Hedera: Scalable Indexing and Exploring Entities in Wikipedia Revision History

Tuan Tran and Tu Ngoc Nguyen

L3S Research Center / Leibniz Universität Hannover, Germany
{ttran, tunguyen}@L3S.de

Abstract. Much of work in semantic web relying on Wikipedia as the main source of knowledge often work on static snapshots of the dataset. The full history of Wikipedia revisions, while contains much more useful information, is still difficult to access due to its exceptional volume. To enable further research on this collection, we developed a tool, named *Hedera*, that efficiently extracts semantic information from Wikipedia revision history datasets. Hedera exploits Map-Reduce paradigm to achieve rapid extraction, it is able to handle one entire Wikipedia articles' revision history within a day in a medium-scale cluster, and supports flexible data structures for various kinds of semantic web study.

1 Introduction

For over decades, Wikipedia has become a backbone of sematic Web research, with the proliferation of high-quality big knowledge bases (KBs) such as DBpedia [1], where information is derived from various Wikipedia public collections. Existing approaches often rely on one offline snapshot of datasets, they treat knowledge as static and ignore the temporal evolution of information in Wikipedia. When for instance a fact changes (e.g. death of a celebrity) or entities themselves evolve, they can only be reflected in the next version of the knowledge bases (typically extracted fresh from a newer Wikipedia dump). This undesirable quality of KBs make them unable to capture temporally dynamical relationship latent among revisions of the encyclopedia (e.g., *participate* together in complex events), which are difficult to detect in one single Wikipedia snapshot. Furthermore, applications relying on obsolete facts might fail to reason under new contexts (e.g. question answering systems for recent real-world incidents), because they were not captured in the KBs. In order to complement these temporal aspects, the whole Wikipedia revision history should be well-exploited. However, such longitudinal analytics over ernoumous size of Wikipedia require huge computation. In this work, we develop *Hedera*, a large-scale framework that supports processing, indexing and visualising Wikipedia revision history. Hedera is an end-to-end system that works directly with the raw dataset, processes them to streaming data, and incrementally indexes and visualizes the information of entities registered in the KBs in a dynamic fashion. In contrast to existing work that handle the dataset in centralized settings [2], Hedera employs the Map-Reduce paradigm to achieve the scalable performance, which is able to transfer raw data of 2.5 year revision history of 1 million entities into full-text index within a few hours in an 8-node cluster. We open-sourced Hedera to facilitate further research [1].

2 Extracting and Indexing Entities

2.1 Preprocessing Dataset

Here we describe the Hedera architecture and workflow. As shown in Figure 1, the core data input of Hedera is a Wikipedia Revision history dump [2]. Hedera currently works with the raw XML dumps, it supports accessing and extracting information directly from compressed files. Hedera makes use heavily the Hadoop framework. The preprocessor is responsible for re-partitioning the raw files into independent units (a.k.a *InputSplit* in Hadoop) depending on users' need. There are two levels of partitioning: Entity-wise and Document-wise. Entity-wise partitioning guarantees that revisions belonging to the same entity are sent to one computing node, while document-wise sends content of revisions arbitrarily to any node, and keeps track in each revision the reference to its preceding ones for future usage in the Map-Reduce level. The preprocessor accepts user-defined low-level filters (for instance, only partition articles, or revisions within 2011 and 2012), as well as list of entity identifiers from a knowledge base to limit to. If filtered by the knowledge base, users must provide methods to verify one revision against the map of entities (for instance, using Wikipedia-derived URL of entities). The results are Hadoop file splits, in the XML or JSON formats.

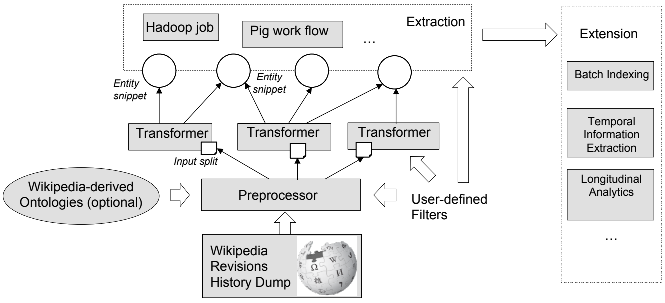


Fig. 1. Hedera Framework Architecture

2.2 Extracting Information

Before extracted in the Map-Reduce phase (Extraction component in Figure 1), file splits outputed from the preprocessor are streamed into a *Transformer*. The main goal of the transformer is to consume the files and emits (*key*,*value*) pairs suitable for inputting into one Map function. Hedera provides several classes of transformer, each of which implements one operator specified in the extraction

[1] Project documentation and code can be found at: `https://github.com/antoine-tran/Hedera`

[2] `http://dumps.wikimedia.org`

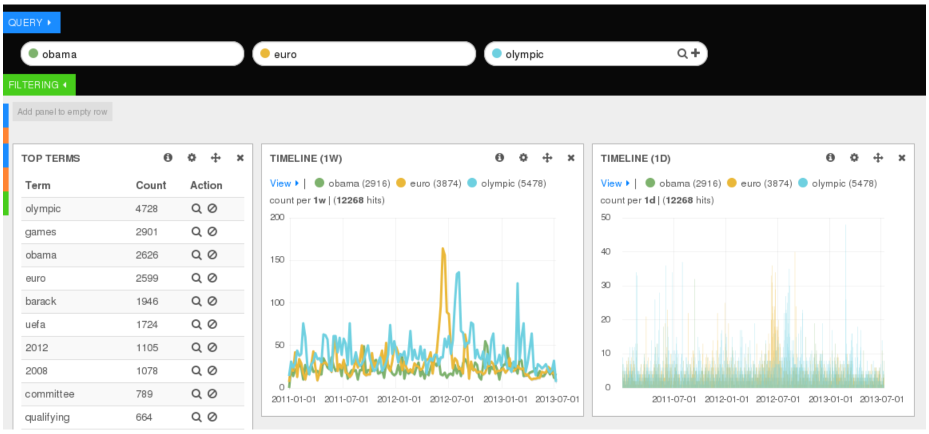
layer. Pushing down these operators into transformers reduces significantly the volume of text sent around the network. The extraction layer enables users to write extraction logic in high-level programming languages such as Java or Pig [3], which can be used in other applications. The extraction layer also accepts user-defined filters, allowing user to extract and index different portions of the same partitions at different time. For instance, the user can choose to first filter and partition Wikipedia articles published in 2012; and later she can sample, from one partition, the revisions about people published in May 2012. This flexibility facilitates rapid development of research-style prototypes in Wikipedia revision dataset, which is one of our major contributions.

3 Indexing and Exploring Entity-based Evolutions in Wikipedia

In this section, we illustrate the use of Hedera in one application - incremental indexing and visualizing Wikipedia revision history. Indexing large-scale longitudinal data collections i.e., the Wikipedia history is not a straightforward problem. Challenges in finding a scalable data structure and distributed storage that can most exploit data along the time dimension are still not fully addressed. In Hedera, we present a distributed approach in which the collection is processed and thereafter the indexing is parallelized using the Map-Reduce paradigm. This approach (that is based on the document-based data structure of ElasticSearch) can be considered as a baseline for further optimizations. The index's schema is loosely structured, which allows flexible update and incremental indexing of new revisions (that is of necessity for the evolving Wikipedia history collection). Our preliminary evaluation showed that this approach outperformed the well-known centralized indexing method provided by [2]. The time processing (indexing) gap is exponentially magnified along with the increase of data volume. In addition, we also evaluated the querying time (and experienced the similar result) of the system. We describe how the temporal index facilitate large-scale analytics on the semantic-ness of Wikipedia with some case studies. The detail of the experiment is described below.

We extract 933,837 entities registered in DBpedia, each of which correspond to one Wikipedia article. The time interval spans from 1 Jan 2011 to 13 July 2013, containing 26,067,419 revisions, amounting for 601 GBytes of text in uncompressed format. The data is processed and re-partitioned using Hedera before being passed out and indexed into ElasticSearch [4] (a distributed real-time indexing framework that supports data at large scale) using Map-Reduce. Figure 2 illustrates one toy example of analysing the temporal dynamics of entities in Wikipedia. Here we aggregate the results for three distinct entity queries, i.e., *obama*, *euro* and *olympic* on the temporal *anchor-text* (a visible text on a hyperlink between two Wikipedia revision) index. The left-most table shows the top terms appear in the returned results, whereas the two timeline graphs illustrate the dynamic evolvement of the entities over the studied time period (with

[3] `http://pig.apache.org`
[4] `http://www.elasticsearch.org`



TOP TERMS

Term	Count	Action
olympic	4728	🔍 ⊘
games	2901	🔍 ⊘
obama	2626	🔍 ⊘
euro	2599	🔍 ⊘
barack	1946	🔍 ⊘
uefa	1724	🔍 ⊘
2012	1105	🔍 ⊘
2008	1078	🔍 ⊘
committee	789	🔍 ⊘
qualifying	664	🔍 ⊘

Fig. 2. Exploring Entity Structure Dynamics Over Time

1-week and 1-day granuality, from left to right respectively). As easily observed, the three entities peak at the time where a related event happens (Euro 2012 for *euro*, US Presidential Election for *obama* and the Summer and Winter Olympics for *olympic*). This further shows the value of temporal anchor text in mining the Wikipedia entity dynamics. We analogously experimented on the Wikipedia *full-text* index. Here we brought up a case study of the entity co-occurrance (or temporal relationship) (i.e., between Usain Bolt and Mo Farah), where the two co-peak in the time of Summer Olympics 2012, one big tournament where the two atheletes together participated. These examples demonstrate the value of our temporal Wikipedia indexes for temporal semantic research challenges.

4 Conclusions and Future Work

In this paper, we introduced Hedera, our ongoing work in supporting flexible and efficient access to Wikipedia revision history dataset. Hedera can work directly with raw data in the low-level, it uses Map-Reduce to achieve the high-performance computation. We open-source Hedera for future use in research communities, and believe our system is the first in public of this kind. Future work includes deeper integration with knowledge bases, with more API and services to access the extraction layer more flexibly.

Acknowledgements

This work is partially funded by the FP7 project ForgetIT (under grant No. 600826) and the ERC Advanced Grant ALEXANDRIA (under grant No. 339233).

References

1. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. *DBpedia: A nucleus for a web of open data.* Springer, 2007.
2. O. Ferschke, T. Zesch, and I. Gurevych. Wikipedia revision toolkit: efficiently accessing wikipedia's edit history. In *HLT*, pages 97–102, 2011.