# Populating a Linked Data Entity Name System

**Mayank Kejriwal** (Information Sciences Institute; kejriwal@isi.edu)

DOI: 10.1145/3098888.3098897

Resource Description Framework (RDF) is a graph-based data model used to publish data as a *Web of Linked Data* (Bizer *et al.* 2009). RDF is an emergent foundation for large-scale *data integration*, the problem of providing a unified view over multiple data sources. The structure in RDF data can be conveniently visualized using *directed labeled graphs*, as illustrated in the real-world graph fragments in Figure 1. Nodes in the graph represent entities (e.g. the node with label *dbpedia:Allen_,Paul* represents the entity Paul Allen in the DBpedia knowledge graph) and edges represent either attributes of an entity (e.g. '01/21/1953' is the birthdate of Paul Allen) or relationships between two entities (e.g. Paul Allen is the co-founder of the company entity, Microsoft). Facts in the knowledge base are formally represented as a set of *triples*, with a triple comprising a labeled edge (denoted as a *property*) in the RDF graph along with its incoming and outgoing nodes.

An Entity Name System (ENS) is a thesaurus for entities, and is a crucial component in a data integration architecture (Kejriwal 2014). For example, consider an application that queries multiple knowledge graphs. Since entities like Microsoft and Paul Allen are represented in different ways in different graphs, a robust system would need to infer that the different *mentions* of a single entity should be linked to a *canonical* thesaurus entry. Automatically populating an ENS is equivalent to solving an AI problem called *Entity Resolution* (ER), which concerns identifying pairs of entities referring to the same underlying entity (Getoor and Machanavajjhala 2012).

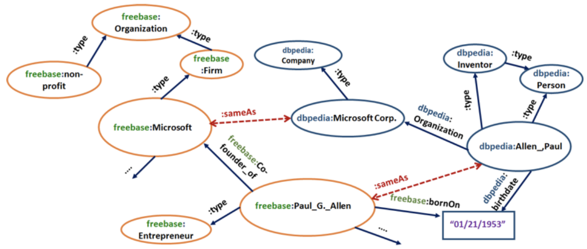


Figure 1: Fragments of real-world RDF graphs, DBpedia and Freebase, with two examples of *Entity Resolution* using a special *sameAs* property

Due to its Web origins, Linked Data exhibits properties that make ER a challenging problem including *scale* (super-linear growth on average since inception), *heterogeneity* (data is published using a wide range of RDF types and properties) and *diversity* (data spans multiple domains, e.g. social media and bioinformatics). Diversity, in particular, is an important concern for state-of-the-art *supervised* *machine learning*-based ER systems, since manually labeled examples for each domain are typically not available. Acquiring such examples is also hard since ER exhibits *class skew*: the number of synonym node pairs in a graph is much smaller than the number of non-synonym node pairs.

## Contributions

We propose an Entity Resolution (ER) approach that includes methods to accommodate the competing requirements of diversity, heterogeneity and scale that are necessary for enabling successful population of Linked Data Entity Name Systems. In the rest of this section, we present a high-level overview of the primary aspects of the approach, followed by a brief empirical summary.

Given the success of supervised machine learning approaches in the ER literature, we address diversity by developing a *self-training* machine learning algorithm that executes in two stages (Kejriwal and Miranker 2013). In the first stage, the algorithm generates its own training set by applying a set of inexpensive, non-adaptive similarity heuristics that are known to be robust in a variety of text domains. Because these heuristics are non-adaptive, they do not require user supervision other than the setting of a conservative threshold, which we also automated in later developments (Kejriwal and Miranker 2015b).

Although such heuristics cannot compete with supervised algorithms in 'difficult' real-world domains, we showed that they can be used to obtain 'easy' matches between nodes that have many attribute values in common (Kejriwal and Miranker 2013). Even with conservative thresholds, some of the matches are noisy and the obtained matches are quite sparse. In the second stage, therefore, the algorithm uses a number of strategies to robustly train a supervised machine learning classifier like an SVM. Strategies that were shown to achieve high success rates in our work include the use of an expressive set of features (spanning numeric, string and phonetic features), boosting and feature discretization (Kejriwal and Miranker 2015a).

To accommodate heterogeneity, we developed a *schema-matching algorithm* that not only matches *edge labels* between two graphs, but also determines *compatibly typed* nodes across the graphs. In Figure 1, for example, the algorithm would determine that *freebase:Microsoft* and *dbpedia:Microsoft Corp.* are compatibly typed (i.e. they are both companies) and should be classified, either as a match or non-match, by the self-trained SVM, whereas *freebase:Microsoft* and *dbpedia:Allen_,Paul* are incompatibly typed.

Finally, we accommodate scale *directly* in the design of all the algorithms described above. We show that an approximate version of the algorithms, particularly the training set generator, can be implemented in a shared-nothing parallel paradigm like MapReduce, and successfully handles *data skew*, an important concern in many data-intensive problems.

Empirically, we evaluate the efficacy of our methods by actively avoiding prior assumptions about input domains, and through evaluations on ten RDF test cases spanning multiple domains, including movies, books, people and restaurants. On all datasets, the approach outperforms a popular unsupervised baseline (derived from Locality Sensitive Hashing) by a large margin, and achieves performance (within $10\%$ on a popular accuracy measure) competitive with supervised SVM-based baselines (Kejriwal and Miranker 2015b).

We test the scalability of our approach by implementing our algorithms in both serial and MapReduce architectures. Evaluations in public cloud infrastructure (Hadoop clusters on Microsoft Azure) show that the system scales near-linearly, and is able to effectively resolve entities across encyclopedic graphs (e.g. DBpedia and Freebase) with millions of nodes and edges using relatively small clusters ($<$40 compute nodes).

## References

Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data-the story so far. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, pages 205–227, 2009.

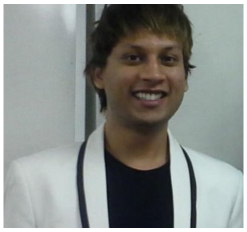
Lise Getoor and Ashwin Machanavajjhala. Entity resolution: theory, practice & open challenges. *Proceedings of the VLDB Endowment*, 5(12):2018–2019, 2012.

Mayank Kejriwal and Daniel P Miranker. An unsupervised algorithm for learning blocking schemes. In *2013 IEEE 13th International Conference on Data Mining*, pages 340–349. IEEE, 2013.

Mayank Kejriwal and Daniel P Miranker. Semi-supervised instance matching using boosted classifiers. In *European Semantic Web Conference*, pages 388–402. Springer, 2015.

Mayank Kejriwal and Daniel P Miranker. An unsupervised instance matcher for schema-free rdf data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35:102–123, 2015.

Mayank Kejriwal. Populating entity name systems for big data integration. In *International Semantic Web Conference*, pages 521–528. Springer, 2014.



**Mayank Kejriwal** completed his PhD at the University of Texas at Austin under Daniel P. Miranker. Currently, he works as a researcher and Computer Scientist in the Information Integration group at the Information Sciences Institute, part of the Viterbi School of Engineering at the University of Southern California in Los Angeles, California.