International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 06 | June-2016 www.irjet.net p-ISSN: 2395-0072

A Novel Approach for Effective Search on Graphs with Weighted Attributes

B.Vigneswari

Dept. of Computer Science
Shrimati Indira Gandhi College, TamilNadu, Trichy

Abstract - *Many real world network problems often concern multivariate nodal attributes such as image, textual and multi-view feature vectors on nodes, rather than simple univariate nodal attributes. This paper focus on large graphs where nodes have attributes, such as a social network where the nodes are labeled with each person's job title. In such a setting, we want to find subgraphs that match a user query pattern. For example, a 'star' query would be, "find a CEO who has strong interactions with a Manager, a Lawyer, and an Accountant, or another structure as close to that as possible". Such graphs are called weighted attribute graphs (WAGs). Nodes of a WAG exhibit multiple attributes with varying, non-negative weights. WAGs are ubiquitous in real-world applications. For example, in a co-authorship WAG, each author is a node; each attribute corresponds to a particular topic (e.g., databases, data mining, and machine learning); and the amount of expertise in a particular topic is represented by a non-negative weight on that attribute. A ranking function which unifies ranking between the graph structure and attribute weights of nodes is proposed in this paper. Further proposed a fast and effective top-k graph search algorithm for WAGs.*

Key Words: **Weighted attribute graph, graph search, top-k algorithms**

1.INTRODUCTION

A graph is a non-empty finite set V of elements called vertices together with a possibly empty set E of pairs of vertices called edges. Here are a few examples of graphs:

Vertex set V={a, b, c, d} and edge set E={(a, b), (b, d)}

Vertex set V={1, 2, 3, 4} and edge set E={(2, 4)}

Vertex set V={wolf, goat, cabbage} and edge set E={(wolf, cabbage)}

Vertex set V={A, B, C} and edge set E={}

Graphs provide a natural way for representing entities that are "connected" (e.g., authors are connected by their co-authorships). A graph is commonly denoted by *G=(V,E)* , where V is the set of nodes or vertices and E is the set of links or edges. This paper focus on graphs that have multiple attributes (e.g., an author has multiple areas of expertise) and (2) attributes on nodes have non-negative weights associated with them (e.g., an author has varying degrees of expertise across different topics). Such data graphs are called weighted attribute graphs (or WAGs for short). WAGs are ubiquitous in real-world applications. Examples include (1) co-authorship networks with multiple expertise as attributes, (2) friendship networks with various activities (such as liking, commenting, or clicking on different types of posts) as attributes, (3) communication networks with various connection types as attributes, etc. Moreover, the outputs of mixed-membership community discovery and role-discovery algorithms are WAGs. Such algorithms take as input a graph and assign multiple communities (or roles) to nodes, with varying memberships—i.e., they output WAGs. Note that graph search is different than graph clustering even though both combine structure and node attributes.

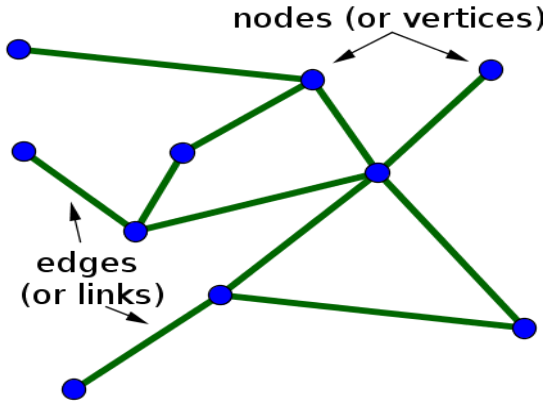

Fig 1. Graph with Edges & Vertices

In many modern problems, studying a network of entities with multiple attributes rather than a simple univariate attribute is interesting. For example, when an entity represents a person in a social network, it is widely accepted that the nodal attribute is most naturally a vector with many personal information including demographics, interests, and other features, rather than merely a single attribute, such as a binary vote as assumed in the current literature of social graph estimation based on Markov random fields (Banerjee et al., 2008, Kolar et al., 2010). Existing algorithms and theoretical analysis relies heavily on covariance selection using graphical lasso, or penalized pseudo-likelihood. They can not be easily extended to graphs with multi-variate nodal attributes. An important task on WAGs is to quickly and effectively answer a user's search, where the search can include (1) constraints on the connectivity of nodes and (2) constraints on the weighted attributes. Consider the case where a WAG is produced by a mixed-membership community-discovery algorithm. Being able to search that output enables us to gain more insight into the data and its patterns.

2. EXISTING SYSTEM

Graph querying is widely adopted to retrieve information from emerging graph databases, e.g., knowledge graphs, information and social networks. Given a query, it is to find reasonable top answers (i.e., matches for the query) from a data graph. Searching real-life graphs, nevertheless, is not an easy task especially for non-professional users. (1) Either no standard schema is available or schemas become too complicated for users to completely possess. (2) Graph queries are hard to write and interpret. Structured queries (e.g., XQuery and SPARQL) require the expertise in complex grammars while keyword queries can be too ambiguous to reflect user search intent. Moreover, most of these methods adopt predefined ranking model, which is barely able to bridge the gap between the queries and the true matches. (3) Moreover, it is a daunting task for users to inspect a large number of matches produced from querying large-scale heterogeneous graph data.

Drawbacks of Existing System

Due to variations in graph characteristics and application requirements, graph matching is not a single problem, but a set of related problems.

3. PROPOSED SYSTEM

The proposed system has three components: I-Graph, S-Graph, and R-Graph. Given a graph G = (V, E) and a matrix W whose wij entry corresponds to the weight of attribute j on node i, I-Graph builds a hybrid index on the graph that incorporates both the weighted attributes and the structure of the network. Upon receiving a search query, S-Graph utilizes the output of I-Graph to quickly retrieve the best k matches based on both weighted attributes and structure. R-Graph ranks the results by utilizing a novel ranking function that sums the divergence scores of the nodes in the solution. S-Graph supports range and point queries on the weighted attributes.
The contributions are as follows:

- **Problem:** Initiate the study of graph search on WAGs.
- **Flexibility:** Introduce a flexible querying mechanism on WAGs, and a novel ranking technique, R-WAG, which unifies the ranking of both structure and attribute weights into a single measure.
- **Efficiency:** First prove that finding the optimal match for a given graph search on a WAG is NP-complete. Next, introduce S-Graph for search on WAGs. S-Graph utilizes I-Graph's novel hybrid indexing scheme. I- Graph unifies a WAG's attribute weights, structural features (e.g., vertex degree), and graph structure into one structure that is easy to search. S- Graph uses a novel algorithm that is efficient and returns the "optimal" answer to a query in terms of its quality given the ranking function in R- Graph.
- **Effectiveness:** Demonstrate S- Graph's effectiveness and broad applicability through extensive case studies on real world data and comparisons.

4. PROJECT DESCRIPTION

i)Defining Attributes

Attributes are associated values belonging to a graph, vertices or edges. These can represent some property, like data about how the graph was constructed, when the graph is plotted, or simply the weights of the edges in a weighted graph.

Arguments

Name : Character constant, the name of the attribute.
Index : Numeric vector, the ids of the vertices or edges. It is not recycled, even if value is longer.

Value : Numeric vector, the new value(s) of the attributes, it will be recycled if needed.

Details

There are three types of attributes in graph: graph, vertex and edge attributes. Graph attributes are associated with graph, vertex attributes with vertices and edge attributes with edges.

ii)Weighted Attribute Graph Creation

A graph query on a WAG is a subgraph, which includes constraints on both connectivity and nodal-attribute weights. One extreme scenario of a graph query is one that only contains connectivity specifications between the participating nodes and no nodal-attribute annotations. The other extreme assumes that, in addition to connectivity specifications, a graph query also contains the attribute-weight specifications for all the query nodes. S- Graph offers a wide range of queries by supporting these extreme cases; and also those queries where only a subset of query nodes contain weights on a subset of attributes. In addition, the attribute weights can be specified as discrete values or as ranges

iii)Ranking

Given a graph query (point or range), the main task is to return its top-k best matches. To do so, need to rank the results considering divergence on the graph structure and on the weighted attributes.

A key issue is how to select a ranking function by determining reasonable weights for the transformations. Instead of assigning equal weights or calibrating the weights by human effort, it automatically learns the weights from a set of training instances. Each instance is a pair of a query and one of its relevant (labeled as "good") answers. The learning aims to identify for each transformation a weight, such that if applied, the model ranks the good answers as high as possible for each instance. It begins with a cold-start stage where no manual effort is required for instances labeling, and can be "self-trained" in the warm-start stage, using user feedback and query logs. When no user query log is available, it randomly extracts a set of subgraphs from the data graph as "query templates." It then injects a few transformations to the template and generates a set of training queries. Intuitively, each training query should have the corresponding templates as its "good matches," since the query can be transformed back to the template

iv)Searching

It efficiently finds top matches, leveraging approximate inferencing. It treats a query as a graphic model and the matches as assignment to its random variables (query nodes). By propagating messages among the nodes in the graphical model, the inference can identify top assignments (matches) that maximize the joint probability (ranking score) for the model. Together with a graph sketch data structure, this technique dramatically reduces the query processing time, with only small loss of match quality (less than 1% in our validation).

v)Top-K

Top-k Pattern Matching in a WAG: Given a WAG, a pattern query (point or range), and an integer k, identify a set of k subgraphs from the WAG such that, (i) the k subgraphs are ranked in ascending order of their overall divergence score (F) to the pattern query; (ii) any subgraph not present in the set has a larger divergence score with the pattern query than the k-th subgraph's overall divergence score

The top-k graph querying intends to quickly identify the answers of high-quality w.r.t. a ranking function. Efficient top-k querying is a well-studied problem in relational databases, XML , and RDFs . In contrast to these class techniques, knowledge graph querying has its unique properties and comes with new challenges and opportunities: (1) The matching score that combines multiple similarity measures has to be calculated and ranked online, (2) query answers are often inexact, and (3) query graphs are usually small.

5. CONCLUSION

Proposed weighted attribute graphs, which can model a wide range of data arising in diverse scenarios and applications; and study the problem of graph search on WAGs. Introduce a three-part approach—consisting of R-Graph, I- Graph, and S-Graph —that can perform fast best-effort search on WAGs. I- Graph produces a novel hybrid index structure that incorporates weighted attributes, structural features, and the graph structure. S- Graph uses a novel algorithm to efficiently return the best answers to a graph query. R- Graph ranks the results based on a unified measure of both weighted-attribute and graph-structure divergences.

REFERENCES

[1] S. Agrawal, S. Chaudhuri, and G. Das, "DBXplorer: A system for keyword-based search over relational databases," in Proc. 18th Int. Conf. Data Eng., 2002, pp. 5–16.

[2] N. N. Dalvi and D. Suciu, "Efficient query evaluation on probabilistic databases," Int. J. Very Large Data Bases, vol. 16, no. 4, pp. 523–544, 2007.

[3] W. Fan, J. Li, S. Ma, N. Tang, Y. Wu, and Y. Wu, "Graph pattern matching: From intractable to polynomial time," Proc. VLDB Endowment, vol. 3, pp. 264–275, 2010.

[4] B. Gallagher, "Matching structure and semantics: A survey on graph-based pattern matching," in Proc. Amer. Assoc. Artif. Intell. Fall Symp., 2006, pp. 45–53.

[5] Y. Tian and J. M. Patel, "TALE: A tool for approximate large graph matching," in Proc. IEEE 24th Int. Conf. Data Eng., 2008, pp. 963– 972.

[6] H. Tong, C. Faloutsos, B. Gallagher, and T. Eliassi-Rad, "Fast besteffort pattern matching in large attributed graphs," in Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2007, pp. 737–746.

AUTHOR

B.Vigneswari M.Phill (CS)
Research Scholar,
Shrimati Indira Gandhi College,
Trichy.