WikiQuery: A comprehensive solution for knowledge base construction and multimodal query

Hongtao Lin, Lisheng Wu
800 Dongchuan Rd. Shanghai Jiao Tong University
{Linkin_HearT, YourEmail}@sjtu.edu.cn

Abstract

Knowledge base has become a trending focus in both field of database system and natural language processing (NLP). In this report, we first analyzed the data structure in Wikidata, one of the largest open-source knowledge base available. Based on several insights on data, we constructed a minimal database for general queries. To further speed up the performance, we proposed an efficient schema targeted for specific queries. Experiments showed our design made a 40% speed up compared to baseline. Also, we designed a basic QA web system and proved the promising future for knowledge-based QA system.

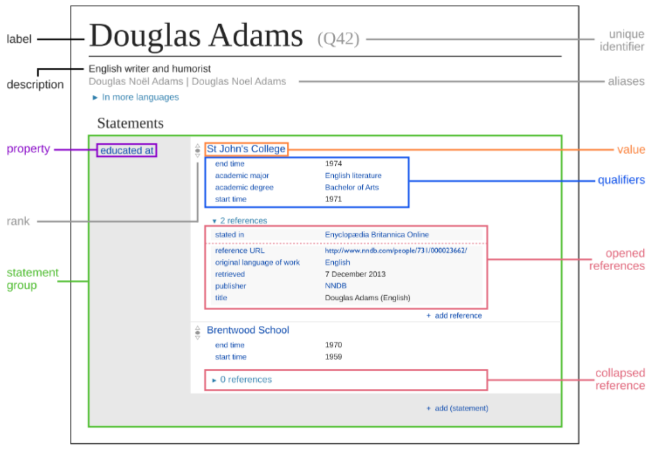
Introduction

Since the prevalence of computer science, researchers has long sought for ways to store human knowledge into computer, making them as "knowledgeable" as our human beings. That's the very origin of knowledge graph. van de Riet and Meersman (van de Riet 1992), Stokman and de Vries (Stokman 1988), and Zhang (Zhang 2002), present a formal theory of knowledge graphs in terms of semantic networks, in which meaning is expressed as structure, statements are unambiguous, and a limited set of relation types are used. Following up, several definitions and variation of knowledge graph are proposed. (Corby 2010) focus on the structural properties (narrow down the graph as a directed labeled graph). (Pujara 2013) focus on using probabilistic soft logic (PSL) to manage uncertainty in knowledge graphs that have been extracted from uncertain sources.

Besides storing and querying knowledge graph, constructing a knowledge graph is also a heated topic. Currently, most databases are edited and maintained by human, such as Freebase and Wikidata. There are also knowledges extracted from large-scale, semi-structured web knowledge bases such as Wikipedia, like DBpedia and YAGO. Furthermore, information extraction methods for unstructured or semi-structured information are proposed, which lead to knowledge graphs like NELL, PROSPERA, or KnowledgeVault.

For our project, we select Wikidata as the main resource, since it's well-defined and easy to access. Based on the data source, we perform database construction, optimization, query design on it.

The report is organized as follows: Section 2 gives an overview on the statistics and some example of data stored in Wikidata; Section 3 considers some optimization methods in database construction, based on the observations of data and demand from queries; Section 4 investigated several optimizations in terms on specific queries; Section 5 states about how we deal with natural language queries. Section 6 and 7 evaluate the overall system and suggest some future directions to go.



Douglas Adams (Q42)
English writer and humorist
Douglas Noël Adams | Douglas Noel Adams
▸ In more languages

Statements

educated at

St John's College

end time	1974
academic major	English literature
academic degree	Bachelor of Arts
start time	1971

▾ 2 references

stated in	Encyclopædia Britannica Online
reference URL	http://www.nndb.com/people/731/000023662/
original language of work	English
retrieved	7 December 2013
publisher	NNDB
title	Douglas Adams (English)

+ add reference

Brentwood School

end time	1970
start time	1959

▸ 0 references

+ add (statement)

Wikidata Overview

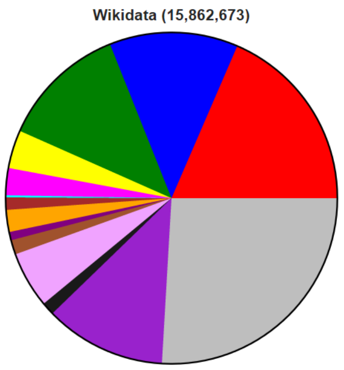
Wikidata is a collaboratively edited knowledge graph, with edit permission to both human and machines. It's operated by the Wikimedia foundation, which also hosts the various language editions of Wikipedia. After the shutdown of Freebase, the data contained in Freebase is subsequently moved to Wikidata. A special highlight for Wikidata is that for each entity, provenance metadata like references and qualifiers can be included.

As illustrated above, an entity in Wikidata has the commonly used label, description and aliases, associated with several statements. Besides, for each statements, it may also have qualifiers (education start time) and references (the statistics is stated in a trackable website). These information are the unique advantages for Wikidata as a complete and convincing source of knowledge graph.

According to the official website, Wikidata has just announced its fourth year celebration, with over 24 million entities and more than 2,000 properties in it.

Database Design

Wikidata is no doubt one of the largest knowledge base available on the web. But the question that comes after is: Do we need to store all of them so as to make a knowledge graph? The answer is definitely no, which we will illustrated below.



According to our investigation on all properties, we found that more than 80% of them are used by a small portion of statements (small means less than 5000). Besides, according to the official report released in Oct. 2015 (See above), over 25% of the entities does not have a subclass/instance of as its property (Illustrated as gray), which means it may be isolated and less informative.

Based on the above analysis, we choose to filter out only high frequency properties and entities without subclass property. Specifically, we rank the properties according to their "usage", i.e., the number of statement related to it, and filter out those with more than 4,000 usages. Also, since some of the properties with type "external-id" would be uninformative in querying, we decide to filter them out. The above process left only 12% of properties, which is around 300, a significant decrease in number.

On the other hand, we consider our schema from perspective of designed queries. Specifically, we observed the following 3 insights, and made adjustments accordingly:

1. We found the schema "references" to be useless in such a condition, since no query asked to illustrate the resource for statement. Thus we choose to discard it in database we actually stored. Same reason is also applicable for "sitelink" and "badge".
2. The original JSON file classifies datavalues into 6 value types, and assigned different structures to them. Our queries, however, do not ask for values with such details (e.g. the precision for a measurement). Also, design datavalues with different structure would require same number of schemas which may yield inefficiencies in it. Based on the above two reasons, we choose to concatenate the information of a value into a human-friendly format. Take the "globecoordinate" for example, we may only store a string value representing latitude and longitude of a place. That's enough for querying.
3. For "label", "description" and "alias", each is associated with a huge amount of languages. But we may only care about Chinese and English versions. So we choose to only store the above data if it's in Chinese or English.

As discussed above, we constructed a *simplified* version of Wikidata: [ER-Model]. This schema is more like a standard triple representation of knowledge: (subject, predicate, object) formed in most knowledge graph system. Besides this, our designation also support qualifiers that further constraint on the triple.

Note that we also built scripts that can store the *complete* information in the dumped JSON file, which is included and configurable within script. The complete E-R model is included at the end of report.

The schema refinement leads to a huge deduction in time spent, from 15min per 10,000 entities to 40s.

[Add: anything you wanna say about storage type, foreign key???]

Query Optimization

The above constructed schema would serve as a general purpose knowledge graph without much specifications. For the designed queries, we can perform ever better by building individual optimizations.

[Better add some experiment results and theoretical explanations in it!]

Knowledge-based QA

There have been increasing interest on knowledge-based question answering (KB-QA) system. Starting from template-based QA system around 1990s, we have arrived at the era of deep learning, with substantial improvement on KB-QA (Yih 2015).

Due to limitation in time and resource, however, we turned to simple and effective template-based QA system. We expected our system to query a wide range of answers, which is categorized as two types.

The first type is the values in statements. We emphasize more on the query of object value. We start from 300 selected properties described in Section 2, and articulated several templates target for this property. Supposed we want to match queries with properties of “date of birth”, we can use regular expression like “when (is|was) the birthday of ($entity)”. The above expression will extract a half-complete tripe: ($entity, “date of birth”, None), and thus we can query the “claim” table for answer.

One thing to note is that we may encounter conditioned queries. Questions like “What’s the population of China?” is ambiguous because we do not consider the specific year. To deal with this kind of queries, we may need the information about qualifier, which is on our to-do list.

The second type is the description of a subject. Beside querying on triples, people are also interested in knowing the definitions of a subject. This type of question is often highly organized (e.g.: “Who is Obama?”). Thus we can easily extract the query from the template.

These templated were integrated in the “natural language querying” part of our web application. For more detailed documentation, please turn to our demo website.

Future Work

We believe this piece of work is a solid starting point for future explorations. There may be a lost future work can be considered in the following two parts:

Firstly, on issues of efficiency in database design. Despite simplifying schema, we are not satisfied with the speed of data import. We also believe there leaves much room for improvement: we can try using [Add:] … Also, the storage size can be further compressed by [Add.]

Beside, we can deliberate more into KB-QA system. A more systematic way to do KB-QA, which has once populated during 2000s, is to do semantic parsing first, retrieving a more structured information about the sentence. Then we can convert the sentence into first-order logic (λ-calculus) and directly query from standard knowledge base. This is a simple but effective direction to go.

Conclusion

In this report, we investigated the whole process of processing, constructing and querying on Wikidata, an open-source knowledge base.

References

RP van de Riet, RA Meersman. Knowledge Graphs. 97 (1992)

Frans N. Stokman, Pieter H. de Vries. Structuring Knowledge in a Graph. 186–206 (1988)

Lei Zhang. Knowledge graph theory and structural parsing. (2002)

Olivier Corby, Catherine Faron Zucker. The KGRAM Abstract Machine for Knowledge Graph Querying. (2010)

Jay Pujara, Hui Miao, Lise Getoor, William Cohen. Knowledge Graph Identification. 542–557 (2013)

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A Collaboratively Created Graph Database For Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, New York, 2008. ACM.

Denny Vrandecic and Markus Krötzsch. Wikidata: a Free ´ Collaborative Knowledge Base. *Communications of the ACM*, 57(10):78–85, 2014.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 6(2), 2013

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. YAGO: A Core of Semantic Knowledge Unifying WordNet and Wikipedia. In *16th international conference on World Wide Web*, pages 697–706, New York, 2007. ACM

Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. Coupled semisupervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110, New York, 2010. ACM.

Xin Luna Dong, K Murphy, E Gabrilovich, G Heitz, W Horn, N Lao, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge Vault: A Web-scale approach to probabilistic knowledge fusion. In *20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601– 610, New York, 2014. ACM

Yih, Wen-tau, et al. "Semantic parsing via staged query graph generation: Question answering with knowledge base." *Association for Computational Linguistics (ACL).* 2015.

Ndapandula Nakashole, Martin Theobald, and Gerhard Weikum. Scalable knowledge harvesting with high precision and high recall. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 227– 236, New York, 2011. ACM

This Is an Example Subsection Heading

The text below a second-level heading begins without indentation. This is example text. It is 10 point Times New Roman.

This is the second paragraph. It in formatted with the Text-indent style. This is example text. This is example text. It is 10 point Times New Roman. This is example text. It is 10 point Times New Roman.

This is a Subsubsection Heading

This is example text. It is 10 point Times New Roman. This is example text. It is 10 point Times New Roman.

This is example indented text. It is 10 point Times New Roman. This is example indented text. It is 10 point Times New Roman. This is example indented text. It is 10 point Times New Roman. This is example indented text. It is 10 point Times New Roman..

> This is an example of an extract or quotation. Note the indent on both sides. Quotation marks are not necessary if you offset the text in a block like this, and properly identify and cite the quotation in the text.

This Is an Example of a Figure Caption.