E-Commerce Recommender System Using Graph-Based Collaborative Filtering

Mahanti Indah Rahajeng 13517085
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517085@std.stei.itb.ac.id

Abstract— **Recommender system is growing so fast in the recent years. There are various methods to build a recommender system. With big amount of data, it is common to face problem such us slow processing data. According to graph attribute and graph utility, a graph-based data may be an alternative solution to manage the big amount of data. In this paper, author will discuss about e-commerce recommender system that using graph-based data and the very common algorithm for recommender system that is collaborative filtering.**

Keywords—**collaborative filtering, e-commerce, graph, recommender systems.**

I. Introduction

Recommender system has become a crucial and important thing that very often used for e-commerce. Even big companies such as Netflix, Amazon, and Facebook all develop their own recommender system. Recommender system is responsible to give appropriate recommendation or suggestion product with user personalized and user interests. Modern companies design recommender system to statistical significance, boost sales, increase company income, and holding user attention to company products [1]. Recommender system can be implemented in various products for example in movie, book, music retail, etc.

Nowadays, there are various algorithms and approaches to build a recommender system. Simply, recommender systems can be classified into four categories based on the approach to make recommendation: content-based, collaborative filtering, knowledge based, and hybrid approaches [2]. In this paper, I will discuss the most common one that is collaborative filtering. Collaborative filtering will predict the appropriate product based on the preference from the other user with the similar personalized and interests. It also has two different approaches, based on item similarity or user similarity, or we can combine both.

Recommender system will certainly be used to managing a huge amount of data with differences correlation. With this condition, we usually face the overload data problem that make slow processing and it is a common problem for this Big Data era. A graph-based data can be the alternative solution of the overload data problem. Managing data with graph can be easier because of its high degree of scalability and flexibility. The node is representing the entities such as users and products. The edge that connect two nodes is representing the relation between the two nodes, whether user to product, user to user, or product to product. The connectedness between entities is the core of the recommender system design [3].

According to all considerations mentioned earlier, I would like to discuss about e-commerce recommender system using graph-based data and the collaborative filtering algorithm with user similarity approach in this paper.

II. Theoretical Framework

A. Graph

Graph is denoted in $G = (V, E)$. V is a nonempty set of vertices/nodes and E is a set of edges/arcs that connect a pair of nodes. A graph can be drawn as a set of nodes in the two-dimensional plane that connected by a set of edges. Graphs can be classified into several categories depend on the approach. Based on directional orientation, there are three types of graph:

1. Undirected graph
   All the edges of undirected graph has no specific directional orientation.
2. Directed graph
   All the edges of directed graph has one specific directional orientation. A directed graph allows loop edge.
3. Bidirectional graph
   All the edges of bidirectional graph can have opposite directional orientation or bidirectional orientation [4].

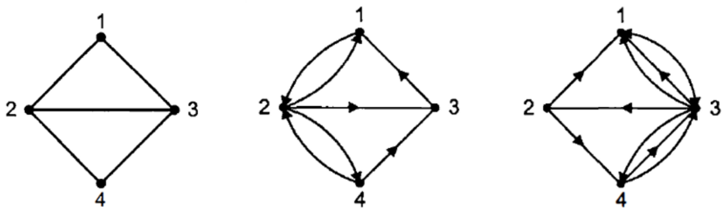

Figure 1 An undirected graph, a directed graph, and a bidirectional graph
Source: [4]

Graph has some basic terminology:
1. Adjacent
   Two vertices are adjacent if there is an edge that connect them.

2. Degree
   The number of edges that connect to a vertex. In directed graph, there is out-degree that is the number of outgoing edges and in-degree that is the number of incoming edges.
3. Path
   Sequence of alternating vertex and edges that each successive vertex is connected by the edge.
4. Connected
   Two vertices are connected if there is a path from one vertices to another one. Graph is connected if it has all pairs of vertices connected by at least one path.
5. Subgraph
   Subset of vertices and edges is called subgraph.
6. Weighted Graph
   A graph that all edges has a numerical value or weight.
7. Bipartite Graph
   A graph that the vertices can be divide into two disjoint subsets of edges. The vertex of one subset always connect to the vertex of the another subset [4].

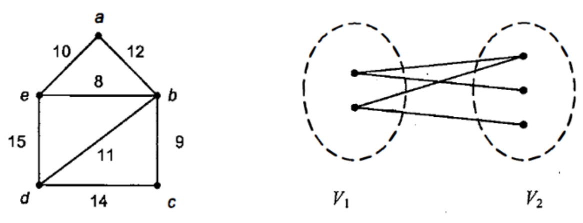

Figure 2 A weighted graph and a bipartite graph
Source: [4]

B. E-Commerce Applications

Recommender system benefits both users as well as sellers in e-commerce applications. It minimizes the amount of information that users need to process before they find the artifact they are looking for. For the seller, recommender system benefits in:

1. Improving cross-sell
   Cross-selling implies offering additional related products to a user when he shows interest in purchasing a product.
2. Improving up-sell
   Up-selling means recommending a similar but more expensive product package to the user.
3. Turning browsers into buyers
   Recommender systems help e-commerce systems to hold the attention of new and occasional users.
4. Providing various targeted products
   Recommender systems can offer personal service based on the understanding of the preferences of each customer.
5. Providing personalized advertisements
   Related to point 4, the recommender systems offer the kinds of advertisement that a user would welcome and interest.
6. Enhancing loyalty of customers
   Personalization leads to loyalty. If an user uses an e-commerce site that remembers enough contextual information about him, he is more likely to revisit the site in the future.

C. Recommender System

In common, recommender system can be divide into two main categories: content-based and collaborative filtering. But, as I mentioned before we can expand it into four categories based on the approach used to make recommendation:

1. Content-based
   User will get recommendation items similar to his preferences in the past.
2. Collaborative filtering
   User will get recommendation items that another user with similar personalized and preferences like in the past.
3. Knowledge based
   This use a knowledge component model and recommend user what he needs.
4. Hybrid approaches
   This approach combine various methods to give user recommendations [2].

All of the approaches has their own advantages and disadvantages. It depends on which side we want to approach and characterizing the recommender system that we build.

D. Collaborative Filtering

Collaborative filtering today forms the core underlying technology for many recommender systems. It has two different approaches that can be used to build the system. The first is using user similarity. The second one is item similarity. Actually, we can choose one of the approaches or even combine it to get a better recommendation result. In this paper, I decide to use user similarity because it has proven to give a better accuracy of recommendation compared to item similarity [5].

Using this method, the system needs to build users profile based on their behaviors at first. Collaborative filtering requires time to build user profile and provide preference data that can be analyzed. After the user profile built, the system compares it to different user profile that has similar preference to provide a recommendation. We can say that the users collaborate to build global profile and preferences. The participation of the user is very important to give the recommendation accurately [3]. By using this approach is the same as we make assumption that users having similar preferences in the past will continue to have same preferences in the future.

As a noted, there are various problems with collaborative filtering

- Cold start
  Collaborative filtering also gives poor performance for new items, items that have not been rated by users. As the number of users gets bigger, the number of comparisons over all combinations affects the scalability as well.
- Banana Problem
  Recommender system is sensitive to the frequency of ratings for specific items. For example, since bananas are frequently purchased by users in many stores a recommender system using market sales data to infer preferences will always recommend bananas. Similarly, there might be a user with unusual preferences and have no similar users and which could lead to poor recommendations.
- Subjectivity in Ratings

Ratings of similar users are combined to make recommendations in collaborative filtering, but users often use different ranges of ratings to express personalized preferences [7].

E. Jumping Connections

Mr. Batul J. Mirza studied an algorithm named *Jumping Connections*. Jumping connection is different with the other recommender system algorithm. It is not in terms of predicted ratings of items but in terms of the combinations of users and items that they bring together. The nature of connection jumped also helps in explaining the recommendations.

Mr. Batul J. Mirza intuitively thought that the two nodes described in a given pair can be reached from one another, by a single jump. Simply, we can assume that jumps can be composed in the following sense: if node B can be reached from A in one jump, and C can be reached from B in one jump, then C is reachable from A in two jumps. Then, the simplest jump is the skip jump, which connects two members in user set if they have at least one item in common [7].

III. BUILDING THE RECOMMENDER SYSTEM

A. Modelling Graph of User and Item

Graph of user and item represents the correlation between users and the items they bought. According to this correlation, I think that the appropriate type of graph to use is bipartite and directed graph that has direction from user node to item node. For example, there are data of users and item they bought

Table 1 Example data of users and item they bought

Users	**Item**
U1	I1, I2, I3
U2	I1, I2
U3	I2, I4
U4	I3, I4
U5	I4

A sample graph of the data above is shown in Figure 3.

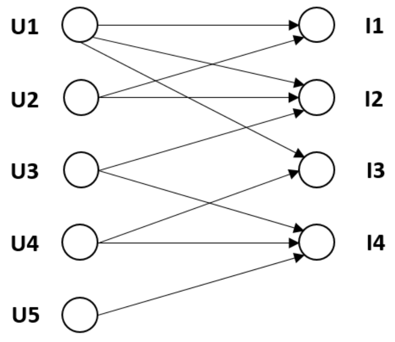

Figure 3 Graph of User and Item

From graph (G) above, we can write

$V(G) = \{ I1, I2, I3, I4, U1, U2, U3, U4, U5 \}$

$E(G) = \{ (U1, I1), (U1, I2), (U1, I3), (U2, I1), (U2, I2), (U3, I2), (U3, I4), (U4, I3), (U4, I4), (U5, 15) \}$

V is set of vertices and E is set of edges

B. Modelling Graph of User and User

Graph of user and user represents the similarity between user and the others. According to this correlation, I think that the appropriate type of graph to use is undirected and weighted graph that the weight value represents similarity of two users.

There are some methods to measure similarity between users, such as Common Neighbor, Jaccard's Coefficient, and Adamic-Adar. However, I decide to use common neighbor measurement because it is the simplest one. This measurement calculates the number of overlapping neighbor that two nodes have as the similarity score of those nodes [4]. Each measurement is denoted in B(x). If x is a user node, then B(x) is set contains all items that user x bought. Common neighbors similarity between node a and node b can be written as

$$CN(x, y) = |B(x) \cap B(y)| \qquad (1)$$

With the equation (1), we can get the similarity value as follow

Table 2 User Similarity Value

x,y	**CN(x,y)**
U1,U2	2
U1,U3	1
U1,U4	1
U1,U5	0
U2,U3	1
U2,U4	0
U2,U5	0
U3,U4	1
U3,U5	1
U4,U5	1

Similarity value show that more similar the two users preference, more its similarity value number. However, the graph weight value is the opposite of similarity value. Less weight value, closer those two nodes. Closer two nodes, more similar those two nodes. Each weight value of x and y user node is denoted in W(x,y). So, we can calculate the weight value as

$$W(x, y) = \frac{1}{CN(x, y)} \qquad (2)$$

There is an exception with equation (2). If CN(x,y) has value 0, W(x,y) also has value 0 which means that two nodes are not connected.

With the equation (2) and its exception, we can get the graph weight value as follow

Table 3 Graph Weight Value

x,y	CN(x,y)
U1,U2	0.5
U1,U3	1
U1,U4	1
U1,U5	0
U2,U3	1
U2,U4	0
U2,U5	0
U3,U4	1
U3,U5	1
U4,U5	1

A sample graph of the data above is shown in Figure 4.

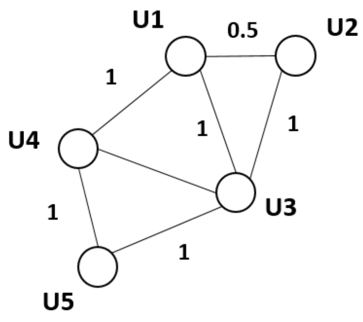

Figure 4 Graph of User and User

From graph (G) above, we can write

$$V(G) = \{\ U1, U2, U3, U4, U5\ \}$$
$$E(G) = \{\ (U1,U2), (U1,U3), (U1,U4), (U2,U1), (U2,U3), (U3,U1), (U3,U2), (U3,U4), (U4,U1), (U4,U3), (U4,U5), (U5,U3), (U5,U4)\ \}$$

V is set of vertices and E is set of edges

C. Modelling Recommender Graph

Simply, we can say that recommender graph is the combination between graph of user-item and graph of user-user. We can just modelling the recommender graph by rendering the graph of user-user with bidirectional edges and reattaching the item or using jumping connections construction.

Recommender graph represents the global correlation between user-item and user-user. Because of that, I think that the appropriate type of graph to use is bidirectional and weighted graph. The bidirectional edges between two user nodes represents that those are similar and the weight value represents similarity of two users. The directed edge from user to item represents that the user bought the item and its correlation has no weight value because it is all the same.

The recommender graph from jumping connections construction is shown in Figure 5.

The recommender graph obtained by rendering the graph of user-user with bidirectional edges and reattaching the items is shown in Figure 6.

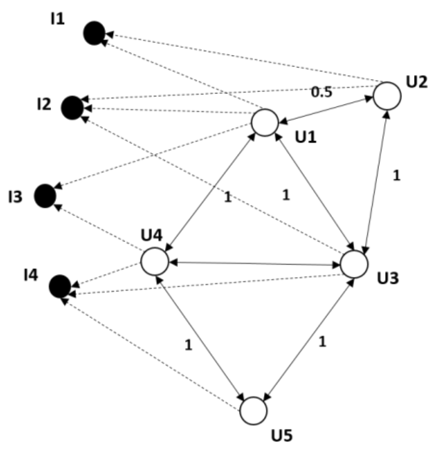

Figure 5 The Jumping Connections Construction

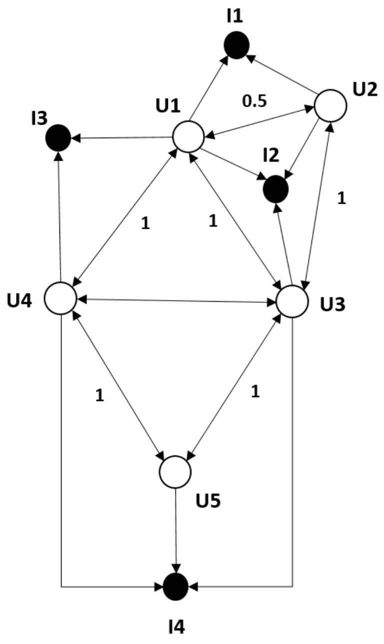

Figure 6 Recommender Graph

D. How Recommender Works

How the recommender works is as simple as search the all path from each user node to each item node. Because the edge between user node to item node has no weight and the value is all the same, we can assume and initialize those edge with weight value 1. Then, the recommender algorithm is as follow

1. Check if I1 is not an element from L(1).
2. If true, do the next step,
   If not true, the total weight is 0.
3. If true, find the shortest path (using whatever algorithm) then save the total weight value.
   If not true, the total weight value is 0.
4. Repeat until all path between U1 and all items is checked.
5. Repeat the step 1-4 for all user node.

From step 1-5, we can get the path length data as follow

Table 4 Path Length Data

User	I1	I2	I3	I4
U1	0	0	0	2
U2	0	0	1.5	2
U3	2	0	2	0
U4	2	2	0	0
U5	3	2	2	0

6. Rank the value for each user from the shortest to the longest path.

Using that algorithm, we can get the data as follow

Table 5 Recommendation Table

User	Recommendation		
	1	2	3
U1	I4	-	-
U2	I3	I4	-
U3	I1	I3	-
U4	I1	I2	-
U5	I2	I3	I1

The table show the most accurate recommendation to the less accurate from 1 to 3. As example, if we want to give U2 items recommendation, we should place I3 to the first recommendation and I4 to the second.

IV. Comparison Between Relational Database and Graph Database

As I mentioned before, a graph-based data could be a solution of big data problem. Graph database has a better performance compared to the common relational database. It has proven by the experiment Mr. Joseph George Davis and his associate did. They analyze the big and connected data with graph and relational database using collaborative filtering technique. The experiment result of execution time is shown in Table 6 and the experiment result of storage size is shown in Table 7.

Table 6 The Execution Time Comparison
Source: [6]

Data (record)	Execution time (sec)	
	Relational database	Graph database
100	0.0028	0.002
1000	0.016	0.006
10000	0.164	0.090
100000	1.315	0.219
500000	6.244	1.616
1000000	12.896	1.728

From the query complexity analysis it is seen that when it comes to analyzing connected data the relational database often uses joins to associate entities. This presents a complexity of O(N^2) because join queries are executed programmatically in nested loops whiles graphs queries have a complexity of $O(\log N)$ due to the traversal of the graph data structure.

Table 7 The Storage Size Comparison
Source: [6]

Data (record)	Storage Size (kbs)	
	Relational database	Graph database
100	792	400
1000	840	2000
10000	2744	18200
100000	14024	105100
500000	37596	535000
1000000	59096	839000

From those result, they concluded that when it comes to dealing with big and connected data the graph database has an urge over the relational database.

V. Conclusion

Recommender system is very useful to implement in e-commerce. However, there is a one big problem to solve according to a huge amount of data. We need to connect it and analyze it to build a good recommender system. Graph-based data comes with solution. Graph database has proven better than relational database to process big data. While using graph database, complexity is lower than using relational database.

There are so many different algorithms and approaches to build the recommender system follow by their own advantages and disadvantages. In this paper, I use three graph-based data: graph of user-item represents correlation between user and item he bought, graph of user-user represents similarity between two user, and recommender graph to produce recommendation items to particular user. Before modelling graph of user-user, I measure the similarity between each user using common neighbor measurement. To model the recommender graph, I just simply rendering the graph user-item and graph user-user and I also tried to use jumping connections algorithm. Because of that, the recommender graph is more like random graph and not following the regular order. The algorithm I used to make a recommendation is just that ordinary, not the efficient one. This paper just need a lot of improvement.

VII. Acknowledgment

I would like to express my gratitude to God Almighty for the opportunity and guidance so I can finish writing this paper. Secondly, I would also be thankful to Mr. Rinaldi Munir as my lecturer in Matematika Diskrit in this semester for enthusiasm and knowledge he shared to us. Last, I would say thank you to my parents and everyone who supported me to finish this paper.

I realize this paper is far away from being perfect. I probably did mistakes while writing this paper. So, constructive criticism and suggestion is all I need. Hopefully, this paper would be useful for all reading it.

References

[1] V. Venkatraman, S. Vijay, A. Jain, K. Vedantham, “Recommender Systems using Graph Theory” vol. 5 no. 08 August 2013, *International Journal of Engineering Science and Technology (IJEST).* ISSN : 0975-5462.

[2] S. S. Lakshmi, Dr. T. A. Lakshmi, “The Survey of Recommender Systems”, *International Journal of Engineering Science and Technology (IJEST)-Spesial Issue-April 2017*. ISSN: 2231-5381.

[3] H. Cung, M. Jedidi, “Implementing a Recommender system with graph database (Seminar)”, Universite de Fribourg, 2014.

[4] R. Munir, *Matematika Diskrit, Edisi 3*, Bandung: Penerbit Informatika, 2010, ch 8.

[5] A. A. Putra, R. Mahendra, I. Budi, Q. Munajat, “Two-Steps Graph-Based Collaborative Filtering Using User and Item Similarities: Case Study of e-Commerce Recommender System”, *International Conference on Data and Software Engineering (ICoDSE)*, 2017.

[6] J. G. Davis, J. K. Panford, J. B. Hayfron-Acquah, “Big and Connected Data Analysis with Graph and Relational Databases Using Collaborative Filtering Technique” vol. 15 no. 12 December 2017, *International Journal of Computer Science and Information Security (IJCSIS)*.

[7] B. J. Mirza, “Jumping Connections: A Graph-Theoretic Model for Recommender System (Thesis)”, Virginia Polytechnic Institute and State Univeristy, 2001.

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Desember 2018

[illegible]

Mahanti Indah Rahajeng
13517085