Subgraph Matching With Set Similarity In a Large Graph Database

Shruti Rakte[1], Dhanshri Patil[2], Pratiksha Shinde[3], Aniket Chimurkar[4], Deepa Abin[5]
[1, 2, 3, 4, 5] Department of Computer Engineering
[1, 2, 3, 4, 5] Pimpri Chinchwad College of Engineering, Pune, Maharashtra, India.

Abstract- *In real-world graphs such as social networks, Semantic Web and biological networks, each vertex usually contains rich information, which can be modelled by a set of tokens or elements. We use a subgraph matching with set similarity (SMS2) query over a large graph database, which retrieves subgraphs that are structurally isomorphic to the query graph, and meanwhile satisfy the condition of vertex pair matching with the weighted set similarity. To develop an efficient search engine using subgraph similarity search on large probabilistic graph databases. Based on the index and signatures, we use an efficient two-phase pruning strategy including Vertical and horizontal pruning, which exploits the unique features of both weighted set similarity and graph topology. We use an efficient dominating set based subgraph matching algorithm guided by a dominating set selection algorithm to achieve better query performance.*

Keywords- SMS2, DS match algorithm, Vertical and horizontal pruning.

I. INTRODUCTION

With the emergence of many real applications such as social networks , Semantic Web , and biological networks, graph databases have been widely used as important tools to model and query complex graph data. Much prior work has extensively studied various types of queries over graphs, in which subgraph matching is a fundamental graph query type. Given a query graph Q and alarge graph G, a typical subgraph matching query retrieves those subgraphs in G that exactly match with Q in terms of both graph structure and vertex labels . However, in some real graph applications, each vertex often contains a rich set of tokens or elements representing features of the vertex, and the exact matching of vertex labels is sometimes not feasible or practical.

Motivated by the observation above, in this paper, we focus on a variant of the subgraph matching query, called subgraph matching with set similarity (SMS2) query, in which each vertex is associated with a set of elements with dynamic weights instead of a single label. The weights of elements are specified by users in different queries according to different application requirements or evolving data.

II. LITERATURE SURVEY

[1] Liang Hong Lei Zou, Xiang Lian ,Philip S. Yu , Subgraph Matching with Set Similarity in a Large Graph Database,

In real-world graphs such as social networks, Semantic Web and biological networks, each vertex usually contains rich information, which can be modelled by a set of tokens or elements. we study a subgraph matching with set similarity (SMS2) query over a large graph database, which retrieves subgraphs , and meanwhile satisfy the condition of vertex pair matching with the weighted set similarity. An efficient two-phase pruning strategy in- cluding set similarity pruning and structure-based pruning, which exploits the unique features of both set similarity and graph topology. An efficient dominating-set-based subgraph matching algorithm guided by a dominating set selection algorithm to achieve better query performance.

[2] Z. Sun, H. Wang, H. Wang, B. Shao, and J. Li, Efficient subgraph matching on billion node graphs, PVLDB,

The potential to manage tremendous scale graph knowledge is valuable to an growing number of purposes. So much work has been committed to helping normal graph operations similar to subgraph matching, reachability, typical expression matching, and many others. In many instances, graph indices are employed to velocity up question processing. For very large graphs, super-linear methods are usually infeasible and the quandary of subgraph matching on billion-node graphs. A novel algorithm that sup- ports efficient subgraph matching for graphs deployed on a distributed reminiscence retailer. Rather of counting on tremendous-linear indices, an efficient graph exploration and gigantic parallel computing are used for query processing. Thus, the feasibility of performing subgraph matching on internet-scale graph knowledge.

[3] P. Zhao and J. Han, On graph query optimization in large networks, PVLDB,

At the core of many developed community operations lies a long-established and crucial graph query primitive: the way to search graph structures efficiently within a enormous community? Sadly, the graph query is rough due to the NP-whole nature of subgraph isomorphism. It turns into even difficult when the network examined is significant and various.A excessive efficiency graph indexing mechanism, SPath, to address the graph question problem on giant networks. SPath leverages decomposed shortest paths around vertex local as common indexing units, Via SPath, a graph query is processed and optimized past the traditional vertex-at-a-time trend to a extra efficient path-at- a-time approach. Candidate paths are further joined together to help recuperate the question graph to finalize the graph query processing. SPath with the trendy GraphQL on each actual and artificial data sets.The reports exhibit the effectiveness and scalability of SPath, which are more functional and efficient indexing process in addressing graph queries on huge networks.

[4] M. Hadjieleftheriou and D. Srivastava, Weighted set-based string sim- ilarity, in IEEE Data Engineering Bulletin, 2010.

Remember a universe of tokens, each and every of which is related to a weight, and a database such as strings that may be represented as sub- sets of these tokens. Given a question string, also represented as a set of tokens, a weighted string similarity question identifies all strings in the database whose similarity to the question is better than a consumer des-ignated threshold. Weighted string similarity queries are priceless in pur- poses like information cleaning and integration for locating approximate suits within the presence of typographical mistakes, a couple of format- ting conventions, information transformation blunders, and so on. Semantic properties that may be exploited to design index buildings that help very effective algorithms for query answering.

[5] Y. Tian and J. M. Patel, Tale: A tool for approximate large graph matching

Large graph datasets are common in many emerging database applications, and most notably in large-scale scientific applications. To fully exploit the wealth of information encoded in graphs, effective and efficient graph matching tools are criticalA novel technique for approximate matching of large graph queries is present and novel indexing method that incorpo- rates graph structural information in a hybrid index structure is also used. This indexing technique achieves high pruning power and the index size scales linearly with the database size. An innovative matching paradigm to query large graphs technique is used. This technique distinguishes nodes by their importance in the graph structure. The matching algorithm first matches the important nodes of a query and then progressively extends these matches.

[6] Y. Tian, R. C. McEachin, C. Santos, D. J. States, and J. M. Patel, Saga:

A subgraph matching tool for biological graphs, Bioinformatics, vol.23, no. 2, pp.2007.[6] Large databases of biological graphs, such as pathways and protein inter- action networks, are now available and many of these databases are rapidly growing in size. Graph matching queries provide a powerful method for understanding cellular functions encoded in these graph datasets. Previous research on graph querying methods has largely focused on exact or near exact graph matching. Approximate graph matching is more useful in these applications. A novel approximate graph matching technique called SAGA (Substructure Index based Approximate Graph Alignment). SAGA employs a flexible graph distance model to measure similarities between graphs. To expedite query execution on large databases, a novel index is built on the database graphs. Furthermore, SAGA is orders of magnitude faster than existing methods. The results produced by SAGA can help life sciences researchers discover conserved function units among different pathways, detect potential pathways involved in or affected by a particular disease, and integrate different pathway databases to produce more complete data. In addition to pathway analysis, SAGA can be used to compare biomedical documents also.

[7] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, and Z. Ives,

Dbpedia: A nucleus for a web of open data, in ISWC,DBpedia is a group effort to extract structured expertise from Wikipedia and to make this information available on the web. DBpedia enables you to ask sophisticated queries against datasets derived from Wikipedia and to hyperlink other datasets on the net to Wikipedia data. The extraction of the DBpedia datasets and the ensuing is published on the internet for human-andmachine- consumption.Some emerging functions from the DB- pedia group and internet site authors can facilitate DBpedia content within their websites. Ultimately, the current popularity of interlinking DBpe- dia with other open datasets on the internet and description how DBpedia might function a nucleus for an emerging web of open data is efficietntly researched..

III. IMPLEMENTATION DETAILS

This approach includes offline processing and online processing

Offline processing:

This construct a novel inverted pattern lattice to facilitate efficient pruning which is based on the set similarity. Since the dynamic weight of each element makes existing indices inefficient for answering SMS queries, the need is to design a novel index for SMS query. Motivated by the anti-monotone property of the lattice structure , then frequent patterns from element sets of vertices in the data graph G, and organize them into a lattice. Also store data vertices in the inverted list for each frequent pattern P, if P is contained in the element sets of these vertices. The lattice together with the inverted lists is called inverted pattern lattice, which can greatly reduce the cost of dynamic weighted set similarity search. To support structure-based pruning, encode each query vertex and data vertex into a query signature and a data signature respectively by considering both the topology and set information, and hash all the data signatures into signature buckets.

Online processing:

This propose of finding a cost-efficient dominating set of the query graph , only search candidates for vertices in the dominating set, different dominating sets will lead to different query performances. Thus, a dominating set selection algorithm to select a cost- efficient dominating set DS(Q) of query graph Q. The dominating-set- based subgraph matching is motivated by two observations: (1) finding candidates in SMS queries are much more expensive than that in typical subgraph search, because set similarity calculation is more costly than vertex label matching. As a result, the filtering cost can be reduced by searching dominating vertices of V (Q) rather than all query vertices. (2)Some query vertices may have a large amount of candidate vertices, which leads to many of the unnecessary intermediate results during subgraph matching. Therefore, the subgraph matching cost can also be reduced by decreasing the size of intermediate results. For each vertex u ∈ DS(Q), This propose a two phase pruning strategy, including set similarity pruning and structure-based pruning. The set similarity pruning includes anti- monotone pruning, horizontal pruning, and vertical pruning, which are based on inverted pattern lattice . Based on the signature buckets, Also the structure-based pruning technique by utilizing novel vertex signatures. After the pruning, This propose an efficient DS-Match subgraph matching algorithm to obtain subgraph matches of Q based on candidates of dominating vertices in DS(Q). DS-match utilizes topological relations between dominating vertices and non- dominating vertices to reduce the scale of intermediate results during subgraph matching, and therefore reduces the matching cost.

IV. RESULTS

However, a query graph is much smaller than the data graph in subgraph isomorphism problems, while the two graphs usually have similar size in graph alignment problems. To solve subgraph isomorphism problems, graph alignment algorithms introduce additional cost as they should first find candidate subgraphs of similar size from the large data graph. In addition, existing exact subgraph matching and graph alignment algorithms do not consider weighted set similarity on vertices, which will cause high post processing cost of set similarity computation. Two graphs on the same set of N nodes, but with possibly different sets of edges (weighted or unweighted). The correspondence between the nodes of the two graphs (like the people in PC don't vary across graphs) is already known. Graph similarity involves determining the degree of similarity between these two graphs (a number between 0 and 1). Consider a series of T graphs, each of them over the same set of N nodes, but with possibly different edges (weighted or unweighted). Assume that the correspondence between the nodes is already known. Subgraph matching involves identifying the coherent or well-connected subgraphs that appear in some or all of the T graphs. Pruning: A matching subgraph should not only have its vertices (element sets) similar to that in query graph Q, but also preserve the same structure as Q. Thus, in this section, we design lightweight signatures for both query vertices and data vertices to further filter the candidates after set similarity pruning by considering the structural information.

Table 1.

Key Set Count	proposed scheme query graph count	Existing System query graph count
2	30	4
4	16	3
8	8	3
16	3	2

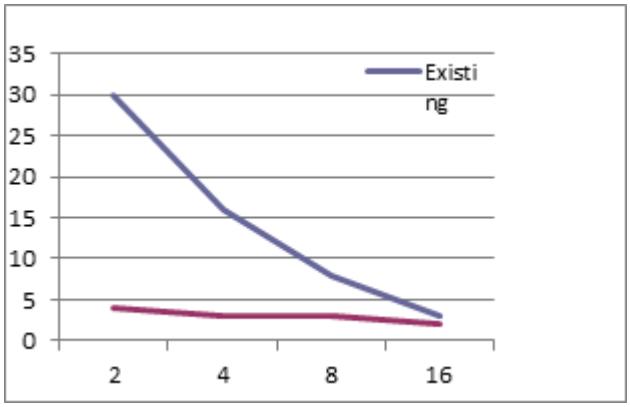

Figure 1. Graph

V. CONCLUSION

In our system, we have used "subgraph matching with set similarity" which exists in a wide range of applications. We have also used efficient pruning techniques by considering both vertex set similarity and graph topology. Our system will provide user with the relevant and more accurate results in online searching as well as in offline searching if the query is previously searched. It also recommends related searches to the user based on history. Our system can be used efficiently in many web related applications. As it supports offline search also,we can use it without internet access which is very useful.

REFERENCES

[1] Liang Hong Lei Zou, Xiang Lian ,Philip S. Yu , Subgraph Matching with Set Similarity in a Large Graph Database, vol. 27, no. 9, 2015.

[2] Zhao Sun, Hongzhi Wang, Haixun Wang, Bin Shao, Jianzhong Li, Efficient subgraph matching on billion node graphs, PVLDB, vol. 5, no. 9, 2012.

[3] Peixiang Zhao and Jiawei Han, On graph query optimization in large networks, PVLDB, vol. 3, no. 1-2, 2010.

[4] M. Hadjieleftheriou and D. Srivastava, Weighted set-based string similarity, in IEEE Data Engineering Bulletin, 2010.

[5] Sharon Bruckner1, Falk Hu¨ ffner1, Richard M. Karp2, Ron Shamir1 and Roded Sharan1, Torque: topology-free querying of protein interaction networks, Nucleic Acids Research, vol. 37, no. suppl 2, pp. W106W108, 2009.

[6] Jiefeng Cheng , Jeffrey Xu Yu , Bolin Ding , Philip S. Yu t, Haixun Wang, Fast graph pattern matching, in Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on. IEEE, 2008, pp. 913922.

[7] Yuanyuan Tian, Jignesh M. Patel, Tale: A tool for approximate large graph matching, in ICDE, 2008.