Recommendation Based on Graph Algorithms

Dušan Zeleník, Mária Bieliková

Slovak University of Technology in Bratislava
Faculty of Informatics and Information Technologies
Ilkovičova 2, 842 16 Bratislava, Slovakia
`{zelenik,bielik}@fiit.stuba.sk`

Abstract. Media content recommendation is nowadays a common problem. Traditional algorithms based on collaborative filtering require an up-to-date dataset of users and their preferences, which is difficult to gather for huge database of items. Content-based approach suffers from the complex computation of similarity among items. In this paper we briefly describe our approach to recommendation with a focus on the natural change of user's interests in movies. We make use of a graph representation and experimented with modified graph algorithms. We design a representation of the data about movies in a graph structure and a method which uses our data model for recommendation. We discuss our findings with implemented algorithms which we experimentally evaluated with real users.

1. Introduction

In today's world there is a large amount of media content produced every day: movies, TV shows, TV programs etc. With so many options, choosing the right content can be difficult and overwhelming for the user. As for television, there are tens of channels from which the user can choose at any given moment. If he chooses the wrong channel, he misses something he would enjoy on a different one. The problem becomes even larger with more and more popular televisions or videos on-demand. The user has literally millions of possibilities from which to choose. The task becomes very difficult without a support which filters these possibilities.

Recommender systems analyze the taste, the mood or the context in which the user is at the moment. Based on the analysis, they create an accurate recommendation that suits the particular user [2]. There are various techniques used to create recommendations. The two main categories of recommendation systems are content based and collaborative. In both categories we should go through the entire entity base to find the correct item to be recommended. In collaborative approach this represents matrix of users and items, in content-based it is matrix of items and their similarity. Nevertheless, there is no guarantee that the estimation is correct and the recommended item is accurate enough for the user. Many recommendation systems try to recommend item by pairing the extracted knowledge base with the user's context and taste. As a result, the recommender systems may suffer from performance issues what makes them unusable in real time.

In this article we aim to design and evaluate a recommendation method that uses a new approach for recommending items with various attributes such as movies. Instead of the approaches mentioned above we decided to design a method which makes use of graph structure. We experimented with graph algorithms to provide comparison and their pros and cons. Graph-based recommender systems have been tested in the past and have shown promising results [3]. Our contribution is in comparison of graph algorithms and their modifications. We applied our method for multimedia recommendation (e.g. movies, TV shows and TV programs) in web application called Televido, which enabled us to experiment with various algorithms and real users.

2. Recommendation algorithms

Our recommendation algorithms are graph algorithms, which traverse the graph structure from initial nodes. The initial nodes represent user's interest. An initial node might be a movie or a genre the user likes. These initial nodes are either selected explicitly by the user as a query or implicitly based on the user model and the feedback from the user using standard methods. We actually do not need user model to be connected to nodes. We only need the list of items which are relevant for user and use it as a query on the fly.

The algorithms try to find the nodes which are the closest in the graph to all initial nodes, which are then returned as recommendations. There are multiple ways of looking at the problem of finding the closest nodes, especially in a very complex graph, which is why we designed and implemented four separate algorithms to compare.

Thanks to these parameters, the recommendation algorithms are quite versatile. For example, the same algorithm with different parameters can be used to recommend movies which are currently being shown at the cinemas or only the TV programs which will be on tomorrow night.

We particularly dedicate our effort to implement following algorithms:

- **Union colors.** The Union Colors algorithm is based on the basic Breadth-first search (BFS) graph algorithm.
- **Mixing colors.** The Mixing Colors algorithm is similar to Union Colors algorithm. These two algorithms differ in the way they deal with the representation and the meeting of colors
- **Energy spreading.** Algorithm is based on Spreading activation, which is a method for searching associative networks, neural networks, or semantic networks. Similar to previous algorithms, it is a variation on the simultaneous BFS algorithm.
- **Modified Dijkstra**. The Modified Dijkstra's Algorithm is, as the name suggests, based on the well known Dijkstra's algorithms for finding the shortest path in a graph. We implement a variation on this algorithm, which works for multiple initial nodes.

3. Experimental evaluation

The goal of the experimental evaluation we performed was to determine the best algorithm of the four algorithms we designed and to determine the accuracy of each algorithm. Based on the results of the experiment we intend to work on the most successful algorithms in the future and further modify them to make their results even better.

We filled the databases with a dataset consisting of information about a large amount of real movies and TV programs. In the end, the graph database consisted of roughly 165 000 nodes and 870 000 relations.

The experiment we used to test our methods was based on collecting explicit feedback from users. The user's task was to pick some initial nodes – movies, people or genres. The system generated 4 different sets of recommendations based on the initial nodes using the four recommendation algorithms we designed. We presented each user with 7 different scenarios, in each scenario the user was supposed to pick initial nodes in a different way, for example in the first scenario we asked him to pick two animated movies. The scenarios were implemented into the experiment to ensure the recommendation algorithms were tested on their versatility. After selection of the initial nodes each recommendation algorithm recommended five movies to the user. We showed the recommendation results of all algorithms to the user at once so that the speed of the algorithms would not bias the experiment. The user proceeded to order the algorithms based on their accuracy and was also asked to rate the most accurate algorithm on a scale from 1 to 5, 5 being the highest. In total, 30 users participated in the experiment. The users were mostly students in the field of informatics or other, non-technical fields. However, not all of the users who began the experiment completed every scenario. In total, 168 scenarios were completed by the users and recorded, which, on average, works out to 5.6 completed scenarios per user.

The results of the experiment are shown in Table 1. Here we can see comparison of the average ratings and positions. To discuss the results we need to add that Dijkstra and Energy Spreading are very expensive algorithms. These algorithms need relatively more time than other two proposed algorithms to compute results. The numbers in "Average rating" column mean the average rating from the cases when the algorithm was rated by the user (we only asked the users to rate the algorithm they picked as the first) and the number of times the algorithm was rated. "Average position" means the average position of the algorithms as sorted by the user, which means the lower the number the better.

Table 1. Results of experiments.

Algorithm	Average rating	Average position
Union Colors	2.322 out of 59	2.3095
Mixing Colors	2.9298 out of 57	2.2202
Energy Spreading	3.1892 out of 37	2.3095
Modified Dijkstra	3.1333 out of 15	3.1607

The results show that the users usually picked the Union Colors algorithm as the first. This results might be biased - because of the way the experiment user interface was designed the first algorithm was always the same. Since the rating of the algorithm was the lowest, we assume the users left the first algorithm picked when they weren't satisfied with the recommendation or were simply lazy.

Next in line are the Mixing Colors and the Energy Spreading algorithms. The first had a slightly better position but the second had a better rating. These algorithms appear to be the best of all, although they require some improvement. The Modified Dijkstra's rating is not bad, but it was not picked as good nearly as often as other algorithms.

The experiment we performed also had an auxiliary goal which was to find out information about the performance times of the algorithms. The only algorithm which had performance issues was the Modified Dijkstra's algorithm, which needs to keep a priority queue in order to work. The other algorithms' performance was satisfactory considering the scale of the data model graph.

4. Discussion and conclusions

We designed a graph-based data model and proposed four recommendation algorithms. The evaluation shows promising results, but it also shows the need to further research. Observing four different algorithms helped to achieve an improvement especially in the precision of recommendation. The experiment showed that we can disregard the Modified Dijkstra's algorithm and probably also the Union Colors algorithm and focus on improving the rest. The data model can also be improved by instructing new types of relationships, for example keywords. The main advantage of the graph representation for the task of item recommendation is performance. The implemented methods seem to be fast enough to work in real time. Our proposal is relevant for recommendation of huge number items and variable interests of users. Proposed traversal algorithms could be used in domains where we need to represent entities with variety of attributes and relations.

Acknowledgement: This work was partially supported by the the Scientific Grant Agency of Slovak Republic, grant VG1/0675/11 and by the Slovak Research and Development Agency under the contract No. APVV-0208-10. Presentation of this contribution was partially supported by Softec Pro Society.

References

to other papers publishing the results that are summarized here

[1] Demovic, Lubos; Fritscher, Eduard; Kriz, Jakub; Kuzmik, Ondrej; Proksa, Ondrej; Vandlikova, Diana; Zelenik, Dusan; Bielikova, Maria, "Movie Recommendation Based on Graph Traversal Algorithms," Database and Expert Systems Applications (DEXA), 2013 24th International Workshop on , vol., no., pp.152,156.

Other references

[2] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in Recommender Systems Handbook. Springer US, 2011, pp. 1–35.

[3] Z. Huang, W. Chung, T.-H. Ong, and H. Chen, "A graphbased recommender system for digital library," in Proc. of the 2nd ACM/IEEE-CS joint conf. on Digital libraries, ser. JCDL '02. NY, USA: ACM, 2002, pp. 65–73.