An Integrated Hybrid Recommendation Model Using Graph Database

Angira Amit Patel
Shri Chimanbhai Patel Post Graduate Institute of Computer Applications.
Ahmedabad, India
angira.it@gmail.com

Dr. Jyotindra N. Dharwa
AMPICS, Ganpat University. Ganpat Vidyanagar, Mehsana,
India
jyotindra.dharwa@ganpatuniversity.ac.in

Abstract— **Graph database is revealed as alternative of traditional relation database for the reason that graph is flexible and self-explaining structure, which can cope with any kind of complex structure. Recently, graph database is extensively used to represent specially multi linked data of web, RDF data, social network, chemical structure, gene, network structure, publication links, and many more. This research illustrates potential use of graph database for recommendation system along with its convenience to develop hybrid recommendation system. Here, use of property graph model is demonstrated for solution of state-of-affairs hybrid recommendation system, complexity of beneath integrated data structure and required course of actions. This investigation anticipates an appropriate use of graph database to integrate various recommendation algorithms like content based recommendation; utility based recommendation as well as knowledge based recommendation. The main intention behind development of hybrid model is that helps in solving challenges of real world like cold start and many more. This research provides complete guidelines to anyone who wants to implement graph database for recommendation system along with various recommendation algorithms.**

Keywords— Graph database; Recommendation System; Hybrid Recommendation System; Collaborative Recommendation System; Content based Recommendation System; Knowledge Based Recommendation System; Utility Based Recommendation System

I. Introduction

Recommender systems (RS) is very useful tool which guaranties that right information are serve to right users at right time [6]. RSs are useful in many domains, such as web personalization, information filtering and e-commerce. One of the most popular areas of applications for recommender systems is web environment personalizing by providing a list of items related to user's interests for example providing recommendations of books, movies, music and items to purchase. Various types of recommendation techniques have been purposed such as non-personalized, content-based, collaborative filtering, utility-based, knowledge-based and hybrid. Hybridization of various recommendation techniques has been carried out to combine multiple techniques to improve performance and to remove limitations of underline techniques.

The most common issues associated with development of recommendation system is cold-start, grey sheep, sparse data, privacy, trust, scalability, synonymy, integration, domain specific issues and appropriate selection of hybridization techniques [7].

One of the most important research problems is how to integrate external knowledge for improving performance of content-based recommendations. Content features are usually available for both users as well as items. Typically, for users their demographics analyzed. For items its related attributes considered for similarity matching [2]. Say for example, movies database it may include the actors, genre, directors, country of release, etc. Other item like restaurants, these may include the location, cuisine, formal vs. casual etc.

This paper proposes a hybrid recommender system which provides Top-N recommendations for currently online user on the basis of active user's requirement preferences and accumulated knowledge using graph database. The main idea behind use of graph database is it allows integrated storage of various data about items, user's preferences as well as knowledge graph. In short, graph database and various graph traversing techniques are used as an instrumental tool to perform various analytical operations for generating Top-N recommendations.

A graph database [20] uses graph structures to represent and store data which is basically collection of nodes, edges, and properties. A graph is flexible structure, which naturally allows integration of multiple entities all together. That facilitates merging of various RS techniques like utility based recommendation, knowledge based recommendation as well as content based recommendation.

In summary, the contribution of this paper includes: (i) The hybrid RS which integrate user preference based recommendation, knowledge management, content based recommender systems using graph database. (ii) Formulation of the knowledge based recommendation which incorporate user preferences. (iii) A proposed method for graph modeling which naturally integrate various heterogeneous data elements and allows efficient computation. Experiments have been conducted to verify the efficiency of the proposed model with synthetic data set.

This paper organized in three main sections as follow: the first section describe related work in the area of recommendation systems in current scenario. The second section includes motivation behind use of graph database for recommendation system. As graph is flexible and versatile structure having many characteristics, which makes it suitable to represent any complex data structure of real world RS. The third section covers description of our model and algorithm.

II. Related Work

Various recommendation systems have been popular for personalization for long time now but, still few areas are considered as research topic. Knowledge-based recommendation system suggests recommendations based on some knowledge and opinions accumulated from experts. There could be three types of knowledge involved into such a system: catalog knowledge, function knowledge and user knowledge [11]. Recently, personalized RS proposed that analyzed content and the interconnections between the content itself and external knowledge sources referred as a knowledge graph [14]. It has been observed that, there have not been much efforts in directions that using external knowledge for improving performance of content- based recommendations and it is the motto of our work.

Recommender systems typically generate a list of recommendations based on user histories without taking any input relate to preference from users. In this way, users can only passively receive recommended results without any chance to alter or adjust the results [7]. However, in practice, users usually have multiple intentions and may like to actively input their interests to obtain different recommendations. Say for example, one user was purchasing books for him yesterday but, today he is interested to buy books for his kids. To cope up with this issue, utility based recommendation techniques have been evolved which suggests few products based on usefulness of the each product for particular need of the active user [9]. Although many methods have been proposed in the past [8, 10, 13] in this direction each one has different methodology.

Recently, query-based recommendation [5] proposes that allow a user to specify his/him search intention using a novel approach called “heterogeneous preference embedding". The proposed method constructs a user-item preference network from user access logs, in which the vertices represent users and various items and the edges are the relationships between the users and the items. After the construction, an edge-sampling technique is applied to embed the large information network into low-dimensional vector spaces, in which the proximity information of each vertex is encoded into its learned vector representation. Afterwards, that uses the learned representations of vertices with some simple search methods or similarity calculations to conduct the task of query-based recommendation.

The feature-based RS [12] is also purposed that works on the principal of “The people who bought / like a product with X features also like other products with features X”. This technique has been proven effective for frequently changing product catalogues, product catalogue with custom products. In this work, the product similarity has been done based on features of the products.

A graph-based RS has been also proposed that uses using weighted undirected graph to generate recommendations based on similarity between elements [3]. The two layer graph model has used to perform hybrid model which incorporate content based and collaborative-filtering approach for digital library [1]. For graph based RS random walk based scoring algorithm is proposed [4] to rank products/items for every active user according to its expectations.

Our proposed approach is slight similar to reference [15], that uses fuzzy linguistic terms for preference representation and describe hybrid model incorporated knowledge based, utility based and content based recommendation. Our approach is different from this in way of data modeling techniques and computational techniques.

III. Proposed Model

A. Motivation

A graph database management system enables managing data which modeled as a graph. In graph data modeling nodes are entities and edges are relations between entities. This data can be handling through graph oriented operations [20]. Among the existing systems, various systems exists AllegroGraph [24], InfiniteGraph [21], Neo4j [16] and Sparksee [22], which uses a different graph models for data storage.

Our research work make use of property graph [23] model in which nodes can be heterogeneous, nodes and edges both may contain attributes and all edges are directed. Property graph is a very expressive graph which stores attributes related to entities and/or relations. Another unique feature of this model is flexibility that allows representation of multiple entities together and allows integrity among them. It has been proven that the graph database is most suitable structure for associative operations [26] as it becomes very easy to access all neighboring nodes for given particular node using simple graph traversing algorithm. The performance of associative operation is directly in the proportion of only the number of nodes associated with that given node [26]. In short, graph traversals are the executed with constant speed independent of total size of the graph [26]. This characteristic of graph database makes it most suitable structure of RS. The experimental study of performance measurement of relational database (MySql) against graph database (neo4j.org) was conducted. After series of experiments, it has been proven than graph database is faster than relational database when traversal length is more than two [27].

B. A Multi -Layer Graph Model

The unique idea behind proposed model is use of graph model which allows integration of heterogeneous nodes, labeled nodes and labeled edges and directed edges. The multi-layer graph model is used to represent as shown in Figure 1.

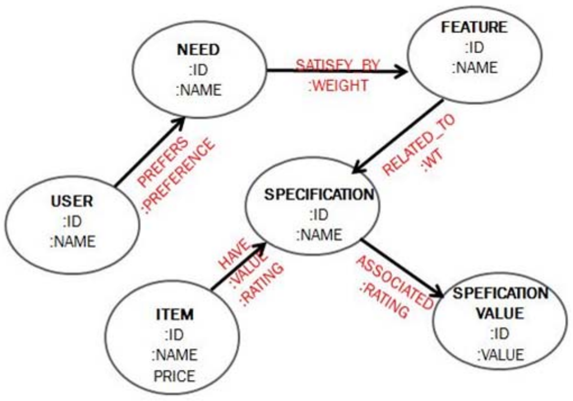

Fig. 1. Multi-Layer Graph Model

The layer 1 comprised of all nodes that store details various users. The relationship among users could establish and functioned for user-user similarity matching. The layer 2 comprised of all nodes related to various needs and its details. The layer 3 comprised of all nodes for various features and its related details. The layer 4 comprised of all nodes related to various items specifications and its associated details. The layer 5 comprised of all nodes related to various items and its associated details. The construction of layer 2, 3 and 4 can be carried out based of preoccupied knowledge and hence it is referred as knowledge graph, which is shown in figure 2.

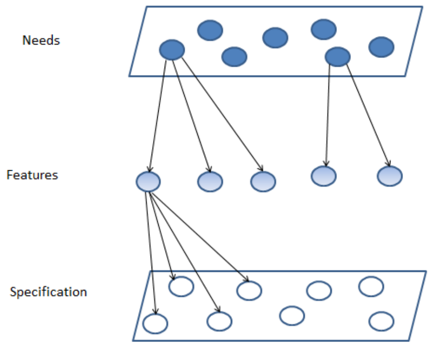

Fig. 2. Knowledge Graph

The need may be correlated with one or many features and this is represented as edges. The weight is allotted to each edge. All this directed and labeled edges provide connection between layer 2 and layer 3. Same way, one feature may be interrelated with many product specifications and this is representing by edges between layer 3 and layer 4. Same way, weights are assigned to each one. The item may have many specifications and this is represented as edges between layer 4 and layer 5. The relationship among items could establish and functioned for item-item similarity matching.

C. System Flow

The computational methodology is shown in the form of system flow in the following figure 3 which is self-expressive. This shows, integration of all various recommendation techniques like utility based, knowledge based and content based into system.

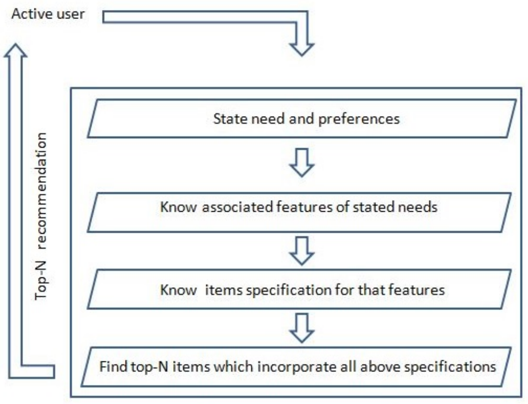
Active user
Top-N recommendation
State need and preferences
Know associated features of stated needs
Know items specification for that features
Find top-N items which incorporate all above specifications

Fig. 3. Computational Methodology

D. Recommendation Algorithm

The steps of recommendation algorithm are summarized as follows. The initial two steps could be preprocessed to improve efficiency of algorithm.

1. Construct knowledge graph (N, F, S)
2. Store catalog of all items along with attributes as P.
   P = $\{p_1, p_2, p_3,\ldots..p_p\}$
   Process each item and calculate rating of specifications based on specification values and its associated rating.
3. Create node for active user U. Active user U states all needs along with preferences (binary). Based on it construct relation among user and needs nodes and specify preference as edge label.
4. Search given a set of needs N for active user U which satisfied above preference.
   N = $\{n_1, n_2, n_3,\ldots..n_n\}$
5. Find set of all associated features F for given set of needs N
   F = $\{f_1, f_2, f_3,\ldots..f_m\}$
6. Find set of all correlated specifications S for given set of features F
   S = $\{s_1, s_2, s_3,\ldots..s_q\}$
7. Find set of items R which satisfies all product specification as per S. Sort R. Find top-N matching items within R and recommend to active user.

IV. EXPRIMENTS

Finally, the proposed model is implemented with neo4j graph database which supports property graph model and Cypher query language which works based on pattern matching. The synthesis data set is used for model evaluation.

A. Knowledge Graph Construction

The step 1 of algorithm implemented to construct knowledge graph, as shown below:

```
LOAD CSV WITH HEADERS FROM "http://.../Need.csv" AS row
CREATE (n:Needs {NeedID: row.NeedID, NeedName: row.NeedName})
RETURN  count(n);

LOAD CSV WITH HEADERS FROM "http://.../Features.csv" AS row
CREATE (f:Feature {FeatureID: row.FeatureID, FeatureName:
row.FeatureName })
RETURN count(f);

LOAD CSV WITH HEADERS FROM "http://.../NeedFeatureMapping.csv"
AS row
MATCH (n:Needs) WHERE n.NeedID = row.NeedID
MATCH (f:Feature) WHERE f.FeatureID = row.FeatureID
CREATE (n) - [s:SATISFY_BY { WEIGHT : toFloat(row.Weight)}] -> (f)
RETURN  count(s) ;

LOAD CSV WITH HEADERS FROM "http://.../Specification.csv" AS row
CREATE (s:Specification {SpecificationID: row.SpecificationID,
SpecificationName: row.SpecificationName})
RETURN  count(s);

LOAD CSV WITH HEADERS FROM
"http://.../FeatureSpecificaMapping.csv" AS row
MATCH (s:Specification) WHERE s.SpecificationID = row.SpecificationID
MATCH (f:Feature) WHERE f.FeatureID = row.FeatureID
CREATE (f) - [r:RELATED_TO { WT : toFloat(row.Weight)}] -> (s)
RETURN  count(r) ;
```

B. *Product Catalog Preparation*

On the basis of product catalog available for online shopping, it prepares all nodes for items. Various specifications of products should be map with each product and makes all product node ready for further search operation, as shown below.

```
LOAD CSV WITH HEADERS FROM "http://.../Product.csv" AS row
CREATE (p:Product {ProductID: row.ProductID, ProductName:
row.ProductName, Price:toFloat(row.Price), ProductRate:toFloat(row.Rating)
}) RETURN  count(p) ;
LOAD CSV WITH HEADERS FROM "http://.../ProductSpecification.csv"
AS row
MATCH (p:Product) WHERE p.ProductID = row.ProductID
MATCH (s:Specification) WHERE s.SpecificationID = row.SpecificationID
CREATE (p) - [h:HAVE] -> (s)
RETURN  count(h) ;
```

C. *Algorithm Implementation*

For active user create node and represent its need preferences (step 3). A few sample code of Cypher query language is presented here that shows step 3 of algorithm as shown bellowed:

```
LOAD CSV WITH HEADERS FROM "http://.../ User.csv" AS row CREATE
(u:User {UserID: row.UserID, UserName: row.UserName})
RETURN  count(u);

LOAD CSV WITH HEADERS FROM "http://.../UserNeedPrefence.csv" AS
row MATCH (u:User) WHERE u.UserID= row.UserID
MATCH (n:Needs) WHERE n.NeedID= row.NeedID
CREATE (u) - [r:PREFERS {Preference: toFloat(row.Preference)}] -> (n)
RETURN  count(r) ;

MATCH (u:User {UserID : "1" }) - [p:PREFERS {Preference : 1}] ->
(n:Needs) - [s:SATISFY_BY] -> (f:Feature) - [r:RELATED_TO] ->
(sp:Specification) WITH u,n,f,sp, count(sp) AS total
RETURN u.UserName, n.NeedName, f.FeatureName, sp.SpecificationName,
total
```

For active user "1", entire knowledge graph can traverse using graph traversing operations (step 4 to 6) as follow.

```
MATCH (u:User {UserID : "1" })-[p:PREFERS]->(n:Needs)-
[s:SATISFY_BY]->(f:Feature)-[r:RELATED_TO]->(sp:Specification) WITH
sp,
COLLECT ( DISTINCT sp.SpecificationName) AS RequiredSpecification
MATCH (p:Product { ProductID:"1" } )-[hh:HAVE]->(s:Specification)
WITH p,s,RequiredSpecification, count(*) AS Count,
COLLECT ( DISTINCT s.SpecificationName) AS Pspecification
WHERE RequiredSpecification  IN Pspecification
RETURN Count
```

Step 7 performs random walk in graph which search most suitable items as per specified specification criteria. And finally, sort generated recommendations and present to active user. That can be easily carried out using graph traversing along with specified criteria of preferences.

D. *Other Algorithm Implementation*

Other algorithms can be also possible to implement to generate recommendations using same graph model, if it could satisfy next stated conditions. First, "FRIEND_OF" relationship established among nodes of user within layer 1 based on available data. Second, all products are categorized based on content similarity. It has been shown in following figure.

```
MATCH (u:User {UserName : "Anjani" }) -  [:FRIEND_OF] -> (u1:User),
(u1) - [:LIKES] -> (p:Product),  (p) - [:Part_OF] -> (c:Category) <- [:Part_OF]
- (p1)
RETURN p1.ProductName, count(*) AS Occurrence
ORDER BY Occurrence DESC
LIMIT 10
```

In summary, it could possible to implement recommendations based on user-user similarity, item-item similarity, content based, user's preference based and knowledge based using proposed graph model. An integrated data model based on graph could serve as instrumental for various different algorithms of recommendations that is unique feature of this research work.

V. Evaluation

The recommendation algorithm is designed in such way that requires local graph search for most of the steps of algorithm. Many researchers has it already proven that local graph search operation is very efficient compare to all relational data models [26]. So, graph model always outperform in terms of efficiency compare to relation database. Our experimental result shows that for synthesis graph with 500 item catalogue, 10 need criteria, many associated features and many associated specification, it produce results in seconds. That proves suitability for online e-commerce environment. Still the evaluation work is in progress for evaluation of recommendation effectiveness with different criteria.

The same graph model could also utilize for implementation of content based and collaborative RS. So, our proposed graph model is proven effective to develop hybrid RS which integrate all various recommendation techniques.

VI. Conclusion And Future Work

A major advantage of the proposed method is its suitability of online environment as it is very efficient. Our graph model is flexible enough and it could be easy to incorporate various other nodes and relations into it whenever required. The proposed recommendation model incorporates expert knowledge and represent into it which is adaptable.

This work could extend for knowledge acquisition, representation and self-learning like neural network. The model could further evaluated using various criteria like novelty, coverage of new products, personalization in recommendation and other competence test of recommended items. Recently, fuzzy query has been proposed for cypher [17], so this model could extend for fuzzy preference.

References

[1] Huang, Z., Chung, W., Ong, T. H., & Chen, H. (2002, July). A graph-based recommender system for digital library. In Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries (pp. 65-73). ACM.

[2] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (pp. 285-295). ACM.

[3] Fouss, F., Pirotte, A., Renders, J. M., & Saerens, M. (2007). Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Transactions on knowledge and data engineering, 19(3), 355-369.

[4] Gori, M., Pucci, A., Roma, V., & Siena, I. (2007, January). ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines. In IJCAI(Vol. 7, pp. 2766-2771).

[5] Chen, C. M., Tsai, M. F., Lin, Y. C., & Yang, Y. H. (2016, September). Query-based music recommendations via preference embedding. In Proceedings of the 10th ACM Conference on Recommender Systems (pp. 79-82). ACM.

[6] Jafari, M., Sabzchi, F. S., & Irani, A. J. (2014). Applying web usage mining techniques to design effective web recommendation systems: A case study.Advances in Computer Science: an International Journal, 3(2), 78-90.

[7] Schafer, J. B., Konstan, J. A., & Riedl, J. (2001). E-commerce recommendation applications. In Applications of Data Mining to Electronic Commerce (pp. 115-153). Springer US.

[8] Park, H. S., Yoo, J. O., & Cho, S. B. (2006, September). A context-aware music recommendation system using fuzzy bayesian networks with utility theory. In International Conference on Fuzzy Systems and Knowledge Discovery (pp. 970-979). Springer Berlin Heidelberg.

[9] Huang, S. L. (2011). Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. Electronic Commerce Research and Applications, 10(4), 398-407.

[10] Huang, S. L. (2011). Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. Electronic Commerce Research and Applications, 10(4), 398-407.

[11] Burke, R. (1999, July). Integrating knowledge-based and collaborative-filtering recommender systems. In Proceedings of the Workshop on AI and Electronic Commerce (pp. 69-72).

[12] Han, E. H. S., & Karypis, G. (2005, October). Feature-based recommendation system. In Proceedings of the 14th ACM international conference on Information and knowledge management (pp. 446-452). ACM.

[13] Shih, Y. Y., & Liu, D. R. (2008). Product recommendation approaches: Collaborative filtering via customer lifetime value and customer demands.Expert Systems with Applications, 35(1), 350-360.

[14] Catherine, R., & Cohen, W. (2016, September). Personalized Recommendations using Knowledge Graphs: A Probabilistic Logic Programming Approach. In Proceedings of the 10th ACM Conference on Recommender Systems (pp. 325-332). ACM.

[15] Patel, A. A., & Dharwa, J. N. (2016, March). Fuzzy Based Hybrid Mobile Recommendation System. In Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies (p. 145). ACM.

[16] Neo4j web site. www.neo4j.org.

[17] Pivert, O., Smits, G., & Thion, V. (2015, August). Expression and efficient processing of fuzzy queries in a graph database context. In Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on (pp. 1-8). IEEE.

[18] Angles, R., & Gutierrez, C. (2008). Survey of graph database models. ACM Computing Surveys (CSUR), 40(1), 1.

[19] Deepayan, C., & Christos, F. (2006). Graph Mining: Laws, Generators, and Algorithms. ACM Comput. Surv., 38(1).

[20] Aggarwal, C. C., & Wang, H. (Eds.). (2010). Managing and mining graph data(Vol. 40). New York: Springer.

[21] InfiniteGraph web site. www.objectivity.com/infinitegraph.

[22] Sparksee web site. www.sparsity-technologies.com

[23] https://neo4j.com/developer/graph-database/

[24] AllegroGraph web site : http://allegrograph.com/

[25] Dominguez-Sal, D., Urbón-Bayes, P., Giménez-Vanó, A., Gómez-Villamor, S., Martínez-Bazan, N., & Larriba-Pey, J. L. (2010, July). Survey of graph database performance on the HPC scalable graph analysis benchmark. InInternational Conference on Web-Age Information Management (pp. 37-48). Springer Berlin Heidelberg.

[26] Singh, O. A COMPARATIVE STUDY OF NOSQL DATA STORAGE MODELS FOR BIG DATA.

[27] Rodriguez, M. A., & Neubauer, P. (2010). The graph traversal pattern. arXiv preprint arXiv:1004.1001.