VLDB 2023 Tutorial on

Full-Power Graph Querying: State of the Art and Challenges

Ioana Manolescu, Madhulika Mohanty
Inria and IPP
France
{ioana.manolescu, madhulika.mohanty}@inria.fr

August 16, 2023

Abstract

Graph databases are enjoying enormous popularity, through both their RDF and Property Graphs (PG) incarnations, in a variety of applications. To query graphs, query languages provide structured, as well as unstructured primitives. While structured queries allow expressing precise information needs, they are unsuited for exploring unfamiliar datasets, as they require prior knowledge of the schema and structure of the dataset. Prior research on keyword search in graph databases do not suffer from this limitation. However, keyword queries do not allow expressing precise search criteria when users do know some.

This tutorial (**1.5 hours**) builds a continuum between structured graph querying through languages such as SPARQL [42] and GPML [17], a recently proposed standard for PG querying, on one hand, and graph keyword search, on the other hand. In this space between querying and information retrieval, we analyze the features of modern query languages that go toward unstructured search, discuss their strength, limitations, and compare their computational complexity. In particular, we focus on (*i*) lessons learned from the rich literature of graph keyword search, in particular with respect to result scoring; (*ii*) language mechanisms for integrating *both* complex structured querying and powerful methods to search for connections users do not know in advance. We conclude by discussing the open challenges and future work directions.

1 Introduction

Graph databases Graph-structured data has many applications, e.g., social network analysis, fraud detection, biological networks, etc. [40]. One particularly notable application is that of investigative journalism (IJ), where complex journalistic investigations are backed by digital data. Such data is oftentimes heterogeneous; a database consists of different datasets, possibly of different data models. In an enterprise setting, data could be kept as such and exploited through dedicated data lake tools. However, for journalists, a data lake solution is unfeasible due to lack of IT expertise, time, and resources. Instead, heterogeneous data can be automatically transformed into graphs, which journalists then exploit. This has been pioneered by the International Consortium of Investigative Journalism (ICIJ) in their initial Panama Papers investigation; the ConnectionLens [4, 9, 14] system goes towards the same goal. Other well-known graph database applications related to IJ are financial crime investigation and fraud detection, used by graph database providers such as Oracle and Neo4J to advertise their products.

Graph querying To query graphs, the W3C's standard SPARQL [42] query language and Cypher [39] are among the best known query standards. Additionally, the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) are developing GQL as a Property Graph query standard, with the graph pattern matching sub-language (GPML) [17] at its core. Structured queries require

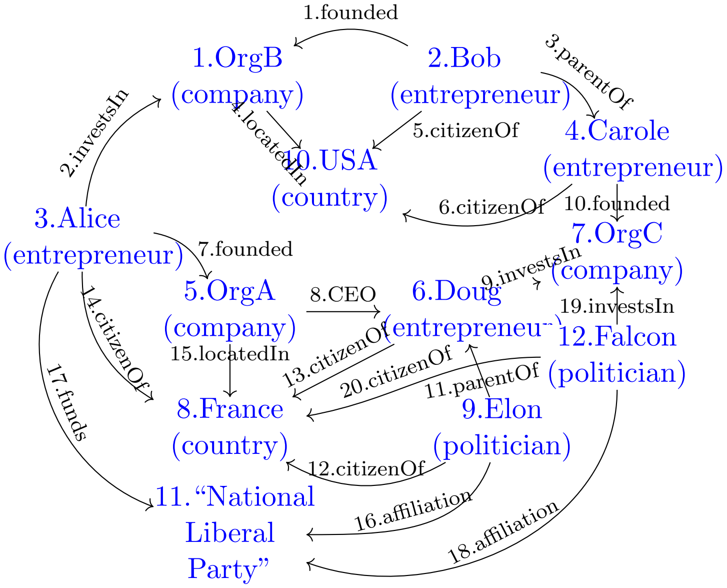

Figure 1: Sample data graph.

users to be familiar with the schema of the graphs; this is typically not the case with journalists, or users discovering datasets. This problem is partially tackled by a useful feature of modern graph query languages, namely *property paths* or *reachability querying*. This allows finding if (and how) two sets of nodes in the graph are reachable to each other. SPARQL 1.1 queries can check, for example, if a French entrepreneur “Alice” has a path to an American entrepreneur “Bob” in the sample graph shown in Figure 1. However, SPARQL 1.1 does not allow returning the matching paths to users. In contrast, a GPML query may also return the paths between two given sets of nodes. This is useful, for example, in the fight against money laundering, by enabling finding routes in which money is laundered across countries, enterprises, etc. None of the graph query languages, however, natively support a general connectivity search between three or more sets of nodes. For instance, in Figure 1, one cannot express the query seeking connections between American entrepreneurs, French entrepreneurs and French politicians. The ability to express such a query is useful for investigative journalists as money laundering usually involves multiple players.

Keyword search in (graph) databases A different paradigm for searching for information in graphs is keyword search. This has been historically studied for structured and semistructured databases, viewed as graphs, e.g., [3, 13, 18, 24, 30]. Users specify m keywords, and request trees (or subgraphs) connecting nodes from the graphs, in which at least one node label matches each keyword. For example, asking “Alice Bob Falcon” leads to returning connecting trees, each containing one node whose label matches each keyword. *Keyword search can thus be seen as at one end of a continuum on an axis of information search in graph, with structured queries being at the other end.*

Computationally speaking, keyword search in graphs is closely related to the Group Steiner Tree Problem (GSTP), which, given m node sets, asks for *the top-score*, e.g., fewest-edges, tree connecting one node from each set; the problem is NP-hard [27]. Algorithms from the literature vary in the score functions that they use, number of solutions they are capable of returning, heuristics used to limit the search (and thus reduce its complexity) etc. It has been shown [15] that these variations lead to important differences in result quality, and even in the very concept of what a result is.

Goals and Objectives At this moment of relative maturity but still vibrant development of graph query languages and especially query evaluation algorithms, the goal of this tutorial is to systematize and compare, in a single framework, graph query *problems* (structured, unstructured, and combined), *dimensions*, such as score functions and other features used to restrict unstructured query results, and an overview of *algorithms* present in the literature. We will also: explain how these problems interface with closely related areas, such as natural language querying of graphs, graph indexing, and graph exploration; and finally, point to open research questions.

2 Tutorial outline

We plan to organize the material as follows.

2.1 Graph Data Models

We very briefly recall standard graph data models, and their specific features: RDF graphs and property graphs. From our perspective, the main differences are: (*i*) whether nodes are "rich" (include their own attributes, à la PG) or just a label (à la RDF); (*ii*) whether graphs just contain data, or are assumed to also come with their ontologies (specific to RDF). In this tutorial, we do not focus on reasoning and ontologies, and just focus on querying graphs as they are. To motivate the need for unstructured querying, we will also introduce ConnectionLens graphs [9], which can be seen as a variant of RDF graphs enriched by Information Extraction, that adds both nodes (extracted entities) and edges (extracted relationships).

2.2 Searching for Information in Graphs

We next introduce the various means for querying data graphs.

1. **Structured Querying:** We recall the useful semantics of the recently proposed structured query standard [17] for Property Graphs known as GPML. In particular, we focus on its support for finding unbounded and unspecified paths between sets of nodes.
2. **Keyword Search:** We introduce the problem of "keyword search" on graphs. We describe the notion of a valid answer and the problem's similarity to the known NP-Hard problem of finding Group Steiner Trees [27] in a graph.

Limitations of Graph Queries We compare the abilities and limitations of each query paradigm. We show that current query languages allow, at best, to return paths whose lengths and labels are not specified in advance. However, they do not allow finding connections between three or more (groups of) nodes. For instance, consider the example query from Section 1:

"What are the connections between an American entrepreneur, a French entrepreneur, and a French politician?"

This query cannot be successfully expressed using any existing graph query language because it requires seeking connections (*trees*, and not just paths) between three sets of nodes, the American entrepreneurs, the French entrepreneurs and the French politicians, respectively. It can neither be achieved by using just keyword search because of its inability to specify and capture the aforementioned *precise* sets of nodes. This highlights the need for a more powerful and expressive language in order to fully express the user's information need, that has both structured and unstructured fragments.

2.3 A Structured Introduction to Keyword Search in Graphs

Next, we discuss the existing keyword search algorithms and introduce the notion of completeness. We also present a classification of the existing keyword search approaches according to the various dimensions and identify their limitations. Overall, we consider the algorithms from the following dimensions:

1. **Notion of an answer:** Existing algorithms differ what is considered an answer. While the majority consider connecting trees as answers [13, 15, 20, 43], a number of algorithms search for a subgraph or clique as an answer [25, 26].

2. **Number of solutions returned:** The keyword search algorithms can either return just one optimal solution [18, 26, 31] or a list of top-k results [13, 20, 30, 32, 45]. For one result, the algorithms provide a set of guarantees about the result being within a known distance from the optimum with respect to a fixed score function. Such algorithms can employ pruning strategies tuned specifically to finding the top-k solutions.

3. **Directionality:** Another categorization of the algorithms is based on the direction of the search. A majority of algorithms only consider a unidirectional search [1, 3, 13, 20, 22, 46]. That is, the result trees have a root node from which there are unidirectional paths to all keyword matching nodes. GAM [9] is the only bidirectional search algorithm.

4. **Data models:** Many keyword search algorithms depend on the intrinsic data model for their performance. The algorithms proposed for RDBMS [3, 16, 21, 22, 32, 33] search for joined tuple trees (JTTs) which are sets of tuples joined by a primary key - foreign key constraint. The techniques proposed in [19, 23] exploit the tree structure of XML data to aid the search. ObjectRank [12] assumes availability of a schema graph. STAR [28] uses the RDF graph taxonomy to reduce the search space. All such algorithms are, thus, schema-dependent.

5. **Use of summaries:** Certain keyword search algorithms do not work directly over the input graph. The ones presented in [12, 29, 41, 47] require a precomputed, compact summary of the graph in order to search for trees.

6. **Score functions:** The algorithms presented in [18, 20, 24, 31] use a fixed score function for scoring their results. The score function's properties are used to limit the search and are therefore, crucial to the runtime guarantees.

7. **Completeness:** The final dimension that we consider is *completeness*. That is, when given sufficient time and space, the search algorithm will find all the solutions regardless of the directions of edges, the data model or score function. We discuss GAM which is a complete algorithm.

Prior surveys studied these algorithms only through their answers [43, 44] and their chosen score function [15].

2.4 Extended Graph Queries

Next, we discuss the existing support and extensions in the graph query languages with respect to unstructured search. We recall the SPARQL1.1 [42] property paths that allow checking for existence of unidirectional paths with only specified labels. We then present the research prototype JEDI [2] which additionally also allows to return such paths. We further discuss the regular path query support in G-CORE [10] and GPML [17] which allow querying and returning an unbounded and bidirectional path with *any* label.

Beyond searching for paths, we then present a recent work [5–7] on extending GPML with Connecting Tree Patterns (CTPs) to allow Extended Queries (EQs) in order to support both structured and unstructured fragments in the same query. The sample query from Section 2.2 can be expressed by the extended queries using the existing query primitives to specify the three sets of nodes corresponding to the American entrepreneurs, the French entrepreneurs and the French politicians, respectively and then, using CTPs to ask for the connections between them. We also present a straight-forward evaluation technique for evaluating the EQs. For evaluating CTPs, we outline MoLESP, an algorithm which speeds up GAM but may be incomplete for certain inputs. Putting it all together, we show a simple strategy for evaluating extended queries, comprising both an unstructured part (CTPs) and a structured one.

2.5 Future Work Directions

With a large number of keyword search algorithms, each being customised for a given dimension, there is still a need for a generic, complete and efficient algorithm. While MoLESP is a step forward in this direction, it is still proven to be incomplete for greater than three keywords. For the execution of EQs, the strategy proposed in [5, 6] uses a straight-forward evaluation mechanism of first evaluating the structured fragment, followed by evaluation of the unstructured fragment. This order may not be always the most optimal. Thus, further work needs to be done in order to decide how to optimally interweave the structured and unstructured parts of the search by using classical techniques such as join ordering, and cost estimation. Additionally, batch evaluation of keyword queries and also EQs in general, remains to be addressed.

3 Target Audience and Duration

This lecture-style tutorial of 1.5 hours is aimed for researchers, graduate students and industry practitioners, who are interested in graph-structured data and its applications. Researchers and graduate students will benefit from an introduction to graphs and ways to query it; they will also find open problems and challenges to be addressed as future work directions. Industry practitioners will get an overview of existing works in graph querying, their scope and limitations along with introduction to recent research prototypes having applicability in various industrial softwares. In general, the tutorial will also enable new users of graph databases to learn about techniques to explore their datasets with ease. No prior knowledge of graphs or databases is expected. However, familiarity with a structured query language like SQL will be helpful.

4 Novelty and Positioning

The tutorial [11] introduced a unified graph data model, the grammar for a structured query language involving regular expression based path querying, and its various evaluation paradigms including approximations. [34] discussed the query processing, primarily the join algorithms, of many prominent graph databases. The tutorial [38] discussed graph exploration using exemplar queries. Our tutorial is the first to cover a continuum of various means of querying graphs – the extended queries support the space starting from purely structured queries for precise information needs to that of unstructured keyword queries for graph exploration. It has not been presented previously.

5 Speakers

Ioana Manolescu (https://pages.saclay.inria.fr/ioana.manolescu/) is a senior researcher at Inria Saclay and a part-time professor at Ecole Polytechnique, France. She is the lead of the CEDAR Inria team, focusing on rich data analytics at cloud scale. She is also the scientific director of LabIA, a program ran by the French government to introduce AI solutions in the public national and local administration. She has been a member of the PVLDB Endowment Board of Trustees, an Associate PVLDB Editor, president of the ACM SIGMOD PhD Award Committee, chair of the IEEE ICDE conference, and a program chair of EDBT, SSDBM, ICWE among others. She has co-authored more than 150 articles in international journals and conferences and co-authored books on “Web Data Management” and on “Cloud-based RDF Data Management”. Her main research interests algebraic and storage optimizations for semistructured data, in particular Semantic Web graphs, novel data models and languages for complex data management, data models and algorithms for fact-checking and data journalism, a topic where she is collaborating with journalists from Le Monde and RadioFrance. Since 2018, she has been working to integrate highly heterogeneous journalistic data sources in graphs with the help of AI and Information Extraction [9, 14] and querying them through keywords [4, 8] and through expressive structured and unstructured queries [5, 7].

Madhulika Mohanty (https://www.madhulikamohanty.com/) is a postdoctoral researcher in the CEDAR team at Inria Saclay. Prior to this, she was an Assistant Professor at IIT Dhanbad in India. She obtained her PhD in 2020 from IIT Delhi with a thesis titled “Techniques for Effective Search and Retrieval over Knowledge Graphs”. During her PhD, she has worked on efficiently querying graphs using structured SPARQL queries [37] and unstructured keyword queries [35, 36]. Recently, she has worked on enabling support for both structured and unstructured queries using extended queries [5, 7]. Her research interests include Graph Data Management, Natural Language Question Answering and Information Retrieval.

References

[1] B. Aditya, Gaurav Bhalotia, Soumen Chakrabarti, Arvind Hulgeri, Charuta Nakhe, Parag, and S. Sudarshan. BANKS: browsing and keyword searching in relational databases. In *VLDB*, 2002.

[2] Christian Aebeloe, Gabriela Montoya, Vinay Setty, and Katja Hose. Discovering diversified paths in knowledge bases. In *VLDB*, 2018.

[3] Sanjay Agrawal, Surajit Chaudhuri, and Gautam Das. Dbxplorer: A system for keyword-based search over relational databases. In *ICDE*, 2002.

[4] Angelos-Christos Anadiotis, Oana Balalau, Théo Bouganim, Francesco Chimienti, Helena Galhardas, Mhd Yamen Haddad, Stéphane Horel, Ioana Manolescu, and Youssr Youssef. Empowering Investigative Journalism with Graph-based Heterogeneous Data Management. In *Bulletin of the Technical Committee on Data Engineering*. IEEE Computer Society, 2021.

[5] Angelos Christos Anadiotis, Ioana Manolescu, and Madhulika Mohanty. Integrating Connection Search in Graph Queries. In *ICDE*, 2023.

[6] Angelos Christos Anadiotis, Ioana Manolescu, and Madhulika Mohanty. Integrating Connection Search in Graph Queries. Technical report, 2023.

[7] Angelos Christos Anadiotis, Ioana Manolescu, and Madhulika Mohanty. More power to SPARQL: From paths to trees. In *ESWC*, 2023.

[8] Angelos-Christos G. Anadiotis, Oana Balalau, Theo Bouganim, Francesco Chimienti, Helena Galhardas, Mhd Yamen Haddad, Stephane Horel, Ioana Manolescu, and Youssr Youssef. Discovering conflicts of interest across heterogeneous data sources with ConnectionLens. In *CIKM*, 2021.

[9] Angelos-Christos G. Anadiotis, Oana Balalau, Catarina Conceição, Helena Galhardas, Mhd Yamen Haddad, Ioana Manolescu, Tayeb Merabti, and Jingmao You. Graph integration of structured, semistructured and unstructured data for data journalism. In *Inf. Syst.*, 2022.

[10] Renzo Angles, Marcelo Arenas, Pablo Barceló, Peter A. Boncz, George H. L. Fletcher, Claudio Gutierrez, Tobias Lindaaker, Marcus Paradies, Stefan Plantikow, Juan F. Sequeda, Oskar van Rest, and Hannes Voigt. G-CORE: A core for future graph query languages. In *SIGMOD*. ACM, 2018.

[11] Marcelo Arenas, Claudio Gutierrez, and Juan F. Sequeda. Querying in the age of graph databases and knowledge graphs. In *SIGMOD*, 2021.

[12] Andrey Balmin, Vagelis Hristidis, and Yannis Papakonstantinou. Objectrank: Authority-based keyword search in databases. In *VLDB*, 2004.

[13] Gaurav Bhalotia, Arvind Hulgeri, Charuta Nakhe, Soumen Chakrabarti, and S. Sudarshan. Keyword searching and browsing in databases using BANKS. In *ICDE*, 2002.

[14] Camille Chanial, Rédouane Dziri, Helena Galhardas, Julien Leblay, Minh-Huong Le Nguyen, and Ioana Manolescu. ConnectionLens: Finding connections across heterogeneous data sources (demonstration). In *PVLDB*, 2018.

[15] Joel Coffman and Alfred C. Weaver. An empirical performance evaluation of relational keyword search techniques. In *IEEE Trans. Knowl. Data Eng.*, 2014.

[16] Pericles de Oliveira, Altigran S. da Silva, Edleno Silva de Moura, and Rosiane Rodrigues. Match-based candidate network generation for keyword queries over relational databases. In *ICDE*, 2018.

[17] Alin Deutsch, Nadime Francis, Alastair Green, Keith Hare, Bei Li, Leonid Libkin, Tobias Lindaaker, Victor Marsault, Wim Martens, Jan Michels, Stefan Plantikow, Petra Selmer, Oskar van Rest, Hannes Voigt, Domagoj Vrgoc, Mingxi Wu, and Fred Zemke. Graph pattern matching in GQL and SQL/PGQ. In *SIGMOD*, 2022.

[18] Bolin Ding, Jeffrey Xu Yu, Shan Wang, Lu Qin, Xiao Zhang, and Xuemin Lin. Finding top-k min-cost connected trees in databases. In *ICDE*. IEEE Computer Society, 2007.

[19] Lin Guo, Feng Shao, Chavdar Botev, and Jayavel Shanmugasundaram. XRANK: ranked keyword search over XML documents. In *SIGMOD*, 2003.

[20] Hao He, Haixun Wang, Jun Yang, and Philip S. Yu. BLINKS: ranked keyword searches on graphs. In *SIGMOD*, 2007.

[21] Vagelis Hristidis, Luis Gravano, and Yannis Papakonstantinou. Efficient ir-style keyword search over relational databases. In *VLDB*, 2003.

[22] Vagelis Hristidis and Yannis Papakonstantinou. DISCOVER: keyword search in relational databases. In *VLDB*, 2002.

[23] Vagelis Hristidis, Yannis Papakonstantinou, and Andrey Balmin. Keyword proximity search on XML graphs. In *ICDE*, 2003.

[24] Varun Kacholia, Shashank Pandit, Soumen Chakrabarti, S. Sudarshan, Rushi Desai, and Hrishikesh Karambelkar. Bidirectional expansion for keyword search on graph databases. In *VLDB*, 2005.

[25] Mehdi Kargar and Aijun An. Keyword search in graphs: Finding r-cliques. In *VLDB*, 2011.

[26] Mehdi Kargar, Lukasz Golab, Divesh Srivastava, Jaroslaw Szlichta, and Morteza Zihayat. Effective keyword search over weighted graphs. In *IEEE Trans. Knowl. Data Eng.*, 2022.

[27] Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations*, 1972.

[28] Gjergji Kasneci, Maya Ramanath, Mauro Sozio, Fabian M. Suchanek, and Gerhard Weikum. STAR: steiner-tree approximation in relationship graphs. In *ICDE*, 2009.

[29] Wangchao Le, Feifei Li, Anastasios Kementsietsidis, and Songyun Duan. Scalable keyword search on large RDF data. In *IEEE Trans. Knowl. Data Eng.*, 2014.

[30] Guoliang Li, Beng Chin Ooi, Jianhua Feng, Jianyong Wang, and Lizhu Zhou. EASE: an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data. In *SIGMOD*, 2008.

[31] Rong-Hua Li, Lu Qin, Jeffrey Xu Yu, and Rui Mao. Efficient and progressive group steiner tree search. In *SIGMOD*, 2016.

[32] Yi Luo, Xuemin Lin, Wei Wang, and Xiaofang Zhou. Spark: top-k keyword query in relational databases. In *SIGMOD*, 2007.

[33] Yi Luo, Wei Wang, Xuemin Lin, Xiaofang Zhou, Jianmin Wang, and Keqiu Li. SPARK2: top-k keyword query in relational databases. In *IEEE Trans. Knowl. Data Eng.*, 2011.

[34] Amine Mhedhbi and Semih Salihoglu. Modern techniques for querying graph-structured relations: Foundations, system implementations, and open challenges. *VLDB*, 2022.

[35] Madhulika Mohanty and Maya Ramanath. Klustree: clustering answer trees from keyword search on graphs. In *COMAD/CODS*, 2018.

[36] Madhulika Mohanty and Maya Ramanath. Insta-search: Towards effective exploration of knowledge graphs. In *CIKM*, 2019.

[37] Madhulika Mohanty, Maya Ramanath, Mohamed Yahya, and Gerhard Weikum. Spec-qp: Speculative query planning for joins over knowledge graphs. In *EDBT*, 2019.

[38] Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. Exploring the data wilderness through examples. In *SIGMOD*, 2019.

[39] Neo4j. Cypher Query Language, 2022.

[40] Sherif Sakr, Angela Bonifati, Hannes Voigt, Alexandru Iosup, Khaled Ammar, Renzo Angles, Walid G. Aref, Marcelo Arenas, Maciej Besta, Peter A. Boncz, Khuzaima Daudjee, Emanuele Della Valle, Stefania Dumbrava, Olaf Hartig, Bernhard Haslhofer, Tim Hegeman, Jan Hidders, Katja Hose, Adriana Iamnitchi, Vasiliki Kalavri, Hugo Kapp, Wim Martens, M. Tamer Özsu, Eric Peukert, Stefan Plantikow, Mohamed Ragab, Matei Ripeanu, Semih Salihoglu, Christian Schulz, Petra Selmer, Juan F. Sequeda, Joshua Shinavier, Gábor Szárnyas, Riccardo Tommasini, Antonino Tumeo, Alexandru Uta, Ana Lucia Varbanescu, Hsiang-Yun Wu, Nikolay Yakovets, Da Yan, and Eiko Yoneki. The future is big graphs: a community view on graph processing systems. In *Commun. ACM*, 2021.

[41] Thanh Tran, Haofen Wang, Sebastian Rudolph, and Philipp Cimiano. Top-k exploration of query candidates for efficient keyword search on graph-shaped (RDF) data. In *ICDE*, 2009.

[42] W3C. SPARQL 1.1, 2013.

[43] Haixun Wang and Charu C. Aggarwal. A survey of algorithms for keyword search on graph data. In *Managing and Mining Graph Data*, Advances in Database Systems. Springer, 2010.

[44] Jianye Yang, Wu Yao, and Wenjie Zhang. Keyword search on large graphs: A survey. In *Data Sci. Eng.*, 2021.

[45] Yueji Yang, Divyakant Agrawal, H. V. Jagadish, Anthony K. H. Tung, and Shuang Wu. An efficient parallel keyword search engine on knowledge graphs. In *ICDE*, 2019.

[46] Zhiwei Zhang, Jeffrey Xu Yu, Guoren Wang, Ye Yuan, and Lisi Chen. Key-core: cohesive keyword subgraph exploration in large graphs. In *World Wide Web*, 2022.

[47] Yuanyuan Zhu, Qian Zhang, Lu Qin, Lijun Chang, and Jeffrey Xu Yu. Cohesive subgraph search using keywords in large networks. In *IEEE Trans. Knowl. Data Eng.*, 2022.