that represent the same real-world entity are grouped together. Each group of equivalent molecules are then converted into an $M \times N$ binary matrix. Second, the $M \times N$ matrix is supplied to the FCA component which performs formal concept analysis to summarize the temporal evolution of the molecules. Third, a summary fusion policy is applied to each output of the FCA component in order to produce a set of summary molecules. Each summary molecule represents the temporal evolution of a single entity over the knowledge graph versions taken as input. Figure 2 depicts the main components of this architecture.

3.3 Conversion of Knowledge Graphs to Groups of Equivalent RDF Molecules

The pipeline receives any number of Knowledge Graphs $\phi_1(D)$, $\phi_2(D)$,…,$\phi_n(D)$ as input where $1, 2, \ldots n$ represent the different temporal versions of the same Knowledge Graph $\phi(D)$. First each graph is individually converted into sets of RDF molecules. As defined in Preliminaries (see §3.1), RDF molecules consist of triples that have the same subject. Thus we obtain RDF molecule sets $S_1, S_2 \ldots S_n$ which correspond to graphs $\phi_1(D), \phi_2(D) \ldots \phi_n(D)$ respectively. The pipeline then identifies equivalent molecules within $S_1, S_2 \ldots S_n$. As $\phi_1(D), \phi_2(D) \ldots \phi_n(D)$ are different temporal versions of the same knowledge graph, it can be inferred that there exists equivalent molecules $M_1, M_2 \ldots M_n$ such that $M_1 \in S_1, M_2 \in S_2 \ldots M_n \in S_n$ and $M_1, M_2 \ldots M_n$ all represent the same real-world entity. For the sake of simplicity it is assumed here that equivalent entities retain the same URI. Practically, semantic similarity measures as demonstrated in [1] can also be integrated with this pipeline to identify equivalent entities in cases where the URI is different.

The equivalent RDF Molecules are then grouped together and each group is converted into a $M \times N$ matrix, where M corresponds to the number of molecules in the group and N corresponds to the number of distinct object or data property the molecules all contain. Each element in the matrix $e_{(i,j \rightarrow k)}$ represents whether molecule M_i contains a predicate j that has an object k. This value can be either `true` or `false` and is represented by a 1 and 0, respectively.

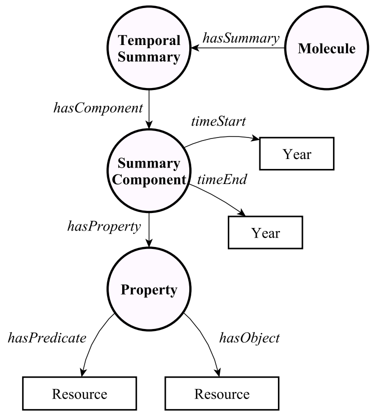

Figure 3: The ontology used to apply summary fusion policy for the creation of summary molecules.

Figure 1 presents a scenario where `Person.v2010`, `Person.v2012`, `Person.v2014` and `Person.v2016` all represent the same real world entity `Person` from Knowledge Graphs from year 2010, 2012, 2014 and 2016 respectively. In this first step of the pipeline, these molecules would be extracted from their respective knowledge graphs, identified as equivalent molecules, and then grouped together. This group of molecules would then be converted into a 4×7 matrix, 4 being the number of molecules in the group and 7 being the total number of distinct object/data properties in all the molecules. This matrix is shown in Table 1a. The column headers, i.e. properties for this matrix are shown separately in Table 1d. Once the $M \times N$ matrix has been created, it is then passed on to the next component in the pipeline to perform Formal Concept Analysis and summarize the temporal evolution in equivalent RDF entities.

3.4 Applying Formal Concept Analysis to Obtain a Summary of Evolution

As first mentioned in Definition 3.2, formal concept analysis studies binary object-attribute tables to describe the relationship between *objects* and their *attributes*. Our approach first converts knowledge graphs to RDF molecules. Within a single knowledge graph, an RDF molecule can be considered as an *object* while its object or data properties can be considered as *attributes*. When RDF molecules are modeled in this way, we are able to apply the formal concept analysis algorithm to compute formal concepts. The creation of these binary matrices is discussed in the previous section.

We apply the algorithm by V. Vychodil [10] on each group of RDF molecules obtained from the previous step. The algorithm returns a set of formal concepts $< M, P >$ where M is a set of all the molecules that have all the properties contained in P. In our approach the output $< M, P >$ from formal concept analysis gives us a set of molecules that have the same properties throughout different knowledge graph versions.

Table 1 demonstrates the input and output of formal concept analysis when applied to the RDF molecules in our motivation example. Table 1a shows the 4×7 binary matrix representing the

Algorithm 1: CreateSummary(FCAResult)

```
summary ← list() ;         // Initialize empty list
n ← FCAResult.length;
for j ← 1 to n do
    (M, P) ← FCAResult[j];
    if $M.length \geq 1$ and $P.length \geq 0$ then
        summary ← summary + (M, P) ;
        FCAResult ← FCAResult − (M, P) ;
n ← FCAResult.length ;          // Recompute length
m ← summary.length;
for j ← 1 to n do
    (M, P) ← FCAResult[j];
    if $M.length1$ and $P.length \geq 0$ then
        for k ← 1 to m do
            (X, Y) ← summary[k];
            if $M[1] \in X$ then
                $P \leftarrow P - Y$
        if $P.length \geq 0$ then
            summary ← summary + (M, P) ;
return summary
```

Figure 4: Algorithm for selecting distinct properties that form the *Summary Molecule*

object-attribute table. Table 1b shows the output of the FCA algorithm when supplied with this matrix. It supplies the information of which properties of the molecule has remained the same over which years. For example, Table 1b shows us that the entity Person had the same job (`hasJob` → *JobX*) and lived in the same city (`livesIn` → *CityA*) over the years 2010 and 2012.

The output of the FCA algorithm is supplied to the next step in order to create summary molecules.

3.5 Applying Fusion Policy to Integrate Summary into Knowledge Graph

To integrate the result obtained from the formal concept analysis algorithm, we resort to extend the fusion policies defined by Collarana et al. [1]. They defined the following: (1) the *Union Policy*, which includes all predicates-object pairs, removing the one that are syntactically the same; (2) the *Subproperty Policy*, which tracks if a property of one RDF molecule is an `rdfs:subPropertyOf` of a property of another RDF molecule, i.e., $\{r_1, p_1, A\}, \{r_2, p_1, B\} + O + subPropertyOf(p_1, p_2) \models \{\sigma_r(r_1, r_2), p_2, \sigma_v(A, B)\}$. As a result of applying this policy, the property p_1 is replaced with a more general property p_2; and (3) the *Authority Graph Policy*, which allows for defining one RDF graph as a prevalent source selecting its properties in case of property conflicts, i.e., properties annotated as `owl:FunctionalProperty`, equivalent properties `owl:equivalent-Property`, and equivalent classes annotated with `owl:equivalent-Class` or `owl:sameAs`.

For the purpose of creating a *compact summary* of the temporal evolution of RDF molecules, we define an additional fusion policy

	P1	P2	P3	P4	P5	P6	P7
M1	1	0	1	0	1	0	0
M2	1	0	1	0	0	1	0
M3	0	1	0	1	0	1	0
M4	0	1	0	1	1	0	1

(a) FCA Input

Molecule	Property
M1	P1,P3,P5
M2	P1,P3,P6
M3	P2,P4,P6
M4	P2,P4,P5,P7
M1, M2	P1,P3
M2, M3	P6
M3, M4	P2,P4
M1, M4	P5

(b) FCA Output

	Molecule
M1	Person.v2010
M2	Person.v2012
M3	Person.v2014
M4	Person.v2016

(c) Molecules

	Property
P1	`livesIn` $\rightarrow$ *CityA*
P2	`livesIn` $\rightarrow$ *CityB*
P3	`hasJob` $\rightarrow$ *JobX*
P4	`hasJob` $\rightarrow$ *JobY*
P5	`isSpouseOf` $\rightarrow$ *P*
P6	`isSpouseOf` $\rightarrow$ *Q*
P7	`isFatherOf` $\rightarrow$ *M*

(d) Properties

Table 1: Preparing equivalent RDF Molecules for Formal Concept Analysis: `Person` molecules from 4 different knowledge graph versions are grouped and compiled into a matrix for performing formal concept analysis. Rows correspond to each molecule, while columns correspond to distinct properties.

called the *Summary Policy*. *Summary Policy* selects distinct properties from the output of the FCA algorithm, and follows the ontology shown in Figure 3 to create a *temporal summary graph*. A visual representation of the temporal summary graph according to our motivation example is shown in Figure 1e. For the sake of compactness, the summary graph is designed to have only one occurrence of each distinct property.

A *temporal summary graph* is created by first selecting all distinct properties out of the result obtained from the previous step. This is done by applying the algorithm shown in Figure 4 on a list of results like the one shown in Table 1b. The algorithm takes as input the result R of the FCA algorithm. For every $(M, P) \in R$, $|M| = 1$, it returns (M, P') such that, $\forall(X, Y) \in R$ where $|X| > 1$ and $M \in X$, $P' = P \cap Y$ and $P' \neq \phi$.

For example, for the row [{M4}, {P2,P4,P5,P7}] in Table 1b, the above algorithm is applied to obtain [{M4}, {P7}], i.e. [{Person.v2016}, {`hasChild` $\rightarrow M$}]. This can be seen in Figure 1e, where the property `hasChild` $\rightarrow M$ occurs only under the summary component of 2016. Every other property in Person.v2016 occurs also in other versions of the Person entity. This is represented in the summary graph under the summary components of (2010,2016) and (2014, 2016). The summary graph allows the possibility to reconstruct the original versions of the molecules. For example, in order to obtain the 2010 version of the Person molecule, it is only needed to select the edges of the summary graph that are annotated with the year 2010, and apply *Union Policy*.

4 CONCLUSIONS AND FUTURE WORK

In this paper, we envision an approach for creating entity summaries automatically out of different temporal versions of a knowledge graph. To do so, the proposed approach utilizes the concepts of RDF molecules, Formal Concept Analysis, and Fusion Policies. We have explained the architecture and pipeline where only one parameter is needed, i.e., the entity type. The entity evolution summaries created by the approach may serve to create documentation, to display a visualization of the entity evolution, or to make an analysis of changes. As future work, we plan to implement and evaluate the approach considering its scalability and applicability.

Acknowledgements: This work is supported by the German Ministry of Education and Research (BMBF) in the context of the project MLwin ("Maschinelles Lernen mit Wissensgraphen", grant no. 01IS18050F).

REFERENCES

[1] D. Collarana, M. Galkin, I. T. Ribón, M. Vidal, C. Lange, and S. Auer. MINTE: semantically integrating RDF graphs. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS*, pages 22:1–22:11, 2017.

[2] K. M. Endris, S. Faisal, F. Orlandi, S. Auer, and S. Scerri. Interest-Based RDF Update Propagation. In *The Semantic Web - ISWC 2015*, volume 9366, pages 513–529, 2015.

[3] J. D. Fernández, A. Llaves, and O. Corcho. Efficient rdf interchange (eri) format for rdf data streams. In *International Semantic Web Conference*, pages 244–259. Springer, 2014.

[4] T. Käfer, A. Wins, and M. Acosta. Modelling and Analysing Dynamic Linked Data using RDF and SPARQL. In *4th International Workshop on Dataset PROFIling and fEderated Search for Web Data (PROFILES) at the 16th International Semantic Web Conference (ISWC)*. CEUR-WS, 2017.

[5] J. Michelfeit and T. Knap. Linked data fusion in ODCleanStore. In *ISWC Posters and Demonstrations Track*, 2012.

[6] N. Popitsch and B. Haslhofer. DSNotify – A solution for event detection and link maintenance in dynamic datasets. *Journal of Web Semantics*, 9(3):266–283, sep 2011.

[7] Y. Roussakis, I. Chrysakis, K. Stefanidis, G. Flouris, and Y. Stavrakas. A flexible framework for understanding the dynamics of evolving RDF datasets. In *ISWC 2015 - International Semantic Web Conference*, volume 9366, pages 495–512, 2015.

[8] A. Singh, R. Brennan, and D. O'Sullivan. DELTA-LD: A Change Detection Approach for Linked Datasets. In *4th Workshop on Managing the Evolution and Preservation of the Data Web (MEPDaW)*, 2018.

[9] J. Umbrich, B. Villazón-Terrazas, and M. Hausenblas. Dataset Dynamics Compendium: A Comparative Study. In *Proceedings of the First International Workshop on Consuming Linked Data (COLD2010)*, 2010.

[10] V. Vychodil. *A new algorithm for computing formal concepts.* na, 2008.