2012 IEEE 28th International Conference on Data Engineering

Emerging Graph Queries In Linked Data

Arijit Khan, Yinghui Wu, Xifeng Yan

Department of Computer Science, University of California, Santa Barbara, USA
{arijitkhan, yinghui, xyan}@cs.ucsb.edu

***Abstract*—In a wide array of disciplines, data can be modeled as an interconnected network of entities, where various attributes could be associated with both the entities and the relations among them. Knowledge is often hidden in the complex structure and attributes inside these networks. While querying and mining these linked datasets are essential for various applications, traditional graph queries may not be able to capture the rich semantics in these networks. With the advent of complex information networks, new graph queries are emerging, including graph pattern matching and mining, similarity search, ranking and expert finding, graph aggregation and OLAP. These queries require both the topology and content information of the network data, and hence, different from classical graph algorithms such as shortest path, reachability and minimum cut, which depend only on the structure of the network. In this tutorial, we shall give an introduction of the emerging graph queries, their indexing and resolution techniques, the current challenges and the future research directions.**

I. Introduction

Recent advances in social and information science have shown that linked data is pervasive in the natural world around us [1]. Examples include communication and computer systems, the World Wide Web, online social networks, biological networks, transportation systems, epidemic networks, chemical networks, and hidden terrorist networks. All these systems are networked systems, in which individual components interact with a specific set of components, resulting in massive, interconnected, and heterogeneous networks.

Given a network modeled as *graph*, there have been several types of commonly used, classical queries, such as shortest path [2], [3], reachability [4], subgraph isomorphism [5], PageRank [6], influence maximization [7], [8], [9], [10] and graph clustering [11]. These queries, nevertheless, depend only on the graph structure, and thus may not be capable to capture the rich semantics associated with nodes, edges and structures in the network graph. With the advent of complex information networks, novel graph queries are emerging, which combine both the contents and topology information of the graph. Typical examples of these queries include graph pattern matching [12], [13], [14] and mining [15], [16], [17], similarity search [18], [19], [20], anomaly detection [21], graph skyline [22] and OLAP [23], ranking and expert finding [24], [25], keyword search [26], [27], and graph aggregation [28], among others. To answer these queries efficiently, novel graph databases and graph query processing systems are also proposed, e.g., Neo4j [29], Pregel [30], SPARQL [31], Trinity [32] and GBase [33].

The emerging novel queries on graph database raises several challenges. First, unlike the traditional graph queries as mentioned earlier, the emerging queries integrate both the structure and attribute information of the network. Hence, traditional algorithms and techniques may not directly apply. Second, when graphs become complex and large, scalability becomes an issue. For example, the online social network Facebook includes more than 300 million users. As such, algorithms for even simple reachability queries may not scale well, as observed in [4]. Third, due to the lack of fixed schema, missing type information, as well as the incomplete knowledge about the structure and contents of the real life information networks, it might be infeasible to use conventional SQL or SPARQL framework to answer these queries.

These new challenges require the development of novel query evaluation algorithms as well as efficient indexing routines for fast graph data access. In earlier tutorials [34], [35], Faloutsos et. al. discussed large graph mining using matrix based methods. In [36], Han et. al. systematically introduced data mining and knowledge discovery algorithms for information networks including graph clustering, ranking (RankClus and NetClus), SimRank, PageRank and graph OLAP. Apart from these, in this tutorial we present an overview of the following novel graph queries - graph pattern mining and matching, similarity search, keyword search and graph skyline. Our tutorial, while covers a part of various emerging graph queries, intends to give a first impression on their challenges and solutions, as well as several related future topics.

II. Tutorial Outline

A. Overview

In this tutorial, we briefly overview of the following three categories of emerging graph queries. **(1)** *Mining queries*, which are to find all frequent subgraphs and patterns from a large scale graph or a set of graphs. **(2)** *Matching queries*, which are to find a given query graph or pattern from a target network. **(3)** *Selection queries*. They are to identify the top-k nodes in a target network based on various input criteria (*i.e.,* SimRank, expert search, keyword search, graph skyline). The basic difference between the matching queries and selection queries is that, the matching queries have explicit structures (e.g. query graphs); whereas the structure is implicit in the selection queries (e.g. keyword search).

B. Mining Queries

Finding all frequent subgraphs and patterns is an active research topic in data mining, with applications in online recommendation, viral marketing and intrusion detection. We

1084-4627/12 $26.00 © 2012 IEEE
DOI 10.1109/ICDE.2012.143

introduce the following two types of mining queries.

Frequent subgraph mining. Efficient subgraph mining algorithms have been proposed to discover frequent subgraphs from *a set of* graphs, including AGM [37], FSG [38], gSpan [15], followed by Path-Join, MoFa, FFSM, GASTON, etc. Techniques were also developed to mine maximal graph patterns [39] and significant graph patterns [40]. In the area of mining a *single* massive graph, [41], [42], [43] developed techniques to calculate the *support* of graph patterns, *i.e.,* a measurement for identifying frequent subgraphs where there are overlapping embeddings in the graph. Kuramochi and Karypis [41] proposed the maximum independent set as the support of subgraphs, which is proved to have the downward closure property by [43]. [44] proposed a support measure that is computationally less expensive and often closer to intuition than other measures. All these methods adopt subgraph isomorphism testing as a way to count the support of graph patterns in single or multiple graphs.

Frequent proximity pattern mining. Apart from the subgraph pattern mining, a novel concept of top-k *proximity pattern mining* has been introduced in [16], where strict isomorphism is not desired. Defined as a set of labels that co-occur in neighborhoods, proximity pattern relaxes the rigid structure constraint of frequent subgraphs, while introducing connectivity to frequent itemsets. Proximity patterns have the following three characteristics: (1) *proximity*, the labels in a pattern appear *close* to each other in the network; (2) *frequency*, the labels occur closely for *several times* in the network; (3) *flexibility*, they are not always connected in the same way. The third characteristic makes the proximity patterns elastic enough to capture fuzzy patterns existing in massive attributed graphs. For example, we may consider a social network, where each node (*i.e.,* user) is attached with a set of labels (*e.g.,* movies recommended by the user). Now, the movies associated with two different users, who have a friendship link, might also be related due to the homophily property. Therefore, the top-k proximity patterns among movies not only consider the collection of movies watched by each user (which is a traditional itemset mining problem); instead, they also consider the movies watched by every user's friends and friends of friends. In [16], the authors proposed an information propagation model, called *Normalized Probabilistic Association* to measure the proximity among labels and used a modification of the FP-tree algorithm to mine the top-k proximity patterns.

C. Matching Queries

Finding a query graph or pattern in a target network has applications in RDF query answering, network alignment, entity name disambiguation and schema matching. We introduce the following two types of matching queries.

Graph pattern matching. A *graph pattern* specifies certain conditions in terms of both structure and labels (e.g., each edge in the query pattern can be mapped to a path of maximum length h in the target network). Therefore, the graph pattern matching problem is to find all the matches for a given pattern in a (usually large) data graph. The problem is typically defined in terms of subgraph isomorphism (*e.g.,* [45], [46], [47], [48], [49]). This makes graph pattern matching NP-complete, and hence, hinders its scalability in finding exact matches. In addition, subgraph isomorphism may be restrictive to identify patterns in emerging applications.

Another host of work focus on using regular expressions to query graphs (*e.g.,* UnQL [50] and Lorel [51]). There has also been theoretical work on conjunctive regular path queries (CRPQs) [52] and extended CRPQs (ECRPQs) [53]. The graph pattern matching problem is already NP-complete for CRPQs, and is PSPACE-complete for ECRPQs [53].

Recent work in [12], [13], [14] aim to strike a balance between expressive power of patterns and increased computational complexity incurred. The authors define graph pattern matching in terms of graph simulation [54], to capture patterns commonly found in practice in *polynomial time*. In [12], (1) a richer class of graph pattern is defined, where an edge denotes the connectivity in a data graph within a predefined number of hops, and (2) the matching is defined based on a notion of *bounded simulation*, an extension of graph simulation. To capture the rich semantics of edge types in real life graphs, [14] proposes a class of reachability queries and a class of graph pattern queries, in which an edge is specified with a regular expression of a certain form, expressing the connectivity in a data graph via edges of various types.

A notion of weak similarity was addressed in [55], which extends simulation by mapping an edge to an unbounded path. Extensions of subgraph isomorphism were studied in [56], [57], [58] for XML schema mapping and for Web site matching, which also allow edge-to-path mappings, but are still NP-complete. Bounded connectivity in graph patterns was considered in [58]. Patterns of [58] impose the *same* bound on all edges. To find matches, which remains NP-*complete*, [58] explores joins and pruning techniques.

Incremental algorithms have proved useful in a variety of areas [59]. However, few results are known about incremental graph pattern matching, far less than their batch counterparts. [14] investigates incremental algorithms for graph pattern matching defined in terms of graph simulation [54], bounded simulation [12] and subgraph isomorphism, with provable performance guarantees.

Graph similarity search. Given an attributed network and a small query graph, the graph similarity search problem is to find the query graph in the target network. Unlike graph pattern matching, graph similarity search (1) does not necessarily identify all the matches, and (2) allows approximate matches rather than exact matches, based on a specified similarity measurement. The *exact version* of the problem is computationally hard due to the complexity of the inherent subgraph isomorphism problem. One interesting approach follows the framework of *filtering-and-verification*, which utilizes feature-based indexes to filter out the negative results and generates a candidate set containing some false positives. In the verifica-

tion phase, a precise computation is conducted to generate the final results based on subgraph isomorphism testing. *gIndex* [60] proposes a frequent subgraph based indexing technique.

Due to noise and the incomplete information (structure and content) in many networks, there might not be any exact match for a given query. Hence, it is more appealing to find the top-k *approximate matches*. Tong et al. proposed G-Ray [48] that tries to preserve the shape of the query graph by allowing some approximation in the match. In [61], a neighborhood hash-based linear time graph kernel has been designed for effective similarity search. TALE [19] and SIGMA [62] algorithms provide efficient indexing methods based on the number of missing edges as the quantitative measure of approximate matching. SAGA [18] identifies approximate subgraphs in a graph database that are similar to the query, allowing for node mismatches, node gaps, as well as graph structural differences. In [20], the authors have introduced a novel graph similarity measure by comparing the neighborhood of matched node pairs. It supports both attribute and topology mismatches.

D. Selection Queries

Keyword search. Ranked keyword search queries [63], [26], [27] in XML data identifies the top-k nodes that are close to a given set of keywords. Keyword search on external memory has been proposed in [64]. Kacholia et. al. proposed bidirectional keyword search on graph datasets [65]. Li et al. considers the problem of ranked keyword search over probabilistic XML data [66]. We note that, in all these state-of-the art keyword search problems, the keywords are connected by only conjunctive operators. Therefore, a generalization of the keyword search queries shall consider a series of keywords connected by both conjunctive and disjunctive operators. Besides, we might also consider negated keywords in a query, *e.g., find an actor who worked with one of the directors, 'Steven Spielberg' or 'James Cameron', but was not born in the 'USA'*. We shall introduce efficient techniques to answer such queries.

Graph skyline. Given a set of multi-dimensional points, the skyline query returns a set of points (referred to as the skyline points), such that none of these points are dominated by any other point in the dataset [67]. Skyline query over graph, first proposed in [22], is still an emerging field of study. It may have various interesting applications. For example, consider the following query: *find the top-k potential collaborators of a query author from DBLP network*. To answer this query, we consider two orthometric criteria - distance from the the query author and similarity with his/ her research interests. Therefore, the problem can be formulated as a skyline query, where the two dimensions are the distance from the query author and the similarity with his/her research interests. We introduce various algorithms to answer such graph skyline queries efficiently.

III. Goals of the Tutorial

A. Learning Outcome

Following are the learning outcomes from this tutorial:

- A brief overview of various kinds of graph queries proposed in the past five years and the state-of-the-art graph data management systems.
- Three types of emerging novel graph queries and their applications, which are significantly different from classical graph algorithms and the relational algebra operators.
- Technical details about graph pattern mining and matching, similarity search, keyword search and graph skyline in the context of emerging graph queries.
- New challenges and future research directions in graph query and graph database; *e.g.,* to support these queries, what will be the best graph storage system? Can we perform such queries in a distributed fashion with dynamic load balancing? How to index a graph to answer these queries faster? What are other interesting graph queries?

B. Target Audience

This tutorial is intended to benefit researchers and system designers in the broad area of graph search, mining, storage and processing that include but not limited to RDF query, Web search, Ontology and Semantic Web, linked data, social/information networks, and NoSQL.

IV. About The Authors

Arijit Khan is a PhD student in the Department of Computer Science, University of California at Santa Barbara. His research interests lie in the area of graph mining and queries.

Yinghui Wu is a research scientist in the Department of Computer Science, University of California at Santa Barbara. He got his PhD from the University of Edinburgh in 2010. His research interests lie in the area of database theory and graph database management, with emphasis on graph database models and query languages.

Xifeng Yan is an assistant professor at the University of California at Santa Barbara. He holds the Venkatesh Narayanamurti Chair in Computer Science. He has been working on modeling, managing, and mining large-scale graphs in bioinformatics, social networks, the Web, and computer systems.

V. ACKNOWLEDGMENTS

This work was sponsored in part by the U.S. National Science Foundation under grant IIS-0954125 and by the Army Research Laboratory under cooperative agreement W911NF-09-2-0053 (NS-CTA). The views and conclusions contained herein are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

References

[1] J. Kleinberg, "Navigation in a small world," *Nature*, vol. 406, p. 845, 2000.

[2] Y. Xiao, W. Wu, J. Pei, W. Wang, and Z. He, "Efficiently indexing shortest paths by exploiting symmetry in graphs," in *EDBT*, 2009.

[3] H. Samet, J. Sankaranarayanan, and H. Alborzi, "Scalable network distance browsing in spatial databases," in *SIGMOD*, 2008.

[4] S. Trißl and U. Leser, "Fast and practical indexing and querying of very large graphs," in *SIGMOD*, 2007.

[5] M. Bayati, M. Gerritsen, D. F. Gleich, A. Saberi, and Y. Wang, "Algorithms for large, sparse network alignment problems," *ICDM*, vol. 0, 2009.

[6] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks*, vol. 30, no. 1-7, pp. 107–117, 1998.

[7] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through social network," in *KDD*, 2003.

[8] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *KDD*, 2009.

[9] W. Chen, A. Colin, R. Cumming, T. Ke, Z. Liu, D. Rincon, X. Sun, Y. Wang, W. Wei, and Y. Yuan, "Influence maximization in social networks when negative opinions may emerge and propagate," in *SDM*, 2011.

[10] D. G. T. Lappas, E. Terzi and H. Mannila, "Finding effectors in social networks," in *KDD*, 2010.

[11] K. Schloegel, G. Karypis, and V. Kumar, "Parallel static and dynamic multi-constraint graph partitioning," *Concurrency and Computation: Practice and Experience*, vol. 14, no. 3, pp. 219–240, 2002.

[12] W. Fan, J. Li, S. Ma, N. Tang, Y. Wu, and Y. Wu, "Graph pattern matching: From intractable to polynomial time," *PVLDB*, vol. 3, no. 1, pp. 264–275, 2010.

[13] W. Fan, J. Li, J. Luo, Z. Tan, X. Wang, and Y. Wu, "Incremental graph pattern matching," in *SIGMOD*, 2011.

[14] W. Fan, J. Li, S. Ma, N. Tang, and Y. Wu, "Adding regular expressions to graph reachability and pattern queries," in *ICDE*, 2011.

[15] X. Yan and J. Han, "gSpan: Graph-based substructure pattern mining," in *ICDM*, 2002.

[16] A. Khan, X. Yan, and K.-L. Wu, "Towards proximity pattern mining in large graphs," in *SIGMOD*, 2010.

[17] Z. Guan, J. Wu, Q. Zhang, A. Singh, and X. Yan, "Assessing and ranking structural correlations in graphs," in *SIGMOD*, 2011.

[18] Y. Tian, R. McEachin, C. Santos, D. States, and J. Patel, "SAGA: a subgraph matching tool for biological graphs," *Bioinformatics*, vol. 23, no. 2, pp. 232–239, 2006.

[19] Y. Tian and J. M. Patel, "Tale: A tool for approximate large graph matching," in *ICDE*, 2008.

[20] A. Khan, N. Li, Z. Guan, S. Chakraborty, and S. Tao, "Neighborhood based fast graph search in large networks," in *SIGMOD*, 2011.

[21] H. Tong and C.-Y. Lin, "Non-negative residual matrix factorization with application to graph anomaly detection," in *SDM*, 2011.

[22] L. Zou, L. Chen, M. T. Özsu, and D. Zhao, "Dynamic skyline queries in large graphs," in *DASFAA*, 2010.

[23] C. Chen, X. Yan, F. Zhu, J. Han, and P. S. Yu, "Graph olap: Towards online analytical processing on graphs," in *ICDM*, 2008.

[24] C. Li, J. Han, G. He, X. Jin, Y. Sun, Y. Yu, and T. Wu, "Fast computation of simrank for static and dynamic information networks," in *EDBT*, 2010.

[25] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *PVLDB*, vol. 4, no. 11, pp. 992–1003, 2011.

[26] M. Theobald, R. Schenkel, and G. Weikum, "An efficient and versatile query engine for topx search," in *VLDB*, 2005.

[27] H. He, H. Wang, J. Yang, and P. S. Yu, "Blinks: ranked keyword searches on graphs," in *SIGMOD*, 2007.

[28] X. Yan, B. He, F. Zhu, and J. Han, "Top-k aggregation queries over large networks," in *ICDE*, 2010, pp. 377–380.

[29] Neo4j, "neo4j.org."

[30] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: a system for large-scale graph processing," in *PODC*, 2009, pp. 6–6.

[31] E. Prud'hommeaux and A. Seaborne, "SPARQL Query Language for RDF," W3C Recommendation, 2008.

[32] Trinity, "http://research.microsoft.com/en-us/projects/trinity/."

[33] U. Kang, H. Tong, J. Sun, C.-Y. Lin, and C. Faloutsos, "Gbase: A scalable and general graph management systemn," *KDD*, 2011.

[34] J. Leskovec and C. Faloutsos, "Tools for large graph mining: Structure and difference," in *WWW*, 2008.

[35] C. Faloutsos, G. Miller, and C. Tsourakakis, "Large graph mining: Power tools and a practioner's guide," in *KDD*, 2009.

[36] J. Han, Y. Sun, X. Yan, and P. S. Yu, "Mining knowledge from databases: an information network analysis approach," in *SIGMOD*, 2010.

[37] A. Inokuchi, T. Washio, and H. Motoda, "An apriori-based algorithm for mining frequent substructures from graph data," in *PKDD*, 1998.

[38] M. Kuramochi and G. Karypis, "Frequent subgraph discovery," in *ICDM*, 2001.

[39] J. Huan, W. Wang, J. Prins, and J. Yang, "Spin: Mining maximal frequent subgraphs from graph databases," in *KDD*, 2004.

[40] H. He and A. Singh, "Efficient algorithms for mining significant substructures in graphs with quality guarantees," in *ICDM*, 2007.

[41] M. Kuramochi and G. Karypis, "Finding frequent patterns in a large sparse graph," in *SDM*, 2004.

[42] J. Chen, W. Hsu, M.-L. Lee, and S.-K. Ng, "NeMoFinder: Dissecting genome-wide protein-protein interactions with meso-scale network motifs." in *KDD*, 2006.

[43] M. Fiedler and C. Borgelt, "Support computation for mining frequent subgraphs in a single graph," in *MLG*, 2007.

[44] B. Bringmann and S. Nijssen, "What is frequent in a single graph?" in *PAKDD*, 2008.

[45] J. Cheng, J. X. Yu, B. Ding, P. S. Yu, and H. Wang, "Fast graph pattern matching," in *ICDE*, 2008.

[46] N. Bruno, N. Koudas, and D. Srivastava, "Holistic twig joins: optimal XML pattern matching," in *SIGMOD*, 2002.

[47] L. Chen, A. Gupta, and M. E. Kurul, "Stack-based algorithms for pattern matching on dags," in *VLDB*, 2005.

[48] H. Tong, C. Faloutsos, B. Gallagher, and T. Eliassi-Rad, "Fast best-effort pattern matching in large attributed graphs," in *KDD*, 2007.

[49] B. Gallagher, "Matching structure and semantics: A survey on graph-based pattern matching," *AAAI FS.*, 2006.

[50] P. Buneman, M. F. Fernandez, and D. Suciu, "Unql: A query language and algebra for semistructured data based on structural recursion," *VLDB J.*, vol. 9, no. 1, pp. 76–110, 2000.

[51] S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. L. Wiener, "The lorel query language for semistructured data," *International Journal on Digital Libraries*, vol. 1, no. 1, pp. 68–88, 1997.

[52] D. Florescu, A. Y. Levy, and D. Suciu, "Query containment for conjunctive queries with regular expressions," in *PODS*, 1998.

[53] P. Barcelo, C. Hurtado, L. Libkin, and P. Wood, "Expressive languages for path queries over graph-structured data," in *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, ser. PODS, 2010.

[54] M. R. Henzinger, T. Henzinger, and P. Kopke, "Computing simulations on finite and infinite graphs," in *FOCS*, 1995.

[55] L. D. Nardo, F. Ranzato, and F. Tapparo, "The subgraph similarity problem," *TKDE*, vol. 21, no. 5, pp. 748–749, 2009.

[56] W. Fan and P. Bohannon, "Information preserving XML schema embedding," *TODS*, vol. 33, no. 1, 2008.

[57] W. Fan, J. Li, S. Ma, H. Wang, and Y. Wu, "Graph homomorphism revisited for graph matching," *PVLDB*, vol. 3, 2010.

[58] L. Zou, L. Chen, and M. T. Özsu, "Distance-join: Pattern match query in a large graph database," in *VLDB*, 2009.

[59] G. Ramalingam and T. W. Reps, "A categorized bibliography on incremental computation," in *POPL*, 1993.

[60] X. Yan, P. Yu, and J. Han, "Graph indexing: A frequent structure-based approach," in *SIGMOD*, 2004.

[61] S. Hido and H. Kashima, "A linear-time graph kernel," in *ICDM*, 2009.

[62] M. Mongiovì, R. D. Natale, R. Giugno, A. Pulvirenti, A. Ferro, and R. Sharan, "Sigma: a set-cover-based inexact graph matching algorithm," *J. Bioinformatics and Computational Biology*, vol. 8, no. 2, pp. 199–218, 2010.

[63] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, "Keyword searching and browsing in databases using banks," in *ICDE*, 2002.

[64] B. Dalvi, M. Kshirsagar, and S. Sudarshan, "Keyword search on external memory data graphs," *Proc. VLDB Endow.*, vol. 1, pp. 1189–1204, 2008.

[65] V. Kacholia, S. Pandit, S. Chakrabarti, S. Sudarshan, R. Desai, and H. Karambelkar, "Bidirectional expansion for keyword search on graph databases," in *VLDB*, 2005.

[66] J. Li, C. Liu, R. Zhou, and W. Wang, "Top-k keyword search over probabilistic xml data," in *ICDE*, 2011.

[67] S. Börzsönyi, D. Kossmann, and K. Stocker, "The skyline operator," in *ICDE*, 2001.