Graph based Keyword Search using Temporal Predicates

Poonam S. Sonawane[1], Prof.R.P.Dahake[2]
*[1]Department of Computer Engineering, SPPU, Nashik, Maharashhtra, India
[2]Department of Computer Engineering, SPPU, Nashik, Maharashtra , India

Abstract : ***Graph mining is important in variety of applications such as social network analysis, bibliography analysis, scientific graph databases etc. User fire queries on graph and get archived results. Keyword based result extraction is proposed in literature work but is insufficient to answer temporal queries. The proposed work aims to focus on temporal based keyword queries searching on graph. Custom query syntax is used to fire query containing search keywords with conditional strategies and ranking function. To support temporal queries, edges are used. Edges represent one or more time intervals. Best path iterator is used to travel between nodes with the specified ranking function. Ranking functions are proposed to generate archived search result against the entire temporal graph. For user convenience a custom query generation panel is defined. This panel helps to user to select query keywords, wildcard query keywords, temporal predicate, ranking function and aggregate function conditions. The system finds the results on temporal graph and generates a sub-graph containing all matching nodes.***

IndexTerms - graph keyword search, graph data archiving, ranking functions, temporal graph.

I. INTRODUCTION

Summarized report generation and archived results extraction from large graph dataset is required in many domains such as social network analysis, bibliography analysis, scientific graph databases etc. A large dataset is generated in variety of domains. In collaborative projects the previous data can be kept in archived format and then extracted when required. Apart from time based archive data extraction, user may want to fire a query on temporal graphs to relate existing and current scenario. This is required to analyze how things are changed also helps to observe lifeline of an object/product. With the help of temporal information trend analysis can be done. This time wise data also helps in prediction theory.

In existing literature work lot of work has been done on keyword search on graph. This keyword search is not sufficient to answer the temporal queries.

Consider some examples of temporal queries:

1. Find the papers on graph theory published after 2016.
2. Find people employed in Google after 2016.

Temporal relation model[2] and temporal xml model[3] were used in literature generate to answer such type of temporal queries. The relational database can generate answer for such queries but many applications have graph structure data. The graph structure data cannot be efficiently handled by the relational database.

On graph dataset keyword based search technique is employed in many cases. For answer generation to temporal queries the graph snapshots are generated for defined time interval. The temporal aspect is covered using graph snapshot. The query is executed using keyword based searching on desired number of graph snapshots. Graph based temporal information storage require extensive storage space. For processing such temporal queries, huge number of traversal is required based on matching the number of snapshots with the desired temporal query requirement.

For result extraction from a graph using a query, requires some formatted query as an input to the system, so that system can parsed the query and fire it on dataset. The Sql type queries are one of the solutions. Sql queries are structured queries and widely used in database operations. The sql query has predefined syntax and It is studied by the database operators. Due to the structural complexity of sql queries, such queries are not suitable for any casual user. Casual users who are not computer experts should be able to express a query on temporal graph. Hence query format should be easy to understand to any casual user. Along with result extraction of matched keywords on dataset, the result should be in appropriate structure like in sorted format by considering order of the attribute like time or object name.

A query on temporal graph includes multiple aspects such as: ease in query, graph format, processing efficiency, storage efficiency, query facilities such as sorting, ranking, collaborative data analysis etc. In the following literature work, the various keyword search techniques on graph data set have been discussed. In the further section temporal search techniques, its strategies and limitations are discussed.

II. Related Work:

Lot of research work has been done on keyword search on graph[4]. Minimal tree semantics strategy is used to filter the graph and to find relevant query keyword matching. Sub trees are generated from entire graph by matching the query keywords. Top k results are generated as per the tree height. Small height tree has high matching relevance. But such type of graphs do not encodes the temporal information in graph and lacks in temporal result generation.

For keyword matching unlike tree formation minimum length, path traversal techniques such as Dijkstras algorithm[6] is used to find shortest path between matched keyword nodes. A technique named as BANK[5] supports keyword based search on temporal graphs. This algorithm runs Dijkstras algorithm multiple times for iterating path from every query keyword node. Hence this method takes longer time for result extraction.

Temporal databases were proposed to define various version or for trends identification. Relational Databases, graph and/or semi structured data format such as xml were proposed to preserve temporal information. A transaction time or a valid time is included as one more tuple in dataset. Temporal database structure is proposed to preserve temporal information in database. This structures follows the relational database structure along with the transaction time or valid time included in every tuple. TQUEL[7], TSQL2[8] and SQL3[9] languages were proposed to fire query on such temporal dataset format. But these query syntax are quite complicated and not convenient for casual user. This temporal relational database format is also unable to handle graph data structures.

Temporal graphs are also called as dynamic graph or time evolving graph. Lots of techniques[10][11] have been proposed to fire specific type of queries on temporal graph. User can get a single node information from a graph in terms of its time varying history [10]. Reachability of nodes, sub-graph matching queries are again type of temporal graph processing. Distributed systems are used to store huge graph data,. Graph processing on distributed environment is also proposed in literature[11].

Keyword based temporal graph mining is done in[12]. In this technique temporal graph snapshots are stored independently. The data clusters were generated in prior step as per the temporal information. This technique uses dijkstras algorithm for finding shortest path for marched nodes. This algorithm was executed distinct graph snapshot independently and collective results are generated at the end.

In this system proposed a solution for shortest path problem over temporal graphical data. For such temporal data storage, graph snapshots are used. A clustering is performed over graph snapshot and the collection of these graph snapshots stored independently. The Dijkstras algorithm is used to find to find shortest path on cluster result. If no match found with cluster result then individual graph search technique is applied. This process is time consuming process and each graph snapshot is stored independently on disk and hence high memory storage is required for such technique.[13]

Road network or communication network also generates temporal graph. In such graphs, nodes connectivity varies with respect to time. Along with the connectivity edge weights are also varies. Processing of such dynamic graph is also proposed in literature[14].

Large graph those cannot be loaded in the memory for processing are processed using partitioning A method to overcome the problem of storage space and execution time. This technique combines more than one graph snapshot and generates a single graph structure. This is directed graph, it contains: nodes and edges denoted with one or more time interval. This system helps to support queries containing temporal aspect of data. A technique is proposed to answer user queries and provide archived information based on the relationship among 2 or more intervals. In this system graph nodes and edges represents one or more time intervals. Custom query structure is proposed to fire a query over temporal graph data. System successfully parses the query if appropriate syntax is provided. The query syntax includes query keywords, predicate and ranking functions. Predicate includes the search constraints and ranking function includes the temporal filtering criteria. Ranking provides sorting of data in ascending and descending order of time, relevance and duration.[1]

strategy. The issues and challenges of temporal graph partitioning and structural graph partitioning are also discussed in literature[15].

Keyword based result extraction is proposed in literature, is insufficient to answer temporal queries. Different mechanism needs to be applied, to deal with temporal queries. Graph should contain time information along with keyword or weight information with every nodes and edges.[16]

In existing system very limited functions are incorporated with the custom syntax. There is need to enrich the custom query formats in a very simplest manner and provide more options to user to get desired results. Our proposed work is based on giving added facilities to user for more archived results.[17]

III. Problem Formulation:

Existing work is based on graph. Lot of applications generates graph dataset. Such graph dataset cannot be efficiently handled by the traditional relational database systems. Such dataset is stored in xml or graph format. User will not be able to fire structural query like sql on such graph dataset.

In literature work, lot of system were focuses on keyword based search on graph dataset. This keyword search systems were extract the results by matching graph node values. But such keyword based queries were unable to handle temporal aspects of graph dataset.

Temporal graph snapshots were proposed in literature to deal with temporal graph based queries. These graph snapshots were stored and processed separately. Such applications were require high storage space and retrieval time for keyword based temporal queries.

All graph snapshots are stored together to overcome these problems. This improves storage and execution efficiency. Custom query syntax provide sorting and ranking function. But these operations are not sufficient to answer user's aggregate queries. Along with the collecting temporal answers from entire graph dataset, system should provide sorted and formatted archived temporal result using ranking and aggregate functions such as sum ,count, avg, etc like sql queries. To improve system efficiency the predicate filters are applied along with the query keyword filter in proposed system.

IV. System Architecture:

Following figure 1 represents the architecture of system.

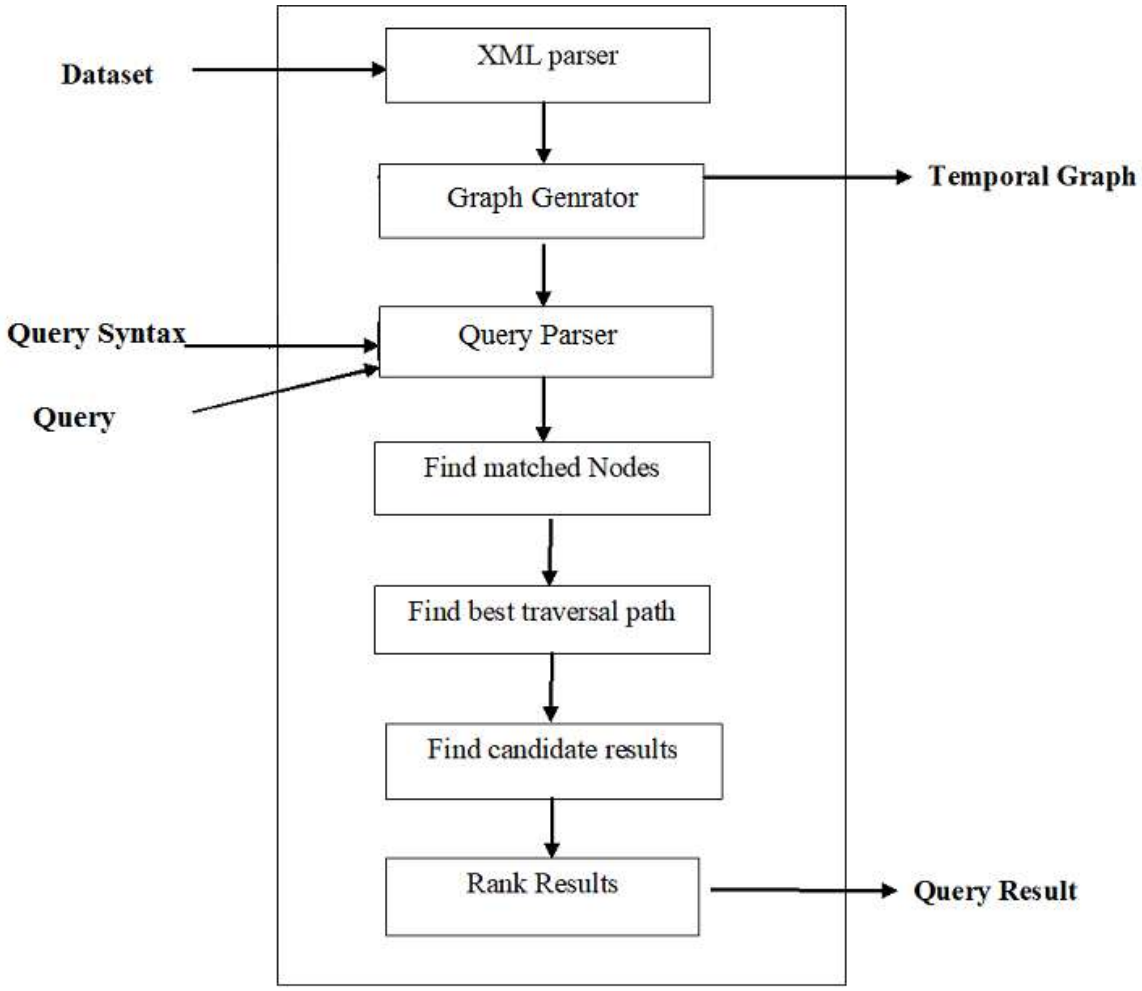


Fig 1.System Architecture

The proposed system provides a solution for temporal graph searching. User can fire a temporal query on graph dataset. The temporal information is stored in a graph at every node and edge. The temporal information of a dataset is in terms of time intervals. A graph contains node and edges labeled with one or more time intervals. Custom query syntax is proposed in this system. User can fire a query on graph using this syntax. System parses the query. If query is syntactically correct then it finds the desired nodes present in a graph using graph traversal method. System extracts the all possible results set as a candidate result. To generate archived result set ranking functions, sorting functions and aggregate functions are applied on these candidate results and final result is get generated.

Xml dataset and user query is input to the system. System filters the xml dataset and generates the graph. Initially from this dataset temporal graph is generated. User has facility to fire a keyword query over a graph. After analyzing the query requirement graph nodes are filtered and desired result is generated. The processing is done in the following faction:

1: Graph Generation

A directed graph is generated based on input dataset. In this graph node and edges are marked with one or more time intervals.
Node is represented using its name or tag or by its value. A graph may have weighed graph where each edge has weighed assigned to it.

2: Query Processor

Custom query syntax is generated for temporal graph searching. The Syntax has following format:

<keyword> + <Predicate>*+ <ranking Function>?

Where,

<Keyword> represents one or more keyword list
<Predicate> represents zero or more temporal predicate
<Ranking function> zero or one represents the ranking order

3: Best Path Iterator

To find best shortest path between query keyword matched modes dijkstras algorithm is proposed.

Since node may have different distant at different time instant, there is need to preserve information time varying node information. To preserve such information NTD triplets are used. It includes Node, time Interval and Distance value.

4:Priority Queue generator

Priority queue preserves the NTD triplet information and its key is the duration value. Best path iterator selects the next NTD triplet value. The selected NTD triplet is added in priority queue. The sorting of this queue is depend on the given ranking function.

5: Candidate Result Generation

A tree is generated from root node n to the matched nodes with the selected priority queue NTD triplets. It creates a set of NTD triplet nodes. This candidate result is more than one combination of required query answers.

6: Apply ranking

Candidate results are get filtered using ranking function mentioned the query. Data will get sort using ranking function for final result generation.

A. Algorithm: Searching Temporal Graphs:

Input: data G,
inverted index,
keywords k1, k2,..,Km,
predicates P,
ranking function Rf
aggregate function Agg

Output: Result R

Processing:

//keyword Extraction

1. Extract matching query matching nodes from graph G using Inverted Index and add it to M
   //Temporal predicate filtering on keyword matches
2. Initialize empty priority Queue pq
3. For each node s in M
   Check if s matches the condition in predicate P then
   Apply best path iterator and add result in priority queue pq
4. Initialize empty result set R
   //Best path iterator from each keyword match for path expansion
5. For all nodes in pq
6. expand node using Dijkstras algorithm
7. update neighbors of nodes present in pq
8. Create NTD triplet and add it in NTD set
   //Result generation based on the best paths, with validity checking
9. For each NTD triplet originating from query keyword k
10. T: Find smallest tree from root n to at least 2nodes from NTD triplet
11. Add T in candidate results
12. Qualify the result with predicate P and add in result set
13. Apply ranking and sorting over data
14. Apply aggregation over result set
15. Display Result

B. Best Path Iterator Algorithm:

Algorithm steps:

Input: source s,
ranking function f
Predicate condition p

Output: Filtered node list in Priority queue pq ,
Aggregation result

Processing:

1. Initialize priority queue pq = EMPTY
2. Initialize Gropuby table =EMPTY
3. If predicate matches node s temporal value
4. Create NTD triplet(s, val(s), weight(s))
5. For each node in pq
6. (n,T ,d) = Get NRD triplet from pq
7. if (n,t) is not visited and check ranking function
8. visited(n,t) =true
9. update d as w(prevn + currn) + d
10. check aggregate function is not NULL
11. get visited node value val(n)
12. if group by function is not NULL
    if val(n) present in Group by
    update Groupby(val(n)) = Groupby(val(n))+aggregation(val(n))
    else
    add Groupby(val(n)) = aggregation(val(n))
    aggregation result = Groupby
13. else apply aggregation over aggregation(val(n))
    aggregation result = aggregation(val(n))

C.Ranking function

Algorithm steps:

Input: prioirity queue pq,
PrevNTD triplet

Output: updated priority queue

Processing:

1. Generate bit vector B =0
2. Initialize i =0, k =0
3. For each node in n in pq
4. If n' T matches the ranking criteria
   B[i] = 1
   Increment i
5. If at least at 1 position of B contains 1 then
   Delete NTD from pq
6. Add NTD at position k
7. Increment k

Datasets:

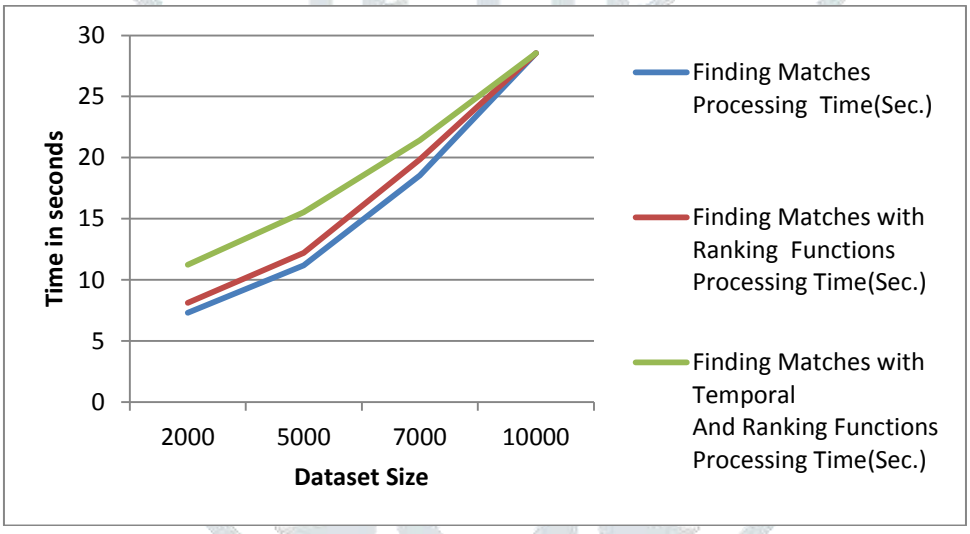
For system testing 2 datasets are used :

1. DBLP

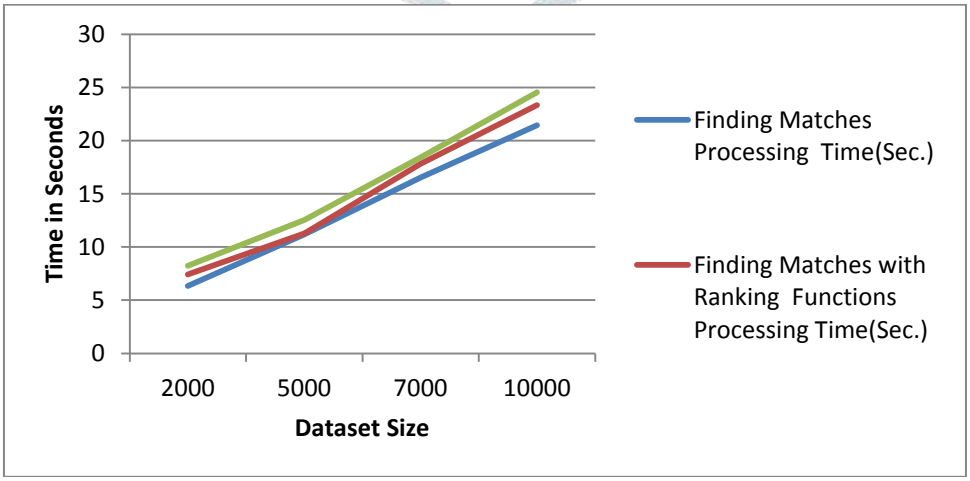
DBLP dataset is in XML format. DBLP dataset contains the temporal information with 53 time instant one for each year.

V. Results and Discussion :

Following graph shows the processing time for different datasizes(number of nodes) and this work is based on existing system.



Following graph shows the processing time for different datasizes(number of nodes) and this work is based on using new approach(Proposed System).



Conclusion:

In this research work a temporal graph based search mechanism is used. To fire a query on graph a custom query syntax is used. For user convenience a custom query generation panel is defined. This panel helps to user to select query keywords, wildcard query keywords, temporal predicate, ranking function and aggregate function conditions. The system finds the results on temporal graph and generates a sub-graph containing all matching nodes. To improve system efficiency the predicate filters are applied along with the query keyword filter. The system efficiency is checked with respect to different queries,The number of results and time of excection are used as a performance measure parameter. In current system Dijkstras algorithm is used for path traversal.if temporal nodes weight in a graph is negative then Dijkstras algorithm fails to give the results. To overcome this drawback,in future system can be implemented using Bellman-Ford algorithm for shortest path traversal.

Refrences :

[1] Ziyang Liu, Chong Wang, Yi Chen, ”Keyword Search on Temporal Graphs,” in IEEE Transactions on Knowledge and Data Engineering Vol.29, No: 8, Aug. 2017

[2] C. S. Jensen, R. T. Snodgrass, and M. D. Soo, ”The TSQL2 Data Model”,In The TSQL2 Temporal Query Language, 1995.

[3] F. Rizzolo and A. A. Vaisman, ”Temporal XML: Modeling, Indexing, and Query Processing”, VLDB J., 17(5):11791212, 2008.

[4] K. Golenberg, B. Kimelfeld, and Y. Sagiv, ”Keyword Proximity Search in Complex Data Graphs”, In SIGMOD Conference, pages 927940, 2008.

[5] B. Kimelfeld and Y. Sagiv,”Finding and Approximating Top-k Answers in Keyword Proximity Search”, In PODS, 2006.

[6] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, ”Keyword Searching and Browsing in Databases using BANKS”, In ICDE, pages 431440, 2002.

[7] R. T. Snodgrass, ”The Temporal Query Language TQuel”, in ACM Trans.Database Syst., 12(2):247298, 1987.

[8] C. S. Jensen, R. T. Snodgrass, and M. D. Soo, ”The TSQL2 Data Model”,In The TSQL2 Temporal Query Language. 1995.

[9] Tsql2 and sql3 interactions. http://www.cs.arizona.edu/people/rts/sql3.html.

[10] G. Koloniari, D. Souravlias, and E. Pitoura, ”On Graph Deltas for Historical Queries”, CoRR, abs/1302.5549, 2013.

[11] A. Fard, A. Abdolrashidi, L. Ramaswamy, and J. A. Miller, ”Towards Efficient Query Processing on Massive Time-evolving Graphs”, In CollaborateCom, pages 567574, 2012.

[12] C. Ren, E. Lo, B. Kao, X. Zhu, and R. Cheng, ”On Querying Historical Evolving Graph Sequences”, PVLDB, 4(11), 2011.

[13] W. Huo and V. J. Tsotras, ”Efficient Temporal Shortest Path Queries on Evolving Social Graphs”, In SSDBM, pages 38:138:4, 2014.

[14] B. Ding, J. X. Yu, and L. Qin, ”Finding Time-Dependent Shortest Paths over Large Graphs”, In EDBT, pages 205216, 2008.

[15] M. Steinbauer and G. Anderst-Kotsis, ”DynamoGraph: A Distributed System for Large-scale, Temporal Graph Processing, its Implementation and First Observations”, In WWW, 2016.

[16]P.Sonawane,R.Dahake “Survey On Graph based Searching Techniques”, In Journal of Emerging Technologies and Innovative Research (JETIR) www.jetir.org , 2018 JETIR June 2018, Volume 5, Issue 6.

[17]P.Sonawane,R.Dahake “Graph based Keyword Search using Temporal Predicates”,In cPGCON 2018,April 5.