A Scalable Virtual Document-Based Keyword Search System for RDF Datasets

Dennis Dosso
Department of Information Engineering, University of Padua
Padua, Italy
dosso@dei.unipd.it

Gianmaria Silvello
Department of Information Engineering, University of Padua
Padua, Italy
silvello@dei.unipd.it

ABSTRACT

RDF datasets are becoming increasingly useful with the development of knowledge-based web applications. SPARQL is the official structured query language to search and access RDF datasets. Despite its effectiveness, the language is often difficult to use for non-experts because of its syntax and the necessity to know the underlying data structure of the database queries. In this regard, keyword search enables non-expert users to access the data contained in RDF datasets intuitively.

This work describes the TSA+VDP keyword search system for effective and efficient keyword search over large RDF datasets. The system is compared with other state-of-the-art methods on different datasets, both real-world and synthetic, using a new evaluation framework that is easily reproducible and sharable.

CCS CONCEPTS

• **Information systems → Retrieval models and ranking**; *Information retrieval*; *Document representation*; *Evaluation of retrieval results.*

KEYWORDS

RDF graphs, keyword search, evaluation

ACM Reference Format:
Dennis Dosso and Gianmaria Silvello. 2019. A Scalable Virtual Document-Based Keyword Search System for RDF Datasets. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331284

1 INTRODUCTION

In the last decade, the Web of Data has become one of the principal means to expose structured data on the Web [15]. The actual paradigm of the Web of Data is the Linked Data realized via *Resource Description Framework* (RDF) datasets (or RDF graphs). An RDF dataset is a set of triples (subject-predicate-object) each connected to the other to form a directed graph. RDF allows for flexible manipulation, enrichment, data discovery, and reuse of data across applications, enterprises, and community boundaries. Today, RDF datasets typically contains thousands of millions of edges [16].

The SPARQL structured query language is the standard means to retrieve and access data from RDF graphs. Its complexity – i.e., syntax and the need to know the underlying data schema – is, however, an obstacle to non-experts. In this regard, keyword search will enable users to easily access these data, overcoming SPARQL complexity.

Over the past two decades, the research community and data technology vendors have developed new approaches for keyword search over structured databases [2, 12, 19], but no prototype has led to a transition from proof-of-concept implementations into fully deployed systems [5]. The main limitations of state-of-the-art systems regard both *effectiveness* and *efficiency*. To target effectiveness, we need to develop methods for retrieving and ranking candidate answers, which are RDF subgraphs in our case. The structure and the semantic relevance to the query of the latter is the object of evaluation. To target efficiency, in relation to query execution time and memory consumption, systems that scale to real-world dataset sizes have to be designed. Scalability is "the most pressing challenge" that search systems over structured data have to face and this aspect is particularly striking for graph data [16] such as RDF.

The research community has dedicated great effort in searching relational databases, conducting extensive evaluation, and identifying benchmarks to evaluate system efficiency and effectiveness [5]. Regarding RDF, a common evaluation framework and shared benchmark to evaluate keyword search systems are still missing.

Markov Random Fields-Keyword Search (MRF-KS)[13] is one of the top systems that has been evaluated using the relational benchmark [4]. Although MRF-KS has been tested on relational databases, in principle it can be used for any kind of data graph. Based on the "virtual documents" approach [7, 17], the text documents are built starting from the content of nodes and edges of the graph and then indexed and retrieved by means of IR techniques. One of the few native RDF search systems, also based on the virtual documents approach, is the *Structured Language Model* (SLM) [6].

The virtual documents approach is promising from the scalability viewpoint, since it moves part of the computations off-line (e.g., documents creation) and exploits IR ranking techniques and data structures (e.g., inverted indexes) to speed up the search process. Even though IR techniques scale well to large datasets, they need to be adapted in dealing with the complexity of structured data.

Contributions of this paper include: (i) *TSA+VDP*, a virtual documents-based keyword search system that is both efficient and effective over different databases, which overcomes the limitations of other state-of-the-art systems; (ii) a new evaluation framework based on the Cranfield paradigm and a new measure (i.e., *tb-DCG*) that weights the overall utility of a subgraph ranking for the end-user in terms of the top-heaviness (best answers are ranked first) and essentialness (non-redundancy) of the ranking.

Outline The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 describes the proposed keyword search system. Section 4 and 5 present the evaluation framework and the experimental results respectively. Finally, Section 6 draws some conclusions and discusses future work.

2 RELATED WORK

Keyword Search has been extensively studied in the context of structured databases such as Relational DB and Knowledge Bases. Good reviews about these topics are [2, 18, 19].

There are three main approaches to keyword search.

The first one is the *schema-based* (e.g. DISCOVER [1]), which uses the schema of the database to formulate SQL queries starting from the keyword query.

The second one is the *graph-based* (e.g. BANKS [3]), which models relational databases as graphs, where the tuples are nodes and the foreign-primary key relationship between them define the edges. While this approach is one of the most commonly used in literature (cfr. [4]), it specifically relies on the exploration of the graph structure. This prevents the system from scaling to very large databases.

The third approach is the *document-based*, that concatenates the text contained in the nodes and edges of the graph into textual documents and leverages on IR methods for ranking. This approach seems promising from the scalability perspective since online graph traversals, which require a lot of time and memory, can be avoided.

Two main systems that follows this approach are considered: SLM and MRF-KS. The first was designed to work for RDF graphs [6]. It is based on the construction of potential answer subgraphs containing query keywords and on a ranking function that uses an adapted language model. The second system, MRF-KS [13], was originally designed for relational databases. In order to work over RDF graphs, it was re-implemented. This system is based on answer trees and their ranking with an MRF function [14]. The system requires performing off-line pre-computations, and include exploring the graph via the Dijkstra algorithm.

3 TSA+VDP SEARCH SYSTEM

The TSA+VDP search system takes as input an RDF graph $\mathcal{G}$ and a user keyword query Q_k. It creates a set of potential answer graphs G_j and the corresponding textual document $\mathcal{D}_{G_j}$ by combining the strings extracted from IRIs and literals contained in the graphs. A retrieval function on the graphs and documents is applied for ranking according to their *relevance* to Q_k. The ranking is then returned to the user.

Figure 1 shows the main components of the system along with a simple example. There are two main phases: the off-line phase, performed before the user query is issued, and the on-line phase.

The off-line phase is performed by the *Topological Syntactical Aggregator* (TSA) algorithm. As shown in Figure 1, the first step of the algorithm is to build a set of subgraphs – the Representative Collection (RC) – from the RDF dataset. The idea is to create subgraphs via a series of BFS-like explorations of the dataset. TSA

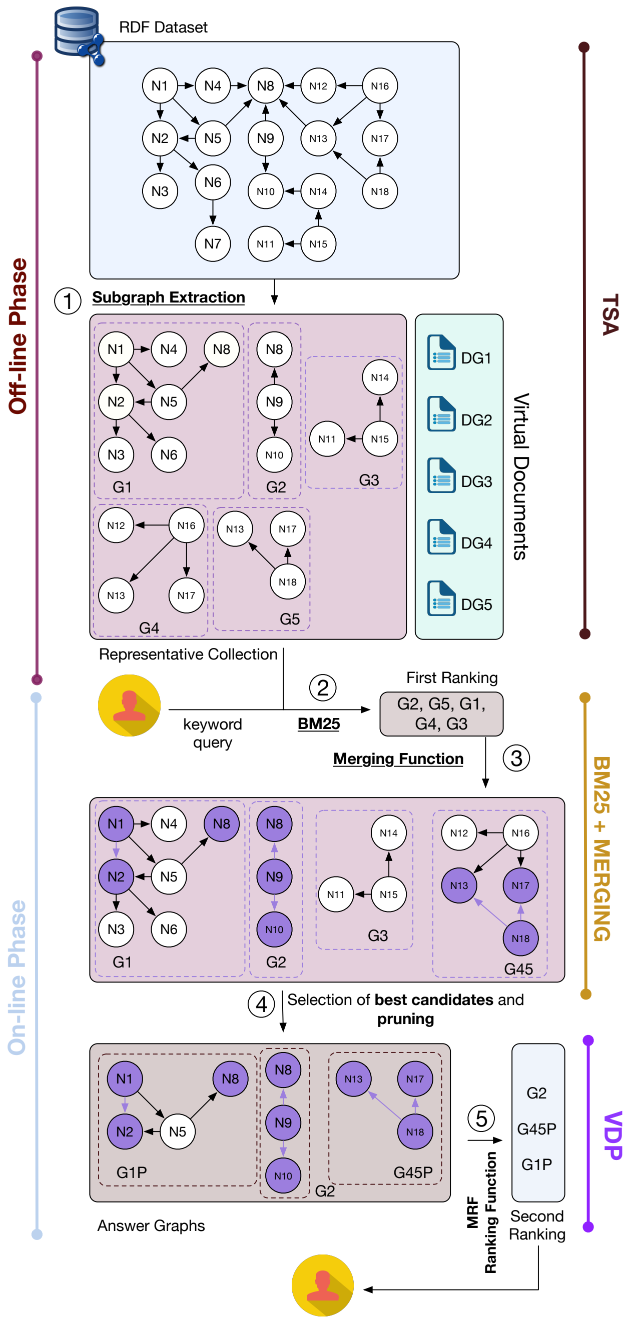

Figure 1: The five steps of the TSA+VDP search system.

considers an IRI node and all the literals connected to it as a big meta-node. For every iteration, the exploration starts from a fresh meta-node (i.e. a source node) with an out-degree greater than a given threshold. Only edges labeled with frequent predicates are traversed. An edge is traversed only if it leads to a fresh source node. The algorithm stops when it encounters a visited node or when it reaches a given graph radius. Each subgraph is then associated with a *Virtual Document* created by connecting the words in the given subgraph nodes and edges. These documents are then indexed with standard IR methods.

The second step of the algorithm begins when the user submits the query. Firstly, a ranking is obtained (top 1,000 results) over all the virtual documents with the BM25 model (step 2 in Figure 1). In step 3 (note that the sub-graph triples containing query keywords are highlighted) the sub-graphs with overlapping triples are merged using a decision algorithm based on a learned threshold.

Afterwards, the *Virtual Document Pruning* (VDP) algorithm (step 4) is applied. First of all, VDP considers the collection of merged graphs and keeps those containing all the query keywords. These graphs are then merged in a not always connected and usually quite small graph, called *query graph*. VDP explores it via BFS and produces new sub-graphs. Finally, the sub-graphs that do not contain all of the query keywords are discarded and the remaining ones go through a pruning phase which eliminates the triples that do not contain at least a keyword. The final result is a collection of answer graphs that are further ordered by a Markov Random Field-based [14] ranking function.

4 EVALUATION FRAMEWORK

The efficacy evaluation is quite straightforward since it is enough to measure the (online and offline) time and memory used by the search systems (i.e. TSA+VDP, SLM and MRFKS) to run the off-line and the on-line search phases.

To measure the effectiveness of the search system a critical difference between standard text retrieval and virtual document-based keyword systems needs to be taken into account. In the former case, the collection of documents is fixed, while in the latter the answer graphs and the virtual documents of the collection are generated dynamically (sometimes on-the-fly) with different strategies and heuristics. Hence, the relevance of a document to a query cannot be verified in advance, as can be done for standard text retrieval.

While [5] defines a shared test collection to evaluate the effectiveness of search systems over relational data, there is no such shared test collection for RDF.

We define a new evaluation framework on the basis of the use of SPARQL queries and their counterpart keyword queries that is easily reproducible and shareable. In our context, the SPARQL query Q_{t_k} associated to a given topic $t_k \in T$, returns the "perfect answer graph" – i.e. the *ground truth graph* $\mathcal{G}_{t_k}$. Thus, all the triples in the answer graph $\mathcal{G}_{t_k}$ are deemed as "relevant" to the topic t_k while all the triples outside $\mathcal{G}_{t_k}$ are "not-relevant".

Definition 4.1. Let Q_k be a keyword query derived from the topic $t_k \in T$ and submitted to a keyword search system and let $\mathcal{G}_{t_k}$ be the ground-truth graph of t_k. We define the ranking returned by the system as $R_k = [ap_1, ap_2, \dots, ap_n]$, where $ap_i = (\mathcal{G}_i, sim_i) \in R_k, i \in [1, n]$ is called *answer pair* s.t. $\mathcal{G}_i$ is the RDF answer graph at rank i and $sim_i \in \mathbb{R}$ is a degree indicating the similarity of $\mathcal{G}_i$ to $\mathcal{G}_{t_k}$. For every couple of answer pairs $\{ap_i, ap_j\} \in R_k$ s.t. $i < j$, $sim_i \geq sim_j$ holds true.

A good keyword search system should return a ranking that is: (i) *top-heavy*: the answer graphs ranked at the top contain more relevant triples than noisy triple; and, (ii) *essential*: redundant triples in different graphs ins the ranking are irrelevant to the user.

Definition 4.2. Given a topic $t_k \in T$, a ranking R_k and the ground-truth graph $\mathcal{G}_{t_k}$, the *Signal-to-Noise Ratio* (SNR) of $\mathcal{G}_i \in R_k$ and the *Graph Relevance Weight* (GRW) of $\mathcal{G}_i \in R_k$ as are defined as follows:

$$SNR(\mathcal{G}_i) = \frac{|(\mathcal{G}_i \cap \mathcal{G}_{t_k}) \setminus S|}{|\mathcal{G}_i|} \quad GRW(\mathcal{G}_i) = \frac{|(\mathcal{G}_i \cap \mathcal{G}_{t_k}) \setminus S|}{|\mathcal{G}_{t_k}|}$$

where S is the union of all the relevant triples in $\mathcal{G}_j \in R_k, \forall j \in [1, i[$.

The SNR recognizes precise and essential graphs, while the GRW is a weighting function which rewards the graphs that contain new relevant triples seen for the first time by the user.

Definition 4.3. Let $t_k \in T$ be a topic, R_k a ranking with length $n \in \mathbb{N}^+$, $\lambda \in [0, 0.1, 0.2, \dots, 1]$ be a threshold value indicating the level of redundancy in the ranking[1] and $b \in \mathbb{N}^+$ a logbase value. We define the *Relevance Gain* (RG) of $\mathcal{G}_i \in R_k$ as

$$RG_b(\mathcal{G}_i) = \begin{cases} GRW(\mathcal{G}_i) & \text{if } i < b \ \wedge \ SNR(\mathcal{G}_i) > \lambda, \\ \frac{GRW(\mathcal{G}_i)}{\log_b i} & \text{if } i \geq b \ \wedge \ SNR(\mathcal{G}_i) > \lambda, \\ 0 & SNR(\mathcal{G}_i) \leq \lambda. \end{cases}$$

The *triple-based Discounted Cumulative Gain* (tb-DCG) of the ranking R_k is defined as

$$tb\text{-}DCG_b(R_k) = \sum_{i=1}^{n} RG_b(\mathcal{G}_i)$$

The tb-DCG is a variation of the DCG measure [11], widely-used in text retrieval.

5 EXPERIMENTAL RESULTS

Three RDF datasets have been considered: LinkedMDB [10] (7M triples), IMDB [2] (116M triples) and the synthetic dataset LUBM (two versions of 1M and 10M triples) [8]. LinkedMDB and IMDB are too big for SLM (> 1000 sec online execution time for every query) and MRF-KS (> 48 hours offline execution time). Hence, to allow effectiveness comparison to TSA+VDP, two reduced versions – rLinkedMDB and rIMDB – of 1M triples each are created by randomly sampling connected components.

We created 50 queries for both LinkedMDB and IMDB; whereas, for LUBM, we used the 14 official queries [3]. The source code of the search systems, the scripts to obtain the test datasets and the code to run the experiments along with the topics and the keyword queries are made available in our git repository[4].

The effectiveness and the efficiency of the considered keyword search systems were tested in terms of tb-DCG ($\lambda = 0.1$), offline time to build the virtual documents, average online time and memory used by the queries for every dataset. The results are shown in Table 1. An ANOVA test among the results was performed to assess the statistical reliability of the systems' performances.

In terms of offline time required to build and index the virtual documents, we can see that SLM has a negligible offline phase, whereas MRF-KS requires up to 2 days to process the rIMDB database. TSA+VDP reports the best trade-off between offline and online time and constantly keeps the offline time under 1h with the solely exception of LUBM10M. The offline execution time is the main bottleneck for MRF-KS. Considering the databases of 1M triples, it is

[1] $\lambda = 0$ redundant triples allowed, $\lambda = 1$ no redundant triples allowed in the ranking.
[2] https://datasets.imdbws.com/
[3] http://swat.cse.lehigh.edu/projects/lubm/queries-sparql.txt
[4] https://bitbucket.org/keywordsearchrdfproject

Table 1: From left to right: dataset name; system name; offline time to build and index the virtual documents; average tb-DCG; average online time; average central memory. † indicates the systems in the top performing group with $p < 0.01$. The best system is in bold.

Dataset	Systems	offline(min)	tb-DCG	online(sec)	memory(MB)
rLinkedMDB	SLM	**1**	0.0005±0.00	334.36±96.01	6.2868±0.55
	MRF-KS	1,368	0.4348±0.10†	113.22±62.67	**2.2252**±0.76
	TSA+VDP	23	**0.5168**±0.13†	**30.69**±09.43†	21.9080±3.48
LinkedMDB	TSA+VDP	64	0.4490±0.12	213.67±87.04	44.528±10.60
rIMDB	SLM	**1**	0.0030±0.00	**34.50**± 1.59†	3.2733±0.06
	MRF-KS	2,880	0.4217±0.06	440.68±189.77	**0.9858**±0.44
	TSA+VDP	12	**0.5449**±0.12†	35.87± 9.53†	19.8860±3.66
IMDB	TSA+VDP	52	0.2697±0.10	248.35±187.37	35.037±23.14
LUBM1M	SLM	**1**	0.0380±0.02	39.60±39.6†	3.4607±1.14
	MRF-KS	1,440	0.0902±0.10†	286.80±130.20	**1.8161**±0.1842
	TSA+VDP	26	**0.2434**±0.1842†	**24.02**±26.27†	5.2571±1.66
LUBM10M	TSA+VDP	510	0.2080±0.170	17.21±11.66	105.34±215.28

interesting to note how TSA+VDP is the algorithm with the highest performances for tb-DCG and the highest efficiency in terms of online time. While MRF-KS attains reasonable values of tb-DCG on rIMDB and rLinkedMDB, its online execution time is significantly higher than TSA+VDP.

LUBM1M is more difficult than the other two databases, perhaps due to the topology of the graph and the nature of the node IRIs. Many triples in LUBM have IRIs and texts that are often repeated within the dataset. It does not have the variability of lexicon found in the other two real-world datasets. Hence, the virtual documents obtained from the subgraphs extrapolated from LUBM1M tend to be more similar to one another than those derived from LinkedMDB and IMDB. The same words are used over and over only in different combinations. Hence, one keyword can be present in many more virtual documents, thus making difficult for the ranking functions to distinguish between relevant and not-relevant documents.

In particular, TSA+VDP is very good on certain queries over LUBM1M, but retrieved no relevant graphs on others. This is due to the inability of BM25 (step 2) to effectively distinguish between relevant and not-relevant virtual documents.

TSA+VDP and MRF-KS are the two top performing algorithms on rLinkedMDB, whereas TSA+VDP performs significantly better than the other systems on rIMDB and on LUBM1M. Moreover, thanks to its heuristics, it performs much faster than the baselines in rLinkedMDB and LUBM1M.

As we can see, only the TSA+VDP algorithm is able to scale to the full-size datasets. Both SLM and MRF-KS time out (> 1000 sec) on all the queries for the full-size datasets. Performances for tb-DCG on LinkedMDB, LUBM10M, and IMDB are significantly lower than the ones on the reduced versions. This is due to the higher number of graphs and virtual documents in the RC. It is harder for BM25 to isolate the more relevant documents and associated graphs, so VDP can rank fewer relevant subgraphs. This phenomenon can be immediately seen on LinkedMDB and LUBM10M but becomes much more evident on the full-sized IMDB database.

The higher number of virtual documents in RC has also an impact on time and memory usage since VDP produces more graphs and the set of final potential answer graphs is also bigger. Consequently, the algorithm requires more memory and more time, since a bigger number of answer graphs needs to be created and ranked.

6 CONCLUSIONS

The TSA+VDP search system is presented herein for keyword search over graph data. The system exploits state-of-the-art IR methods and scales to real-world RDF dataset sizes.

A new evaluation framework for keyword search systems based on structured SPARQL query and corresponding keyword query pairs has been illustrated. This framework can be easily reproduced. It is also based on tb-DCG, a new user-oriented measure, which awards top-heavy and non-redundant rankings.

In the future, TSA+VDP can be tested on other datasets like DBpedia Infobox to further assess the ability of TSA+VDP to scale. Different methods in the construction of the virtual documents and their impact on effectiveness will also be tested.

Acknowledgments This work is partially supported by the Computational Data Citation (CDC-STARS) project of the University of Padua and by the ExaMode project, as part of the European Union Horizon 2020 program under Grant Agreement no. 825292.

REFERENCES

[1] A. Balmin, V. Hristidis, N. Koudas, Y. Papakonstantinou, D. Srivastava, and T. Wang. 2003. A System for Keyword Proximity Search on XML Databases. In *Proc. of 29th International Conference on Very Large Data Bases, VLDB*. Morgan Kaufmann, 1069–1072.

[2] H. Bast, B. Buchhold, and H. Haussmann. 2016. Semantic Search on Text and Knowledge Bases. *Found. and Trends in IR* 10, 2-3 (2016), 119–271.

[3] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan. 2002. Keyword Searching and Browsing in Databases using BANKS. In *Proc. of the 18th International Conference on Data Engineering*. IEEE Computer Society, 431–440.

[4] J. Coffman and A. C. Weaver. 2010. A framework for evaluating database keyword search strategies. In *Proc. of the 19th ACM International Conference on Information and knowledge management*. ACM Press, 729–738.

[5] J. Coffman and A. C. Weaver. 2014. An Empirical Performance Evaluation of Relational Keyword Search Systems. *IEEE Trans. Knowl. Data Eng.* 26, 1 (2014).

[6] S. Elbassuoni and R. Blanco. 2011. Keyword Search over RDF Graphs. In *Proc. of the 20th ACM Conference on Information and Knowledge Management, CIKM 2011,*. ACM Press, New York, USA, 237–242.

[7] J. Feng, G. Li, and J. Wang. 2011. Finding Top-k Answers in Keyword Search over Relational Databases Using Tuple Units. *IEEE Trans. Knowl. Data Eng.* 23, 12 (2011), 1781–1794.

[8] Y. Guo, Z. Pan, and J. Heflin. 2005. LUBM: A benchmark for OWL knowledge base systems. *J. Web Semant.* 3, 2-3 (2005), 158–182.

[9] D. K. Harman. 2011. *Information Retrieval Evaluation.* Morgan & Claypool Publishers, USA.

[10] O. Hassanzadeh and M. P. Consens. 2009. Linked Movie Data Base. In *Proc. of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009*. CEUR-WS.org.

[11] K. Järvelin and J. Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. on Information Systems (TOIS)* 20, 4 2002, 422–446.

[12] A. Kopliku, K. Pinel-Sauvagnat, and M. Boughanem. 2014. Aggregated search: A new information retrieval paradigm. *ACM Comput. Surv.* 46, 3 (2014), 41:1–41:31.

[13] Y. Mass and Y. Sagiv. 2016. Virtual Documents and Answer Priors in Keyword Search over Data Graphs. In *Proc. of the Workshops of the EDBT/ICDT 2016 Joint Conference (CEUR Workshop Proc.)*, Vol. 1558. CEUR-WS.org.

[14] D. Metzler and W. B. Croft. 2005. A Markov random field model for term dependencies. In *SIGIR 2005: Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 472–479.

[15] J. Pound, P. Mika, and H. Zaragoza. 2010. Ad-hoc Object Retrieval in the Web of Data. In *Proc. of the 19th International Conference on World Wide Web, WWW 2010*. ACM Press, New York, USA, 771–780.

[16] S. Sahu, A. Mhedhbi, S. Salihoglu, J. Lin, and M. T. Özsu. 2017. The Ubiquity of Large Graphs and Surprising Challenges of Graph Processing. *PVLDB* 11, 4 (2017), 420–431.

[17] Q. Su and J. Widom. 2005. Indexing Relational Database Content Offline for Efficient Keyword-Based Search. In *Proc. of the 9th International Database Engineering and Applications Symposium (IDEAS 2005)*. IEEE Computer Society, 297–306.

[18] H. Wang and C. C. Aggarwal. 2010. A Survey of Algorithms for Keyword Search on Graph Data. In *Managing and Mining Graph Data*. Springer, 249–273.

[19] J. X. Yu, L. Qin, and L. Chang. 2010. Keyword Search in Relational Databases: A Survey. *IEEE Data Eng. Bull.* 33, 1 (2010), 67–78.