CoEPinKB: A Framework to Understand the Connectivity of Entity Pairs in Knowledge Bases

Javier Guillot Jiménez[1], **Luiz André P. Paes Leme**[2], **Marco A. Casanova**[1]

[1]Department of Informatics, PUC-Rio – Rio de Janeiro – RJ – Brazil

[2]Federal Fluminense University – Niteroi, RJ – Brazil

{jguillot,casanova}@inf.puc-rio.br, lapaesleme@ic.uff.br

Abstract. *A knowledge base, expressed using the Resource Description Framework (RDF), can be viewed as a graph whose nodes represent entities and whose edges denote relationships. The entity relatedness problem refers to the problem of discovering and understanding how two entities are related, directly or indirectly, that is, how they are connected by paths in a knowledge base. Strategies designed to solve the entity relatedness problem typically adopt an entity similarity measure to reduce the path search space and a path ranking measure to order and filter the list of paths returned. This paper presents a framework, called* CoEPinKB*, that supports the empirical evaluation of such strategies. The proposed framework allows combining entity similarity and path ranking measures to generate different path search strategies. The main goals of this paper are to describe the framework and present a performance evaluation of nine different path search strategies.*

1. Introduction

Knowledge bases, such as DBpedia [Lehmann et al. 2015], expressed using the RDF data model, can be viewed as graphs whose nodes represent entities and whose edges denote relationships. In this paper, we present a framework, called CoEPinKB, that supports exploring a knowledge base in order to discover and understand how two entities are connected. This is known as the *entity relatedness problem*, formalized as: "Given an RDF graph G and a pair of entities a and b, represented in G, compute the paths in G from a to b that best describe the connectivity between them".

Our proposal is based on [Talavera Herrera 2017], which introduced a two-step strategy to address the entity relatedness problem: (1) search for relationship paths between pairs of entities; and (2) rank the paths found and select those that are relevant. To address the first step, the author proposed a generic strategy based on the backward search heuristic [Le et al. 2014], which is a breadth-first search strategy that expands the paths starting from each input entity, in parallel, until a candidate relationship path is generated. The expansion process uses activation criteria to prioritize certain paths over others and to filter the entities less related to the target entities, so that it can be easier to identify more meaningful paths. These activation criteria give priority to entities with low degree in the graph and maintain entities that are similar to the last entity reached in a partially constructed path, using some similarity measure. The second step adopts ranking approaches that use the semantics of the relationships between the entities to assign a score to relationship paths. After sorting the set of relationship paths found in the first step, the top-k paths are selected to describe the connectivity of an entity pair.

The first contribution of this paper is the proposal and implementation of a framework that helps address the entity relatedness problem. Our framework differs from the implementation proposed in [Talavera Herrera 2017] in three aspects. First, CoEPInKB was designed to make it easy for developers to add new entity similarity and relationship path ranking measures to generate new path search strategies. Second, CoEPInKB has a simple and practical Web user interface that facilitates the interaction of the users with the framework, and provides an API that facilitates executing different experiments and analyze the results. Lastly, CoEPInKB was engineered to work with any knowledge base accessible using a SPARQL service over HTTP.

The analysis in [Talavera Herrera 2017] evaluated nine relationship path search strategies on two entertainment domains. However, the analysis did not evaluate the performance of these strategies. A second contribution of this paper is the use of CoEPInKB to evaluate the performance of these different strategies with regard to execution time.

The remainder of this paper is organized as follows. Section 2 briefly reviews related work. Section 3 presents the architecture and some technical aspects of the implementation of the proposed framework. Section 4 presents a performance evaluation of path search strategies, using CoEPInKB. Finally, Section 5 presents the conclusions and some directions for future work.

2. Related Work

REX [Fang et al. 2011] is a system that takes a pair of entities in a given knowledge base as input and identifies a ranked list of relationship paths, called by the authors as relationship explanations. It used two BFS on the RDF graph to enumerate relationship paths between two entities and considered the degree of a node as an activation criterion to prioritize nodes.

RECAP [Pirrò 2015], EXPLASS [Cheng et al. 2014] and DBpedia Profiler [Herrera et al. 2016] implemented path finding processes in an RDF knowledge graph with the help of SPARQL queries [Färber et al.]. Likewise, [Järvelin and Kekäläinen 2002] used the *Jaccard index* [Jaccard 1901] to compute an approximated minimal distance between the start and the end nodes, and to discover meaningful connection between the nodes.

Path-ranking measures were proposed in [Cheng et al. 2014, Herrera et al. 2016, Hulpuş et al. 2015, Pirrò 2015]. However, the evaluation methods did not clearly defined the capabilities of the approaches analyzed. The work proposed in [De Vocht et al. 2016] argued that entity similarity heuristics increase the relevance of the links between nodes. The authors compared and measured the effectiveness of different search strategies through user experiments. [Herrera et al. 2017] introduced a benchmark to evaluate path ranking measures.

In [Talavera Herrera 2017], the author introduces a generic search strategy, based on the backward search heuristic, to prioritize certain paths over others. The strategy combines similarity measures such as the *Jaccard index*, the *Wikipedia Link-based Measure* (WLM) [Milne and Witten 2008], and *SimRank* [Jeh and Widom 2002], and ranking measures such as the *Predicate Frequency Inverse Triple Frequency* (PF-ITF) [Pirrò 2015], the *Exclusivity-based Relatedness* (EBR) [Hulpuş et al. 2015], and the *Pointwise Mutual*

Information (PMI) [Church and Hanks 1990]. This work lacks an evaluation of the performance, in terms of execution time, of each of the different path search strategies, as well as a tool with a graphical user interface that facilitates the interaction of users who intend to evaluate these strategies. The present work aims to fill this gap, as it will be described in the following sections.

3. The CoEPinKB Framework

The acronym CoEPinKB stands for understanding the **C**onnectivity **o**f **E**ntity **P**airs **in** **K**nowledge **B**ases. As we mentioned earlier, our approach to address the entity relatedness problem is to apply a two-step strategy, and each of these two sequential phases corresponds to the main components of the framework: the BACKWARD SEARCH component, which executes a breadth-first search starting from each input entity and expanding similar entities to find the most relevant relationship paths; and the RELATIONSHIP PATH RANKING component, which ranks the resulting paths of the previous step. Figure 1 shows an overview of the architecture of CoEPinKB.

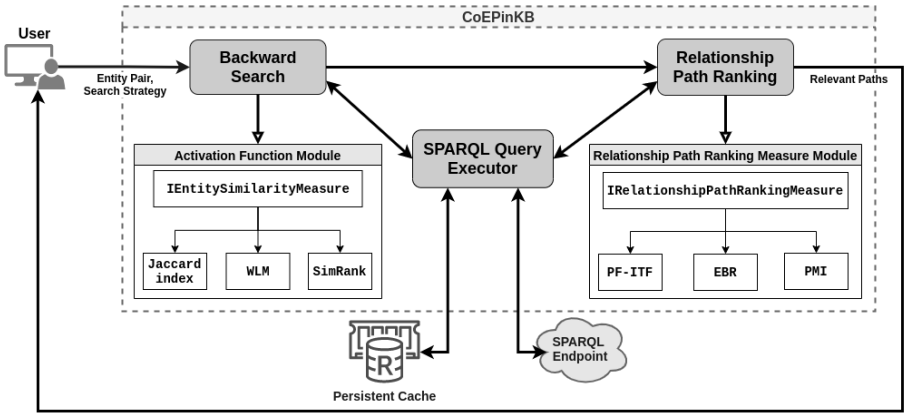


Figure 1. CoEPinKB architecture

The CoEPinKB framework takes as input a pair of entities and a search strategy. A search strategy consists of an entity similarity measure that will be used by the backward search algorithm as the activation function to decide when to expand some neighbor of an entity or not, and a relationship path ranking measure to select the top-k relevant paths between the two entities provided.

During the first phase of the execution of CoEPinKB, the BACKWARD SEARCH component communicates with the SPARQL QUERY EXECUTOR component requesting the required data to execute the backward search algorithm. This last component gets the requested data using two different approaches: (i) first, it tries to get the data from the persistent cache; (ii) if the requested data is not available then it gets the data directly from the SPARQL Endpoint through SPARQL queries, and stores it in the persistent cache to speed up future searches. After the backward search algorithm finishes, the BACKWARD SEARCH component sends a list of relationship paths between the pair of entities to the RELATIONSHIP PATH RANKING component. Then, the second phase begins. Similarly to the previous phase, the RELATIONSHIP PATH RANKING component communicates with the SPARQL QUERY EXECUTOR component requesting the required data to execute the

path ranking algorithm. After the algorithm finishes, the RELATIONSHIP PATH RANKING component sends the list of ranked paths to the user through the user interface.

There are two key flexibility points in the framework - the activation function, implementing the entity similarity measure, and the path ranking measure - which are the core of the BACKWARD SEARCH and RELATIONSHIP PATH RANKING components. These components were designed using an architectural pattern based on interfaces, which increases the extensibility of the framework by making it easier to add new entity similarity measures and relationship path ranking measures. As illustrated in Figure 1, the current version of COEPINKB implements 3 entity similarity measures (*Jaccard index*, WLM, and *SimRank*) and 3 relationship path ranking measures (PF-ITF, EBR, and PMI).

At the data layer, the framework has the SPARQL QUERY EXECUTOR component that interacts with RDF datasets through their SPARQL endpoints. The framework also uses a persistent cache to store the result of the SPARQL queries executed during the expansion of the entities in the RDF graph. The main reason for this decision is that the backward search and the relationship-path ranking algorithms require executing a large number of queries (quite possibly over the network), which can negatively affect the overall performance of the framework.

Figure 2 shows the COEPINKB User Interface and an excerpt of the result when the input entities are `dbr:Michael_Jackson` and `dbr:Whitney_Houston`, when the entity similarity and relationship path ranking measures are *Jaccard index* and EBR, respectively.

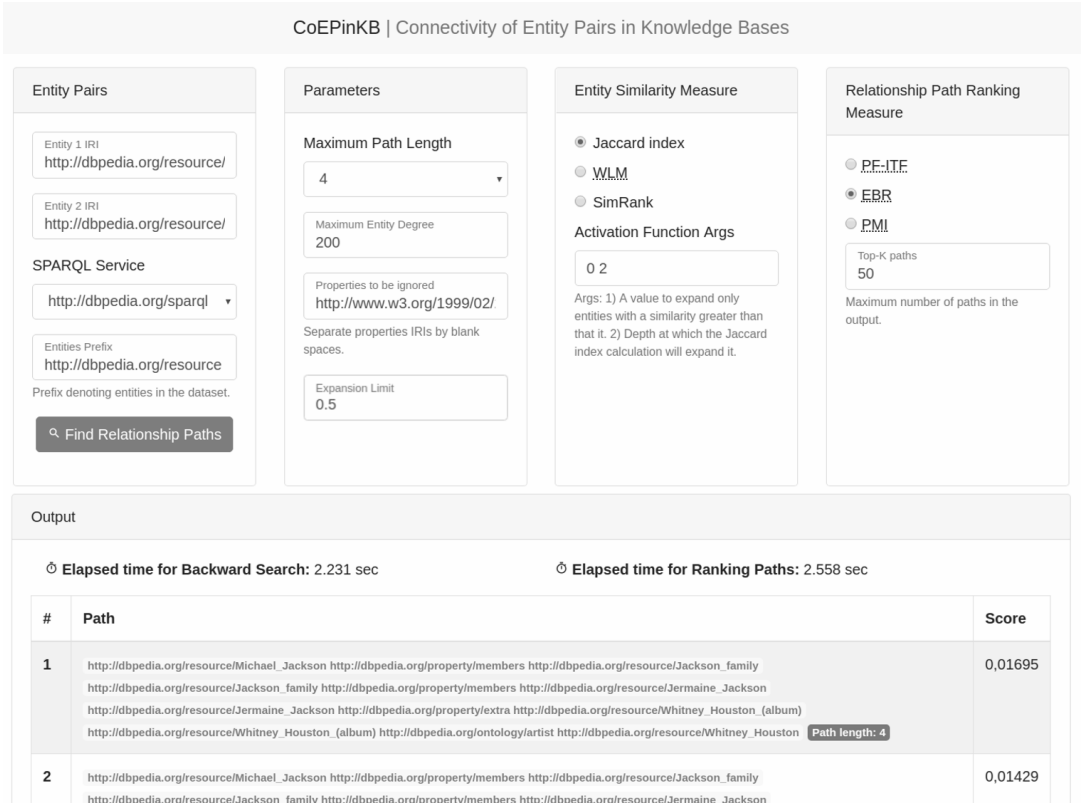


Figure 2. COEPINKB UI

The user also specifies other parameters such as: the maximum path length between the entities (set to 4 by default); the maximum entity degree, in order to discard entities with a high number of neighbors during the expansion; a list of properties irrelevant when building the relationship paths; an entity prefix, to expand only to resources that are considered entities; an expansion limit $\lambda \in [0, 1]$, understood as a percentage, that limits the expansion process; and the maximum number of paths that the user wants. CoEPinKB also provides a RESTful API, so the user can submit a GET request that returns a JSON document containing the corresponding list of relevant paths between the two entities.

The CoEPinKB framework was implemented in Java with the help of some other technologies, such as: Apache Jena, to interact with the RDF data sources; Redis, as our persistent cache; and the Jedis library, which allowed us to interact with a Redis instance from our Java application.

4. Evaluation

In [Talavera Herrera 2017], some experiments were executed to evaluate a family of path search strategies (Table 1) against a benchmark [Herrera et al. 2017] from the music and movies domains, and a baseline, RECAP [Pirrò 2015].

Table 1. Path Search Strategies

#	Abbr	Name
1	J&I	*Jaccard index* & PF-ITF
2	J&E	*Jaccard index* & EBR
3	J&P	*Jaccard index* & PMI
4	W&I	WLM & PF-ITF
5	W&E	WLM & EBR
6	W&P	WLM & PMI
7	S&I	*SimRank* & PF-ITF
8	S&E	*SimRank* & EBR
9	S&P	*SimRank* & PMI

For each domain, the benchmark contains 20 pairs of entities, each with a ranked list with 50 relationship paths based on information about their entities found in IMDb and last.fm, and on information about their properties, computed from DBpedia. The pairwise comparison method was used to identify the path search strategy that achieves the best performance on the benchmark, and to compare the best strategy identified with the baseline. These experiments suggested that the J&E strategy, which adopts the *Jaccard index* and the EBR measure, is the best of the 9 strategies compared and obtained better results than the baselines.

To evaluate the performance of the 9 different path search strategies we used CoEPinKB on a server with an Intel® Core™ i7-5820K CPU @ 3.30GHz and 6GB of memory dedicated to Java applications. All experiments were carried out over DBpedia through the OpenLink Virtuoso SPARQL protocol endpoint. We selected 10 entity pairs from the Entity Relatedness Test Dataset [Herrera et al. 2017], 5 pairs for each domain (music and movies), shown in Table 2.

Table 2. Entity pairs from music and movies domains

#	Entity Pair (music domain)	#	Entity Pair (movie domain)
1	Michael_Jackson, Withney_Houston	6	Elizabeth_Taylor, Richard_Burton
2	The_Beatles, The_Rolling_Stones	7	Cary_Grant, Katharine_Hepburn
3	Elton_John, George_Michael	8	Laurence_Olivier, Ralph_Richardson
4	Led_Zeppelin, The_Who	9	Errol_Flynn, Olivia_de_Havilland
5	Pink_Floyd, David_Gilmour	10	William_Powell, Myrna_Loy

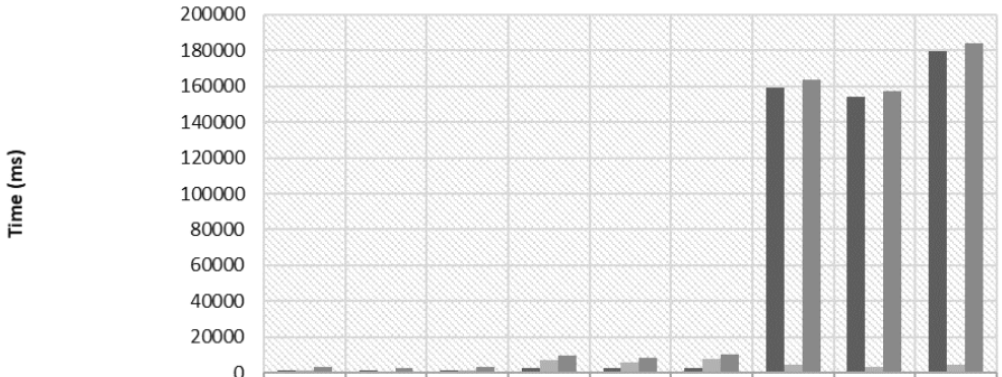
The other parameters were configured as follows:

- The maximum path length between the entities was set to 4, since this was the limit adopted by previous works, as RECAP [Pirrò 2015] and EXPLASS [Cheng et al. 2014], and verified experimentally in [Järvelin and Kekäläinen 2002].
- The maximum entity degree was set to 200. This degree limit was deduced from DBpedia statistics[1], which indicate that 90% of the entities have less than 200 links. This kind of criterion is applied together with entity similarity because, as in [Moore et al. 2012], because it can be assumed that nodes with high degree influence the path search process with potentially very unspecific information.
- The expansion limit was set to $\lambda = 0.5$. So, the adjacency list of each entity is sorted by similarity and only the top 50% of the entities are considered, independently of the size of the list and the similarity scores. We considered 50% of the list because it is a moderate factor to maintain the connectivity between entities and propagate the similarity score in the graph [Sommer 2014].
- A set of properties were ignored during the exploration of the knowledge base because many of these properties describe relationships between entities that are irrelevant for our analysis. For instance, if we considered properties like `http://www.w3.org/1999/02/22-rdf-syntax-ns#type` and `http://www.w3.org/1999/02/22-rdf-syntax-ns#type`, we would have to deal with many paths that are irrelevant for our analysis. There are more than 225 statements in which the subject is the entity `dbr:Michael_Jackson` and the predicate is one of these properties. The property `http://dbpedia.org/ontology/wikiPageRedirects` is also present in many statements (almost 70 times in the case that the entity `dbr:Michael_Jackson` is the object) that mainly link entities with typographical errors or other types of minor errors with the corresponding correct entity.
- The entity prefix was set to `http://dbpedia.org/resource`.
- The maximum number of paths was set to 50, because this value suffices to explore the connectivity between the entities, as reported in [Cheng et al. 2014, Fang et al. 2011, Hulpuş et al. 2015, Pirrò 2015].

For each pair of entities, we searched the top-k relationship paths between them six times (we excluded the first cold start run time, to avoid the warm-up bias) and calculated the average time of the last five executions of the program. Figure 3 shows

[1] http://downloads.dbpedia.org/2015-04/ext/pagerank/

the performance evaluation results. The best path search strategies in terms of performance are: J&E (2558ms), J&I (3049ms) and J&P (3241ms). The experiments in [Talavera Herrera 2017] indicated that J&E and W&E perform better than the others strategies in terms of finding the relevant paths between a pair of entities in the music and movies domains, and also that the J&E strategy performs better than the baselines. Therefore, we may conclude that J&E is the fastest and performs better than the others strategies.



	J&I	J&E	J&P	W&I	W&E	W&P	S&I	S&E	S&P
Avg. time for searching paths	1708	1664	1678	2594	2616	2634	159396	154131	179467
Avg. time for ranking paths	1341	894	1563	7216	5608	7977	4278	3251	4770
Avg. total time	3049	2558	3241	9810	8224	10611	163674	157382	184237

Figure 3. Evaluation results on performance of each path search strategy

The experiments reflect the particularity of how each of the entity similarity and path ranking measures are calculated. The average times for the strategies using the *Jaccard index* or the WLM were quite good and very similar because both entity similarity measures use the features sets A_d and B_d, which are stored and quickly accessed in our persistent cache (recall that $d = 2$). By contrast, the strategies that use *SimRank* have a poor performance due to its recursive definition. In fact, due to the computational complexity of *SimRank*, there are many studies to speed up such calculations [Lizorkin and Velikhov 2008, Li et al. 2010, Reyhani Hamedani and Kim 2021]. As for the path ranking measures, EBR executes fewer calculations than PF-ITF and PMI. For this reason, the average time for ranking paths using EBR is better than the average time using PF-ITF and PMI.

The conclusion is that the path search strategy that adopts the *Jaccard index* and the EBR measures achieves the best performance in both domains of the benchmark, and performs better than RECAP. With regard to execution time, it was verified that the most effective strategies are also the fastest ones.

5. Conclusions

In this paper, we introduced CoEPInKB, a framework that allows empirically evaluating path search strategies that combine entity similarity and path ranking measures, in order to understand the connectivity of entity pairs in RDF datasets. CoEPInKB supports such evaluation by featuring two flexibility points: the entity similarity and the path ranking measures. Also, CoEPInKB was engineered to work with any knowledge base accessible using a SPARQL service over HTTP. Our performance evaluation of the path search

strategies indicated that any strategy that uses *SimRank* as activation function has a poor performance, when compared with the other strategies. We also verified that the most effective strategies are also the fastest ones.

As future work, we plan to test the path search strategies in other knowledge bases, and to implement additional entity similarity and relationship path ranking measures. We also plan to develop a distributed version of the framework using Spark to improve performance.

References

Cheng, G., Zhang, Y., and Qu, Y. (2014). Explass: Exploring Associations between Entities via Top-K Ontological Patterns and Facets. In Mika, P., Tudorache, T., Bernstein, A., Welty, C., Knoblock, C., Vrandečić, D., Groth, P., Noy, N., Janowicz, K., and Goble, C., editors, *The Semantic Web – ISWC 2014*, volume 8797, pages 422–437. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Church, K. W. and Hanks, P. (1990). Word Association Norms, Mutual Information, and Lexicography. *Computational Linguistics*, 16(1):22–29.

De Vocht, L., Beecks, C., Verborgh, R., Mannens, E., Seidl, T., and Van de Walle, R. (2016). Effect of Heuristics on Serendipity in Path-Based Storytelling with Linked Data. In Yamamoto, S., editor, *Human Interface and the Management of Information: Information, Design and Interaction*, volume 9734, pages 238–251. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Fang, L., Sarma, A. D., Yu, C., and Bohannon, P. (2011). REX: explaining relationships between entity pairs. *Proceedings of the VLDB Endowment*, 5(3):241–252.

Färber, M., Ell, B., Menne, C., and Rettinger, A. A Comparative Survey of DBpedia, Freebase, OpenCyc, Wikidata, and YAGO. page 26.

Herrera, J. E. T., Casanova, M. A., Nunes, B. P., Leme, L. A. P. P., and Lopes, G. R. (2017). An Entity Relatedness Test Dataset. In d'Amato, C., Fernandez, M., Tamma, V., Lecue, F., Cudré-Mauroux, P., Sequeda, J., Lange, C., and Heflin, J., editors, *The Semantic Web – ISWC 2017*, volume 10588, pages 193–201. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Herrera, J. E. T., Casanova, M. A., Nunes, B. P., Lopes, G. R., and Leme, L. (2016). DBpedia Profiler Tool: Profiling the Connectivity of Entity Pairs in DBpedia. In *Proceedings of the 5th International Workshop on Intelligent Exploration of Semantic Data (IESD 2016)*.

Hulpuş, I., Prangnawarat, N., and Hayes, C. (2015). Path-Based Semantic Relatedness on Linked Data and Its Use to Word and Entity Disambiguation. In Arenas, M., Corcho, O., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P., Dumontier, M., Heflin, J., Thirunarayan, K., Thirunarayan, K., and Staab, S., editors, *The Semantic Web - ISWC 2015*, volume 9366, pages 442–457. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bull Soc Vaudoise Sci Nat*, 37:547–579.

Jeh, G. and Widom, J. (2002). SimRank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 538–543. ACM.

Järvelin, K. and Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Le, W., Li, F., Kementsietsidis, A., and Duan, S. (2014). Scalable keyword search on large RDF data. *Knowledge and Data Engineering, IEEE Transactions on*, 26(11):2774–2788.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195.

Li, C., Han, J., He, G., Jin, X., Sun, Y., Yu, Y., and Wu, T. (2010). Fast computation of SimRank for static and dynamic information networks. In *Proceedings of the 13th International Conference on Extending Database Technology - EDBT '10*, page 465, Lausanne, Switzerland. ACM Press.

Lizorkin, D. and Velikhov, P. (2008). Accuracy Estimate and Optimization Techniques for SimRank Computation. *Proceedings of the VLDB Endowment*, 1(1):12.

Milne, D. and Witten, I. H. (2008). An Effective, Low-Cost Measure of Semantic Relatedness Obtained from Wikipedia Links. In *Proceedings of the AAAI 2008 Workshop on Wikipedia and Artificial Intelligence*, pages 25–30, Chicago. AAAI Press.

Moore, J. L., Steinke, F., and Tresp, V. (2012). A Novel Metric for Information Retrieval in Semantic Networks. In García-Castro, R., Fensel, D., and Antoniou, G., editors, *The Semantic Web: ESWC 2011 Workshops*, volume 7117, pages 65–79. Springer Berlin Heidelberg, Berlin, Heidelberg. Series Title: Lecture Notes in Computer Science.

Pirrò, G. (2015). Explaining and Suggesting Relatedness in Knowledge Graphs. In Arenas, M., Corcho, O., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P., Dumontier, M., Heflin, J., Thirunarayan, K., Thirunarayan, K., and Staab, S., editors, *The Semantic Web - ISWC 2015*, volume 9366, pages 622–639. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

Reyhani Hamedani, M. and Kim, S.-W. (2021). On Investigating Both Effectiveness and Efficiency of Embedding Methods in Task of Similarity Computation of Nodes in Graphs. *Applied Sciences*, 11(1):162. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

Sommer, C. (2014). Shortest-path queries in static networks. *ACM Computing Surveys*, 46(4):1–31.

Talavera Herrera, J. E. (2017). *On the Connectivity of Entity Pairs in Knowledge Bases*. Doctoral Dissertation, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, Brazil.