Towards Understanding the Changing Web: Mining the Dynamics of Linked-Data Sources and Entities

[work in progress]

Jürgen Umbrich and **Marcel Karnstedt**
Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway, Ireland
firstname.lastname@deri.org

Sebastian Land
Rapid-I GmbH
land@rapid-i.com

Abstract

A huge amount of content found on the Web is dynamic by its nature, particularly with the rise of Web 2.0 and beyond. This is of special interest for the Semantic Web community, not only but particularly regarding resources on the Linked Open Data (LOD) Web. However, the dataset dynamics of the LOD graph are hardly explored so far. Existing approaches from the traditional HTML Web are not sufficient to mine and discover the dynamics with satisfying accuracy and efficiency, as they do not consider the special characteristics of LOD. We present first initial results on this topic and discuss future steps. First results are obtained by mining the groups of URIs with similar change frequencies and by applying time series techniques as well as clustering techniques.

1 Motivation

At the time of writing, we can find several hundred datasets published as Linked Open Data (LOD) on the Web. The LOD cloud contains up to several billion Resource Description Framework (RDF) triples of machine readable information, describing real world entities (resources) and the relations among them. This data forms a huge directed and labelled graph (resources are nodes, relations are edges), which can be accessed, traversed and consumed by humans and intelligent software agents. Currently, the so formed LOD graph (see Figure 1) contains billions of nodes but only a couple of million edges [1]. However, experts in the field commonly agree that this gigantic graph will even grow faster in the future and become more dense by adding more labelled links between the nodes. New data and links between the data are added either by humans or by machines (e.g., from data converters like Any23[2] or by content management systems like Drupal 7). The LOD research community focuses on the different issues in identifying, linking and publishing this data.

A still nearly unexplored field are the dynamics of the contained data sets. With the term *dataset dynamics* we refer to *content and interlinking changes in the Linked Data graph*. Besides content changes, the dynamics of nodes (i.e., entities) and links are of particular interest. These can be roughly categorised as follows:

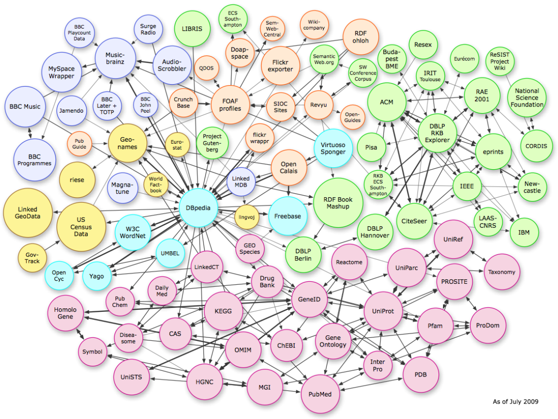

Figure 1: The LOD cloud in mid 2009, courtesy of Cyganiak and Jentzsch.

- **Dynamics of resources**: New nodes are added and old nodes are removed;
- **Dynamics between resources**: New relations are added (in the form of new triples) and old ones might be removed;

In a first attempt [Umbrich *et al.*, 2010] we monitored the change frequency of Linked Data resources and revealed that most of them show the same behavior as HTML Web documents. However, to the best of our knowledge, there is no published work describing attempts to apply techniques from other mature research areas dealing with the analysis and investigation of dataset dynamics. In this work, we take a closer look at how methods from the data mining and machine learning community can be used for this task. We identify suitable and promising techniques, discuss the benefit we expect from their application and present some preliminary results indicating the usefulness.

The motivating use-case for our study of dataset dynamics is to improve concurrent work on an efficient system for performing live queries over the LOD Web [Harth *et al.*, 2010]. The challenge is to decide which (sub-)queries can be run against a cache or data summary and which (sub-)queries have to be executed live over the Web content to guarantee up-to-date results. However, there are several other tasks related to managing Web data that can benefit from a deeper understanding of dataset dynamics: Web crawling and caching [Cho and Garcia-Molina, 2003], maintaining link integrity [Haslhofer and Popitsch, 2009], serving of continuous queries [Pandey *et al.*, 2003].

Regarding the above sketched use cases, there are some main questions we have to focus on. These are:

- What classes of dynamics can and should we distinguish?

[1] `http://esw.w3.org/TaskForces/CommunityProjects/LinkingOpenData/DataSets/[Statistics|LinkStatistics]`

[2] `http://any23.org/`

- What actual methods are suited to classify LOD resources with respect to their dynamics?
- On what features should these methods be based on?
- How can we efficiently predict future changes and the types of changes?

To start the investigation of these questions we firstly focus on one of the underlying main questions, which we see as the most interesting starting point:

What correlations between resources and their dynamics can we identify using methods from data mining, machine learning and graph analysis?

The specific correlations we expect and therefore plan to investigate are correlations between the dynamics of the resources and:

1. their domain names (i.e., their origin)
2. the used vocabulary (i.e., RDF predicates and classes)
3. their linkage (i.e., if one resource changes how likely is it that resources linked to it change as well)

To achieve this, we first started to continuously monitor a large set of LOD resources. Based on the observed dynamics, we can cluster these resources and apply correlation analysis between sets of resources as well as between resources and their features. In this work, we present initial results on this analysis and discuss future steps.

2 Dataset Dynamics on the LOD Web

In this section, we first introduce the data model of RDF and how it contributes to the LOD Web. Secondly, we discuss how our problem of studying dataset dynamics can be mapped to the problem of the dynamics of nodes in a labelled directed graph. Finally, we elaborate on the importance of data mining to reveal new insights into the dynamics of such a graph.

2.1 RDF and Linked Data

The Resource Description Framework [Manola and Miller, 2004] defines a data format for publishing schema-less data on the Web in the form of ($subject$, $predicate$, $object$) triples. These triples are composed of unique identifiers (URI references), literals (e.g., strings or other data values), and local identifiers called blank nodes as follows:

Definition 1. *(RDF Triple, RDF Term, RDF Graph) Given a set of URI references $\mathcal{U}$, a set of blank nodes $\mathcal{B}$, and a set of literals $\mathcal{L}$, a triple (s, p, o) $\in (\mathcal{U} \cup \mathcal{B}) \times \mathcal{U} \times (\mathcal{U} \cup \mathcal{B} \cup \mathcal{L})$ is called an* RDF triple*, We call elements of $\mathcal{U} \cup \mathcal{B} \cup \mathcal{L}$* RDF terms. *Sets of RDF triples are called* RDF graphs.

The notion of graph stems from the fact that RDF triples may be viewed as labelled edges connecting subjects and objects. RDF published on the Web according to the following principles is called *Linked Data* [Berners-Lee, 2006]: 1) URIs are used as names for things: in contrast to the HTML Web where URIs are used to denote content (documents, images), on the Semantic Web URIs can denote entities such as people or cities; 2) URIs should be dereferenceable using the Hypertext Transfer Protocol (HTTP): a user agent should be able to perform HTTP GET operations on the URI; 3) Useful content in RDF should be provided at these URIs: a Web server should return data encoded in one of the various RDF serialisations; 4) Include links to other URIs for discovery: a user agent should be able to navigate from an entity to associated entities by following links, which enables decentralised discovery of new data.

$s_{i,t}$	o_1	o_2	o_3	$\cdots$	o_n
$< URI_1 >$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$\cdots$	$s_{1,n}$
$< URI_2 >$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$	$\cdots$	$s_{2,n}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$< URI_m >$	$s_{m,1}$	$s_{m,2}$	$s_{m,3}$	$\cdots$	$s_{m,n}$

Table 1: Change matrix with $s_{i,t} \in \{-1, 0, 1, 2\}$.

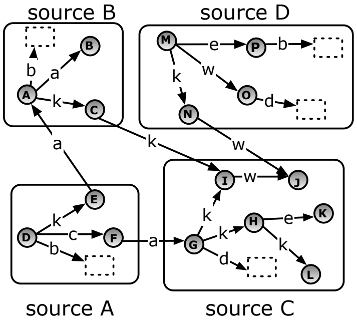

Figure 2: Abstraction of a subgraph of the LOD Web.

As such, Linked Data forms a labelled directed graph as depicted abstractly in Figure 2. We can see that subgraphs are contained in sources (a source is a container of a sub-graph and also has a unique URI, in analogy to Web documents), where resources and sources are interlinked. For the remainder of this paper, we focus on how to analyse the dynamics of the nodes and edges in such a graph. For further simplification, we do not distinguish between changing sources and entities – interested readers are referred to [Umbrich *et al.*, 2010]. We use the following definition of a change of a node:

Definition 2. *(Node state change $s_{i,t}$) A change of the state of a node identified by URI U_i is detected iff the tree of depth 1 with root node U_i differs between two observations o_{t-1}, o_t. We denote a change with $s_{i,t} = 2$, $s_{i,t} = 0$ otherwise. In addition, if a node appears in t we denote this with $s_{i,t} = 1$, if it disappears we use $s_{i,t} = -1$.*

The results of monitoring these changes over time are represented in an $m \times n$ change matrix (m resources and n observations) as illustrated in Table 1.

2.2 Data Mining

Traditionally, change frequencies of Web documents are modeled as Poisson processes [Cho and Garcia-Molina, 2000] and advanced estimators are used to predict the likelihood of the next change [Cho and Garcia-Molina, 2003]. Our current findings uncover that this model holds only for some resources of the LOD Web [Umbrich *et al.*, 2010]. In addition, we strongly believe that by applying more advanced machine learning methods we can perform better predictions of changes. By applying these techniques, we expect to reveal the existance of correlations between the change characteristics of different resources. As such, we will investigate correlation analysis, frequency analysis and techniques for change detection. Concrete techniques that we plan to assess are, among others, SVD and SVM, DFT and wavelet transformation as well as change point detection as known from data streams and graphs.

3 Describing Change Features and Clustering

A first step towards our general objectives is to identify nodes with similar dynamics. Thus, we first investigated how to cluster nodes wrt. to their characteristic dynamics. The main questions we have to answer for that is: What are features describing the dynamics of a resource? Possible describing features are:

- Average change frequency ratio: The dynamics of each node can be simply described by the number of node state changes divided by the overall number of snapshots in the monitored period.
- Statistical summaries of change behaviour: We could use statistical summaries of node state changes, such as central tendencies (arithmetic mean, median or interquartile mean) or statistical dispersion (standard deviation, variance or quantiles).
- Periodicities of changes: such periodicities can be determined by DFT or wavelets transformation, which would overcome obvious issues in using an average change frequency (e.g., one entity changing very often in the beginning of the monitored time but then being rather static, compared to another entity changing regularly in larger intervals over all the time).
- Eigenvalues or principal components: Using reduction techniques like SVD or PCA we can try to extract significant features capturing the characteristics of the dynamics.

Once we decided how to represent dataset dynamics on the basis of the dynamics of nodes we can apply clustering algorithms to group nodes with similar dynamics. Previous experiments indicated the existance of significantly different clusters [Umbrich *et al.*, 2010]. Afterwards, we will use the nodes from the gained clusters to analyse correlations among them and among single nodes and their edges. Methods we have in mind for that are classification approaches and correlation analysis, such as computing the correlation and covariance matrices between URI attributes; e.g., the correlation between node state changes and/or the type of incoming links. We will look into different methods to compute the correlation coefficients; e.g., the Pearson or Pearman's correlation coefficient or entropy-based mutual information/total correlation methods that allow us to detect even more general dependencies. These represent first steps to answer the aforementioned questions about concrete correlations.

4 Preliminary experiments

One question we already investigate in this work (see Section 4.3) is: How many clusters can we identify? Or: What is the "optimal" number of clusters? This is particularly relevant as we first decided to apply k-Means clustering, which requires an a-priori definition of k. If the number of clusters is too small, the clusters contain too many items which are not very similar. If it is too large, we do not capture the actual similarities. With this we want to overcome the limitations of using "soft" categories, e.g., using pre-defined classes like *static*, *low dynamic*, *medium dynamic*, *very dynamic*. This might be sufficient for some applications, but not for the particular approaches we are working on. For instance, we want to be able to decide accurately *when* a crawler has to revisit a site to update an index or how to set a sort of cache coherence time in order to achieve satisfying data freshness. The materials and methods used and applied to get these first preliminary results are described next.

4.1 Data

As the data for our experiments we use a 1% random sample from a data set containing 11 weekly observations of 161K LOD sources crawled using the LDSpider framework [3]. The average observations size is 440MB gzipped and a total number of 2.7M nodes and roughly 7 million links over the whole monitored time. We used the random sample to achieve manageable processing times.

For analysing the sample, we decided to use the data-mining framework RapidMiner[4]. RapidMiner offers a wide variety of data-mining tools, ranging from basic data cleaning to complex transformations and analyses.

4.2 Methods

For detecting node state changes between two observations o_{t-1}, o_t we used a straightforward approach based on a merge-sort scan. After sorting all relevant statements by their syntactic natural order (subject-predicate-object), we performed a pairwise comparison of the statements by scanning two observations in linear time. We record a state change as soon as the order of the statements differed between two observations (e.g., a data producer adds or removes outgoing edges and nodes).

The used sample consists of 26961 URIs and 10 observations (M=26961 × 10 matrix). In a pre-processing, step we transformed the rectangularly shaped input data into smooth series of values to finally compute the frequency spectrum with a Fourier analysis. The concrete steps involved are:

1. **Organise by change events**: We split the input change matrix into four matrices in the first transformation process. One matrix $M^{ev} = [m^{ev}_{i,t}]; i = 1, 2, \ldots, m; t = 1, 2, \ldots, n$ for each change event ev [ev = -1 (disappear), 0 (no change),1 (appear), 2 (change)]. The matrix values are encoded as follows:
$$m^{ev}_{i,j} = \begin{cases} 1 & \text{if } s_{i,t} = ev \\ 0 & \text{otherwise} \end{cases} \tag{1}$$
2. **Interpolation & Smoothing** We interpolated the data values for each matrix by tenfolding each single value and applying an exponential smoothing over the new time series. The interpolation is necessary since we have only 10 observations for each URI. Further, with the interpolation and smoothing we implicitly model the uncertainty of the event. We only know that a change event occurred between two observations, but not exactly when. This results in four matrices with 100 attributes for each URI. (M^{ev}_{intpol} = 26961 × 100 matrix)
3. **Fourier Analysis**: After the interpolation and smoothing of the data we performed a Fourier analysis. The analysis resulted in 32 spectrums for each URI ($M^{ev}_{fourier}$ = 26961 × 32 matrix).
4. **Join of Matrices by their URI**: Finally, we joined the four matrices by their URIs which resulted in our final matrix (M_{final} = 26961 × 128 matrix).

[3] http://code.google.com/p/ldspider/

[4] http://rapid-i.com/content/view/181/190/

4.3 Preliminary Results

Figure 3 shows results analysing the number of clusters for the k-Means clustering with a centroid distance evaluation measure. The high number of different clusters came to our surprise. Clearly, a "soft" category approach with only a handful of clusters as sketched above cannot work out to capture these similarities satisfyingly. Moreover, after a brief manual inspection of the gained clusters, we can intuitively reason about basic correlations, such as nodes from same domains are often found in same clusters. This underlines the appropriateness of the applied methodology and motivates for further work in this direction towards actual correlation analysis. The gained clustering results provide a useful basis for these upcoming tasks.

We also tried to perform an agglomerative clustering. Unfortunately, we could not gain any results with that due to performance issues. Regarding the actually small sample we used, this highlights the need for particularly scalable methods in the context of the huge LOD graph.

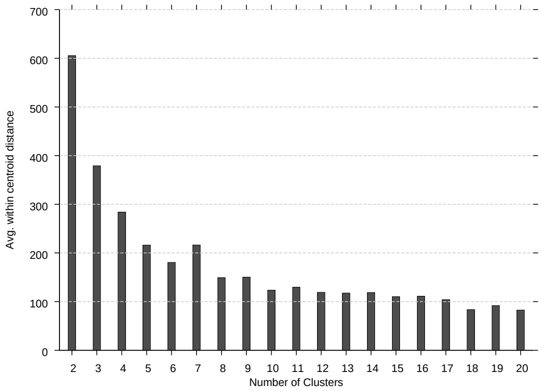

Figure 3: Number of clusters and average centroid distance.

5 Conclusion & Future Work

In this work, we presented our motivation and first ideas to study the dataset dynamics of the LOD Web using machine learning and data-mining approaches. We strongly believe that we can encompass comprehensive details about attributes and features that trigger or relate changes of LOD resources. The gained knowledge can eventually be integrated into a wide range of Web-related applications. In an first attempt, we identified how we can model the time series of changes and analyse its spectrum. Preliminary clustering results indicate the appropriateness of this approach and provide interesting first insights. We hope that this initial report triggers a rich discussion about suited methods and promising approaches in the community.

There exist many directions for future work. Clearly, we need a longer history of changes and thus we will continue our monitoring approach over the next years. Further, we will enrich our change matrix with more information. Possible information includes the incoming labelled links for the nodes, the node state changes of the one hop surrounding nodes and more features about the change event (type and fraction of change). Our future investigations include the following directions:

Change correlations We will investigate siophisticated methods to identify correlations between the dynamics of nodes. Especially, data reduction techniques such as PCA or SVD are of high interest to us and we will explore how and to which extend we can apply them.

Change classification Another area of high relevance for future work is to classify URIs into classes of dynamics. We are particularly interested to find the best features for the classification and we consider to use the outcome of the correlation analysis to increase the classification quality.

Change prediction Knowing in advance at which time in the future a resource is very likely to change is of tremendous value for our use case. Thus, we will explore machine-learning methods to compute the most likely change time based on observed change events. The quality of our predictions can also serve as an evaluation measure of the described approach.

Dealing with incomplete history Eventually, we will explore methods to deal with incomplete change histories. It is hard to monitor a resource constantly over a long time period. In a real world setup, a system will never be able to get a continuous history of change events of a resource. For instance, the window intervals might be too large and multiple changes can happen between to snapshots. In such a case it is of high importance to be able to handle these missing events and still predict and classify accurately.

Acknowledgements

This work based upon works jointly supported by the Science Foundation Ireland under Grant No. SFI/08/CE/I1380 (Lion-2) and under Grant No. 08/SRC/I1407 (Clique: Graph & Network Analysis Cluster) and by the European Community 7th framework ICT-2007.4.4 (No 231519) "e-Lico: An e-Laboratory for Interdisciplinary Collaborative Research in Data Mining and Data-Intensive Science".

References

[Berners-Lee, 2006] Tim Berners-Lee. Linked data, July 2006. `http://www.w3.org/DesignIssues/LinkedData`.

[Cho and Garcia-Molina, 2000] Junghoo Cho and Hector Garcia-Molina. The evolution of the web and implications for an incremental crawler. In *VLDB*, pages 200–209, 2000.

[Cho and Garcia-Molina, 2003] Junghoo Cho and Hector Garcia-Molina. Estimating frequency of change. *ACM Trans. Internet Techn.*, 3(3):256–290, 2003.

[Harth *et al.*, 2010] Andreas Harth, Katja Hose, Marcel Karnstedt, Axel Polleres, Kai-Uwe Sattler, and Jürgen Umbrich. Data summaries for on-demand queries over linked data. In *World Wide Web Conference (WWW '10)*, pages 411–420, 2010.

[Haslhofer and Popitsch, 2009] B. Haslhofer and N. Popitsch. DSNnotify - detecting and fixing broken links in linked data sets. In *DEXA '09 Workshop on Web Semantics (WebS '09)*, 2009.

[Manola and Miller, 2004] Frank Manola and Eric Miller. RDF Primer. W3C Recommendation, February 2004. http://www.w3.org/TR/rdf-primer/.

[Pandey *et al.*, 2003] Sandeep Pandey, Krithi Ramamritham, and Soumen Chakrabarti. Monitoring the dynamic web to respond to continuous queries. In *World Wide Web Conference (WWW '03)*, pages 659–668, 2003.

[Umbrich *et al.*, 2010] Jürgen Umbrich, Michael Hausenblas, Aidan Hogan, Axel Polleres, and Stefan Decker. Towards dataset dynamics: Change frequency of linked open data sources. In *WWW '10 Workshop on Linked Data on the Web (LDOW '10)*, 2010.