A Recommendation Algorithm for Collaborative Conceptual Modeling Based on Co-occurrence Graph

Kai Fu[1,2], Shijun Wang[1,2], Haiyan Zhao[1,2(✉)], and Wei Zhang[1,2]

[1] Institute of Software, School of EECS, Peking University, Beijing, China
{fkai1993,wangshijun,zhhy.sei,zhangw.sei}@pku.edu.cn
[2] Key Laboratory of High Confidence Software Technology,
Ministry of Education of China, Beijing, China

Abstract. Conceptual models are models used to describe objects or systems in the real world. The quality of a conceptual model heavily depends on the domain knowledge and modeling experience of the individual modeler. Collaborative conceptual modeling is an effective way of building models by taking advantage of collective intelligence. This paper proposes a Co-occurrence Graph based Recommendation Algorithm (CGRA) to implement the collaborative mechanism of conceptual modeling systems. CGRA, inspired by association rule mining algorithm, is an incremental data updating algorithm. The computational complexity of CGRA is much lower than that of the traditional association rule mining based algorithms, while the recommendation effectiveness of these two are almost the same in our collaborative conceptual modeling system, which is revealed by the experiments we have conducted.

Keywords: Collaborative Conceptual Modeling · Recommendation · Association Rule Mining · Co-occurrence Graph

1 Introduction

Conceptual models are models used to describe objects or systems in the real world. For most areas of engineering, to build the corresponding conceptual model on the stage of requirement analysis is of great importance. However, the quality of a conceptual model heavily depends on the knowledge and experience of the individuals who build it. It is scarecely feasible to build a good domain specific conceptual model all by one person, due to the limited kowledge of an individual. Traditionally, conceptual modeling requires a dozen of experts who are familiar with the corresponding domain knowledge to gather together and reach a consensus. Top-level experts are not always available for every organization, and the whole procedure of experts' meeting is time consuming.

Fortunately, in the Internet age, we could utilize collective intelligence to compensate for the lack of individual knowledge and top-level expert modelers.

K. Fu and S. Wang—These authors contributed equally to this work and should be considered co-first authors.

L. Liu and M. Aoyama (Eds.): APRES 2015, CCIS 558, pp. 51–63, 2015.
DOI: 10.1007/978-3-662-48634-4_4

Collaborative conceptual modeling system provides a way to make use of collective intelligence [7]. In a collaborative conceptual modeling system, people could build their own conceptual models while enjoying the benefit of collective intelligence from recommendation information pushed by the modeling system. Individuals use their conceptual modeling domain knowledge independently, while collaborative conceptual model will recommend some concept elements with high quality in collective conceptual model to the individual in the process of modeling, to help individuals to explore and establish a better conceptual model. As the individual conceptual models getting better, the quality of the collaborative conceptual model is promoted in return, which forms a positive feedback loop.

Recommendation plays a key role in such collaborative conceptual modeling systems. During the modelling process, the communication among individuals depends mainly on the information from the recommender system as an indirect interaction. So the results of the collective conceptual model for the individuals depends on the recommender system. If the recommendation results do inspire the individuals, a positive feedback loop is established, which further enables the collective conceptual model evolve eventually to a conensual conceptual model by the majority of individuals.

Traditional recommendation approaches include collaborative filtering (CF) based approaches, content based approaches and knowledge based approaches [2]. These approaches are more applicable to the scenarios like online shopping item recommendation, music recommendation and movie recommendation, in which, recommender systems tend to recommend some items that have similar features to those that user has purchased before according to the user's need. However, due to the characteristics of conceptual modeling, most of those approaches are difficult to recommend a good concept element to the individual except for the association rule mining based CF approaches [1][6], since each concept in the individual conceptual model is unique, any concept that the individual currently does not possess would not be similar to those already in the individual conceptual model.

On account of the characteristic of association rule mining based CF approaches, they could be well applied in collaborative conceptual modeling systems. However, there still are some shortcomings to use these approaches in our conceptual modeling system. On one hand, they tend to get too many candidate item sets and therefore lead to high I/O overhead. On the other hand, the computational complexity of those approaches is rather high. In our collaborative conceptual modeling system, the running time of the recommendation algorithm must be taken into account to guarantee real-time reponses to the individual operations.

In this paper, we propose a Co-occurrence Graph based Recommendation Algorithm (CGRA), a simplfied association rule mining based algorithm, to solve the problems mentioned above. The computational complexity of CGRA is much lower than that of the traditional association rule mining based algorithms, while the recommendation effectiveness of these two are almost the same in our collaborative conceptual modeling system.

The term co-occurrence graph and co-occurrence matrix have been widely used in the literature of recommender systems and text indexing [4], usually denoting the relationships between items and users, or those between documents and words. On the contrary, co-occurrence graph in our work has a different meaning: The two vertices of a edge are both conceptual elements.

The remainder of this paper is organized as follows. Section 2 introduces some preliminaries aboutcollective conceptual model and association rule mining. Section 3 presents the basics about our CGRA, including the definition of Co-occurrence Graph (CG, the core data structure we use in our algorithm), the CG building method, our recommendation stragegies and finally a incremental CG updating algorithm. Section 4 discusses the advantages of CGRA over traditional association rule mining based algorithms in a collaborative conceptual modeling system, both theoretically and experimentally. Finally, section 5 concludes our work.

2 Preliminaries

CGRA is a modified version of association rule mining which is based CF approaches, to make the traditional association rule mining recommendation appliable to the scenarios during collaboritive conceptual modeling. Co-occurrence graph is uesd to describe and store a certain kind of association rules. The co-occurrence graph in CGRA is constructed from collective conceptual model, a fusion version of each individual's model. In this section, we first focus on the collective conceptual model, and then review the traditional algorithms and stratergies of association rule mining approaches used in recommender system.

2.1 Collective Conceptual Model

A conceptual model could be represented as a class diagram, with concepts as classes with attributes. Under collaborative scenarios, the collective conceptual models should be capable of reflecting the viewpoints of a group. However, the traditional class diagram representation is hard to describe the characteristic of group. We therefor have to make some changes for the collective conceptual model. For instance, the traditional class diagram model is a hierarchical structure, the attributes are part of the class, and the classes rely on relationship to connect each other. It's difficult to depict the collective class diagram model, owning to the relationship between attributes and classes. In this paper, the hierarchical structure of collective class diagram model is converted into a model of collective class diagram based on graph structure. In this graph, there are three primary types of nodes: concept nodes, relationship nodes and value nodes. Each node attaches a counter recording the corresponding number of users who reference it. Each concept node represents one class. Each relationship node represents the relationship of two connected concept nodes. Each value node represents a value. And each edge has a name. Figure 1 shows a structrual graph representaion (right part) and its corresponding conceptual model(left part).

In addition, we refer to the behavior user create an element or reference the element created by other user as a reference.

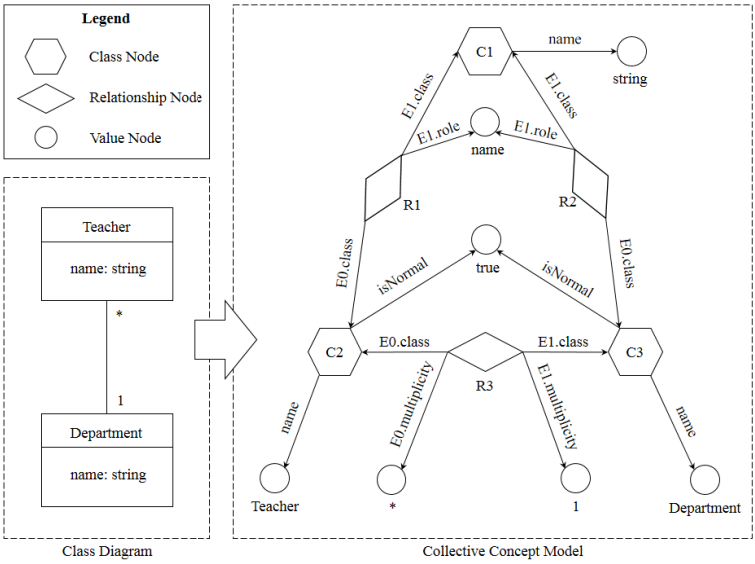

Fig. 1. The conversion of a conceptual model from class diagram representation to its structrual graph representaion

2.2 Association Rule Mining Based CF Approaches

Association rule mining are used to discover interesting relationships hidden in large data sets [1]. To better formulate the problem, we first give some definitions.

$I = \{i_1, i_2, \dots, i_d\}$ denotes the set of all items in a market basket data. $T = \{t_1, t_2, \dots, t_N\}$ denotes the set of all transactions. An itemset is a collection of zero or more items, i.e. a subset of I. K-itemset means an itemset contains k items. A transaction contains a subset of items chosen from I. The support count of $\sigma(X) = \left|\{t_i | X \subseteq t_i,\ t_i \in T\}\right|$.

Association rule is represented as $X \to Y$, where X and Y are disjoint itemsets, i.e. $X \cap Y = \emptyset$.

There are two primary metrics to measure the quality of a given association rule $X \to Y$.

One is the support $s(X \to Y) = \frac{\sigma(X \cup Y)}{N}$, which measures the degree of the correlation between itemsets.

The other is the confidence $(X \to Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$, which measures the significance of the correlation between itemsets.

Using these definitions, the association rule mining problem could be formulated as follows: *Given a set of transactions T, find all the rules having support* $\geq$ $minsup$ *and confidence* $\geq$ $minconf$, *where* $minsup$ *and* $minconf$ *are the corresponding support and confidence thresholds.*

The basic way to solve this problem is to divide it into two major steps, frequent itemset generation and rule generation.

Frequent itemset generation would find all the itemsets that satisfy the $minsup$ threshold (frequent itemsets). And during rule generation step, it extracts all the high-confidence rules from the frequent itemsets found in the previous step (strong rules).

To lower the high computational complexity of the solution, the association rule mining problem in traditional recommender system [2] is a little bit different:

Given a set of transactions T, find all the k-itemset to 1-itemset rules having $|\{rule|1\text{-}itemset_i \text{ as the head of rule}\}| \in [minNum, maxNum]$, *where* $i=1,2,\ldots,d$, *and confidence* $\geq minconf$, *where* $minconf$ *is the corresponding confidence thresholds.*

Note that the support threshold is not set in advance, which prevent the situation that too many or too few rules are generated for a certain item when the traditional association rule mining is applied.

3 The Co-occurrence Graph Based Recommendation Algorithm

The recommendation scenarios in collaboritive conceptual modeling are sort of different from those in the traditional recommender systems. In traditional recommendations, the quality of a recommendation result is measured by whether the recommended items are adopted by the user. Whereas, the recommendation in a conceptual modeling focuses on whether it could inspire the individual modeler to build a better conceptual model at last. Therefore, the the precis ion of one single recommendation is not of that great significance. On the other hand, the recommendation requests would be sent in each step during one individual's modeling activity, requiring an even stricter time constraint compared to traditional recommender systems.

To reach a solution, we further improved the association rule mining based recommendation algorithm. It has much lower computational complexity while maintaining the recommendation quality required by collaboritive conceptual modeling systems. What's more, we promote an incremental recommendation algorithm that could be applied to a real collaboritive conceptual modeling system. In the end, we analysis the effect and advantage of our algorithm compared with the traditional association rules.

3.1 Co-occurrence Graph

To lower the complextiy of association rule mining algorithm, we only focus on the 1-itemset to 1-itemset rules. Under such circumstances, the rules could be represented as a graph, whose vertices are concepts or relationships in the modeling system and edges are the association rules. Such a graph is called a co-occurrence graph, suggesting that the edges are co-occurrence rules of nodes in the graph.

Thus, our problem could be formulated as follows: *Given a set of transactions T, calculate the confidence of all the 1-itemset to 1-itemset rules.*

The transactions are mapped to individual users, and the items to concepts and relationhips between concepts.

In the rest of this paper, we would adopt the following definitions to describe a co-occurrenc graph as is shown in Figure 2. Element node $e_i, where\ i = 1,2,\ldots,n$

denotes a concept node or relationship node in the co-ocurrence graph. Element set $E_{set} = \{e_1, e_2, \ldots, e_n\}$ denotes a collection of element nodes. User $u_i, where\ i = 1,2, \ldots, n$ denotes an individual modeler in the collaborative conceptual modeling system. User set $U_i = \{u_1, u_2, \ldots, u_n\}$ denotes a collection of users that reference a certain element node.

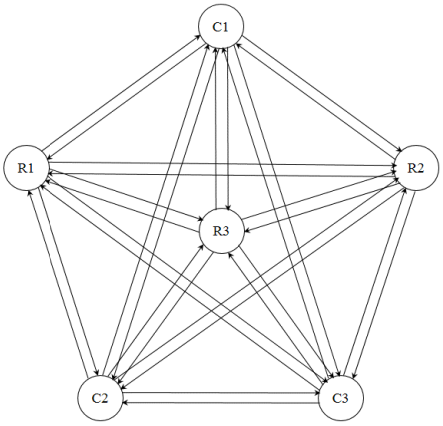

Fig. 2. A co-occurrence graph

3.2 The CG Building Method

For each element e_i, it has a set U_i containing all the users who reference the node. For all the nodes j except the node i in E_{set}, if the user has referenced the element node i, then the probability that he will reference the node j is $p(j|i) = \frac{|U_i \cap U_j|}{|U_i|}$, in which the element node j belongs to the set E_{set} and $i \neq j$.

Then our algorithm could be described as follows.

1) Traverse an element node from the node set E_{set}. If the traverse ends, go to step 7; else select the element node i.
2) For element node i, get the reference user set U_i.
3) Traverse an element node from the node set E_{set} except element node i. If the traverse ends, go to step 1; else select element node j, go to step 4.
4) For element node j, get the reference user set U_j.
5) Build an edge from node i to node j, and set the edge value $P(j|i) = \frac{|U_i \cap U_j|}{|U_i|}$.
6) go to the step 3.
7) End.

3.3 Recommendation Strategies

There are three main recommendation strategies. They respectively corresponds to three different recommendation scenarios. We will first introduce the three recommendation scenarios before we talk about the corresponding recommendation strategies.

The first recommendation scenario is for new users who do not reference any model element, in which case we could not get any information from the new users' behaviors. The second recommendation scenario is that the user has created some elements in his conceptual model, where the user need some recommended elements to promote his understand of the collective conceptual model. The third scenario is the most common scenario in our recommender system: users select one of the elements that he has already referenced.

Strategy One: Directly recommend some elements that have high reference count in our collective model for new users

Strategy Two: Multiply the correlation intensity between the elements user referenced and other elements by support to get a result. Sort these results and get recommendation order of the elements that are not referenced by user.

Strategy Three: Suppose that the user U_1 references the element A. And we get the first m elements which have the largest P value frsom the nodes that are not referenced by U_1 in the co-occurrence graph. Then we sort these nodes by P value and put the first m element nodes in our collective conceptual model. Finally we can get the corresponding concepts of these elements and recommend these concepts to the user.

The process of choosing strategies above is based on the co-occurrence graph. Although the co-occurrence graph has the most significant effect on strategy 3, all the three strategies are based on the related properties of the co-occurrence graph in order to get the results of recommendation as soon as possible. Therefore, the co-occurrence graph has a great influence on the results of recommendation.

3.4 Incremental CG Updating Algorithm

To further reduce the time consumption of our current algorithm, an incremental updating algorithm could be introduced. With the help of this incremental approach, our algorithm could generate a new co-occurrence graph faster when the collective concept model is changed, resulting in more accurate concepts being recommended to users.

Figure 3 is a co-occurrence graph made of three nodes, A, B and C. The corresponding user sets of the three nodes are U_A, U_B, U_C. If the state that user u_i has never referenced A changes to the state that the user u_i references A, the co-occurrence graph must be updated. At the same time, the state of node A in the graph is changed (The number of users that reference node A increases). And all the edges starting from A and all the edges ending in node A are influenced, which means the P value of these edges are changed.

If we updates all the influenced edges, 2n edges need to be updated. However, we notice that if the user u_i has referenced the node A, then we don't need to recommend A to u_i. Thus it has no influence to u_i whether or not updating the edge ending in node A, For u_i, only the edges that start from the node A have influence on the results of recommendation. In addition, if more than one user modify node A in a certain time, the modifying of edges that ending in node A during this time is meaningless. Thus, this recommendation this time should not include node A, no matter how to modify the edges.

To solve this problem, we take the method namely delay modifying. That means that after node A is modified by user u_i, we only need to update all the edges that start from A. And we mark node A that it needs to update all the edges that end at it. We will not update all the edges that end at node A until we need to recommend node A to those users who have never referenced node A. In this way, we need to update only n edges no matter how many users are modifying the node A, because the n edges that end at node A are updated until we need to recommend node A to a user that has never referenced node A.

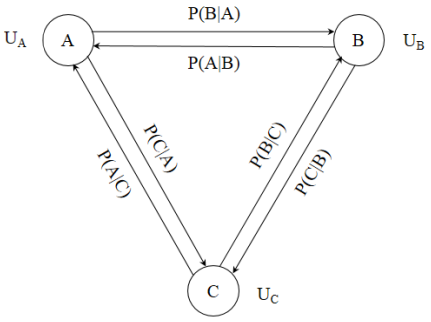

Fig. 3. Calculations in a co-ocurrence graph

The specific algorithm is shown as follows:

1) The user u_i request the recommender system to recommend concepts to him.
2) If the user u_i modify the node A, we will update all the edges that end in node A: $P(B|A) = \frac{|U_A \cap U_B|}{|U_A|}$, in which B belongs to E_{set}. And we value the Flag[A] true: Flag[A]=true. If not, go to step 3.
3) Go through the data records in the array Flag and determine if the nodes in array Flag except node A have been referenced by user u_i. If all of them have been referenced, go to step 7. If not, go to step 4.
4) Get the node X that is marked true and is not referenced by user u_i in array Flag and mark the node X false: Flag[X]=false.
5) Update all the edges that end in node X: $P(X|B) = \frac{|U_X \cap U_B|}{|U_B|}$, in which B belongs to E_{set}.
6) Go through the data records in the array Flag and determine if there exist a node in all the nodes in array Flag except node A that is marked true and has not been referenced by user u_i. If there exists such a node, go to step 4. If not, go to step 7.
7) Use the updated co-occurrence graph to fit the three scenarios in 3.3. And recommend the corresponding element nodes to the user u_i (Node A won't be recommended in this round). The final recommendation results are generated.
8) End.

4 Algorithm Analysis and Experimental Evaluations

In this section, we will first explain why CGRA in this paper is a more appropriate choice to be used in collaborative modeling system compared with the traditional association rule mining algorithms. Then we will prove the efficiency of our algorithm by comparing the experimental results of our algorithm with a traditional association rule mining based algorithm.

4.1 Algorithm Analysis

As is described in section 3.3, there are three recommendation scenarios in collective collaborative modeling system. For the first scenario, it is reasonable to recommend the most often referenced elements in the current collective model. For the third scenario, recommendations of each element need to be provided to the user. In this case, it's time consuming using traditional association rule algorithms, in which 1-to-1 association rules are produced at the same time with m-to-n rules. However, CGRA in this paper will produce 1-to-1 association rules quickly and recommend the model most relevant elements to the user. It reduces a lot of time compared to the traditional association rule algorithm. For the second scenario, our algorithm only uses 1-to-1 association rules not considering m-to-n association rules. In the following, we will prove that it is reasonable.

It is important to recommend items that could attract a user for online stores, movie and music websites for profit. So it is necessary and essential to recommend a user items most relevant to that those user has bought.

However, we aim to help a user build and refine his conceptual model in collaborative modeling. During the process, on one hand, we hope that the user can obtain some helpful elements from the recommender system in our collective conceptual model. On the other hand, the user should not completely depend on the recommend system to build a model. Instead, the user needs these recommended elements to expand thinking to build a more perfect model. In addition, it makes little difference when recommending an element since modeling is a continuous process. As a result, the recommender system using collaborative modeling in this paper doesn't always recommend the most relevant elements to the user in first time.

It doesn't mean that the recommender system recommends random elements to the user. We still recommend rather helpful elements to the user. The order of the recommended elements is not strict in our algorithm.

Although m-to-n association rules could produce related elements earlier compared to 1-to-1 association rules, it is time consuming for huge amount of data to compute the m-to-n association rules due to the exponential computational complexity of this problem [5]. However, due to the fact that a user depends on the recommender system to interact with collective conceptual model during the process of modeling, we hope that the recommender system is capable of adjusting itself quickly to the collective conceptual model.

Considering the fact mentioned above that the order of elements recommended doesn't matter to a collective collaborative modeling system and the m-to-n association rules

are really time consuming, we only consider the 1-to-1 association rule in our algorithm and discard m-to-n association rules to achieve a polynomial computational complexity. The effectiveness of our algorithm will be validated in the experimental evaluation parts of this section.

4.2 Experimental Evaluations

All of the experiments in this paper are based on data for collaborative conceptual modeling due to that the recommender system in this paper is specifically for collaborative conceptual modeling. The data in this paper come from a collective conceptual model named Course Management System created by 15 users, which contains 67 class nodes and 360 relation nodes. We compare the traditional association rule algorithm and our CGRA based on co-occurrence graph. We choose the efficient algorithm FP-Growth improved from the traditional association rule mining algorithm Apriori as the represent of association rule algorithm [3]. FP-Growth algorithm produce both 1-to-1 association rules and m-to-n association rules.

In the third scenario mentioned in 3.3 section, the parameter support and the parameter confidence are both set as zero in FP-Growth algorithm to compute the correlation intention of elements that have been used in model. It takes more than 12 hours to obtain the results. However, it takes only 3 seconds in our algorithm. In addition, the recommendation results of the two algorithms are the same since only 1-to-1 association rules are used in this scenario.

Thus, the results of the two algorithms in the second scenario are compared in the following.

In the Course Management System collective model, two users (user 1 and user 2) participate in the model. User 1 completely adopts recommended elements and does not create any new element by himself. User 2 chooses some elements from the recommended elements, think over and build his own model.

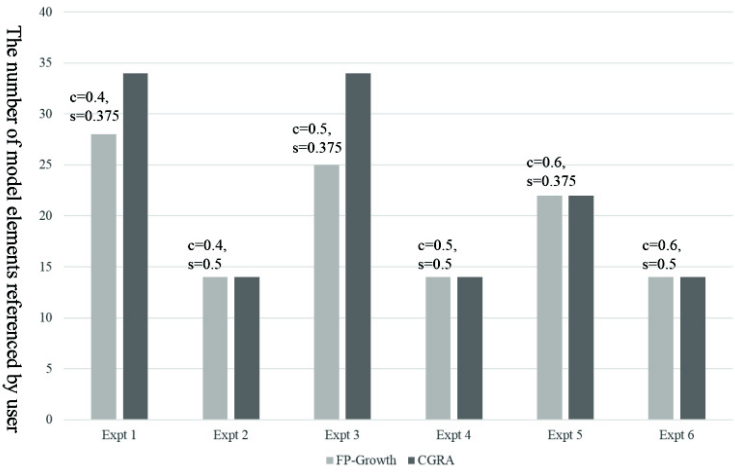

Fig. 4. Recommendation results of user 1 using the two algorithms respectively (*c* for *confidence*, *s* for *support*)

To make the recommendation results more convincing, the parameters support and confidence are set as different values in FP-Growth algorithm to get rid of some unconvincing association rules. We will compare a group of results of CGRA and FP-Growth with different values of support and confidence.

The specific process of the experiment is shown as follows.

In the initial state, the recommender system will recommend the top-four most referenced elements to a user due to the lack of individual modeling information. In the following time, the recommender system will recommend 6 elements at most every time until the user stop the process of modeling.

Figure 4 shows the recommendation results of user 1 using the two algorithms respectively. For FP-Growth algorithm, each bar represents the number of referenced elements at a certain value of confidence and support. For CGRA, each bar represents the number of referenced elements that contains all the elements recommend by FP-Growth. As what the figure 4 shows, our recommendation results cover those in FP-Growth, which means that our algorithm is more effective. In fact, the order of the recommended elements is almost the same except for a few elements. However, it does not influence the final model results.

Due to the small scale of this data set, there is no significant difference between our algorithm and FP-Growth algorithm in time consuming in this experiment. As we all know, it takes a long time for a large amount of data using FP-Growth algorithm. Besides, that our algorithm is faster could be inferred from the behavior of FP-Growth algorithm in scenario 3. The corresponding experimental results are shown in Table 1.

Table 1. Run time evalution

Algorithms	Average running time (s)
FP-Growth (c=0.6, s=0.375)	4
FP-Growth (c=0.6, s=0.5)	3
FP-Growth (c=0.5, s=0.375)	4
FP-Growth (c=0.5, s=0.5)	3
FP-Growth (c=0.4, s=0.375)	4
FP-Growth (c=0.4, s=0.5)	3
CGRA	3

What's more, we notice that when the confidence is set as 0.5, support is set as 0.375 or the confidence set as 0.4 and support set as 0.375, the recommendation result of our algorithm includes more elements than that of FP-Growth until all the elements is recommended by FP-Growth. To prove that our algorithm will recommend more and more helpful elements than FP-Growth algorithm in users' behaviors, we compare the results of our algorithm and the FP-Growth algorithm of user 2 with two groups of parameters. Figure 5 shows the results of the two algorithms of user 2.

In summary, the experiments above show that the recommendation effectiveness of our algorithm is not worse than that of the traditional association rule mining algorithm. However, due to the simple structure of our algorithm, based on co-occurrence graph, the algorithm in this paper is much faster than the traditional association rule mining algorithm. Of course, because the collaborative conceptual modeling tool is

still in its experimental stage, the amount of participants is not large. In the future work we will verify the experimental results with a larger number of participants and a more complex modeling environment.

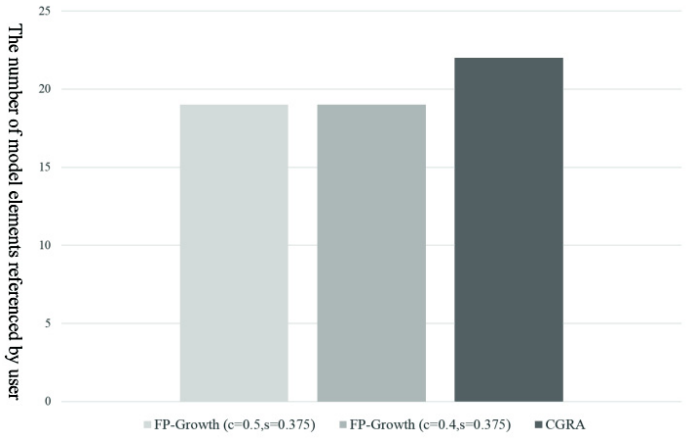

Fig. 5. The results of the two algorithms of user 2

5 Conclusions

The computational complexity of our co-occurrence graph based recommendation algorithm is much lower than that of the traitional association rule mining based algorithms, while the recommendation effectiveness of these two are almost the same in our collaborative conceptual modeling system.

Acknowledgements. This research was supported by the National Basic Re- search Program of China (the 973 Program) under grant 2015CB352201; Science Fund for Creative Research Groups of the National Natural Science Foundation of China (Grant No.61421091); and the National Natural Science Foundation of China under grants 91318301 and 61272163.

References

1. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. ACM SIGMOD Record, ACM. **22**(2), 207–216 (1993)
2. Bobadilla, J., Ortega, F., Hernando, A., et al.: Recommender systems survey. J. Knowledge-Based Systems. **46**, 109–132 (2013)
3. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. ACM SIGMOD Record, ACM. **29**(2), 1–12 (2000)

4. Huang, Z., Chung, W., Chen, H.: A graph model for E-commerce recommender systems. J. Journal of the American Society for Information Science and Technology. **55**(3), 259–274 (2004)
5. Kosters, W.A., Pijls, W., Popova, V.: Complexity analysis of depth first and fp-growth implementations of apriori. In: Perner, P., Rosenfeld, A. (eds.) MLDM 2003. LNCS, vol. 2734, pp. 284–292. Springer, Heidelberg (2003)
6. Lin, W., Alvarez, S.A., Ruiz, C.: Efficient adaptive-support association rule mining for recommender systems. J. Data Mining and Knowledge Discovery. **6**(1), 83–105 (2002)
7. Zhang, W., Zhao, H., Jiang, Y., et al.: Stigmergy-Based Construction of Internetware Artifacts. J. Software, IEEE. **32**(1), 58–66 (2015)