

1-1-2016

Incremental weighted bipartite algorithm for large-scale recommendation systems

HAIHONG E

JIANFENG WANG

MEINA SONG

QIANG BI

YINGYI LIU

Follow this and additional works at: <https://journals.tubitak.gov.tr/elektrik>



Part of the [Computer Engineering Commons](#), [Computer Sciences Commons](#), and the [Electrical and Computer Engineering Commons](#)

Recommended Citation

E, HAIHONG; WANG, JIANFENG; SONG, MEINA; BI, QIANG; and LIU, YINGYI (2016) "Incremental weighted bipartite algorithm for large-scale recommendation systems," *Turkish Journal of Electrical Engineering and Computer Sciences*: Vol. 24: No. 2, Article 8. <https://doi.org/10.3906/elk-1307-91>
Available at: <https://journals.tubitak.gov.tr/elektrik/vol24/iss2/8>

This Article is brought to you for free and open access by TÜBİTAK Academic Journals. It has been accepted for inclusion in Turkish Journal of Electrical Engineering and Computer Sciences by an authorized editor of TÜBİTAK Academic Journals. For more information, please contact academic.publications@tubitak.gov.tr.

Incremental weighted bipartite algorithm for large-scale recommendation systems

HaiHong E*, JianFeng WANG, MeiNa SONG, Qiang BI, YingYi LIU

Beijing University of Posts and Telecommunications, Beijing, P.R. China

Received: 13.07.2013

Accepted/Published Online: 11.12.2013

Final Version: 05.02.2016

Abstract: Personalized recommendation systems are a solution for information overload. In the real recommendation system, the data are dynamic; thus, the real time of the system will be seriously affected if we recalculate after every change, and, in time, sensitive situation of the delayed updated calculation will affect the accuracy of the recommendation. The incremental calculation methods of collaborative filtering, clustering, singular value decomposition, and bipartite algorithm have achieved some progress; however, there is limited research on the incremental computation of the weighted bipartite algorithm. This paper proposes a personalized recommendation algorithm. In this algorithm, the rank of the recommendations is effectively influenced by the rate and times of the same choice. This paper uses the borrowing records of students in college libraries to model the recommendation system and achieve a rank of personalized recommendations for college students. This system, combined with the autonomic learning resources platform, will improve student learning efficiency.

Key words: Recommendation system, incremental weighted bipartite algorithm, personalized point of learning interests

1. Introduction

Although data sparsity has become a severe problem in the implementation of internet recommendation systems, there are huge amounts of user and item data that need to be handled in time. How to calculate these data efficiently is a challenge. Furthermore, most research concentrates on handling static data and proposing new methods to optimize recommendation algorithms. Nevertheless, the data in real systems are always changing with time. These will become complex if we recalculate the entire dataset after each variation. It is obvious that this will affect the system's real-time service, especially in certain time-sensitive situations.

For personalized recommendation systems, the incremental calculation problem mainly occurs when new ratings or choices are added. It is necessary to find a way to calculate the similarity quickly or update the item recommendation order. Personalized recommendations can be realized in several ways, including the collaborative filtering algorithm, clustering algorithm, or singular value decomposition (SVD). These techniques can be used in different situations, which means that the incremental algorithm needs to consider a suitable algorithm. The final target for incremental calculation is that the calculation only be based on newly added information rather than on the whole dataset, when new information is accepted. This should be realized based on the premise of ensuring the recommendation's accuracy in order to improve computational efficiency and the system's response time.

Nowadays, research on the incremental calculation methods of the collaborative filtering algorithm [1,2], SVD [3–8], and the clustering method [9–14] has made progress. Section 2 summarizes these 3 recommendation mechanisms, incremental algorithm design, and algorithm optimization methods.

*Correspondence: ehaihong@bupt.edu.cn

Another kind of recommendation algorithm, which is based on network structure, introduces a new idea. Zhou et al. [15,16] considered users and items as nodes and utilized the selection between user and item to build a bipartite graph. Liu et al. [17] promoted weighted bipartite network recommendation. A new heat conduction algorithm [18] in social network recommendation methods was proposed. It showed that heating conduction has good performance compared to conventional similarity computation.

The addition of a new edge is classified into 4 types in the bipartite network incremental algorithm [17]. This provides a foundation for network-based recommendations; however, there is limited research on weighted bipartite network incremental algorithms. Therefore, this paper proposes an incremental weighted bipartite network-based personalized recommendation method, which involves user ratings or preferences, in order to advance performance and calculation speed.

Section 3 analyzes the incremental computation of bipartite networks in another way and infers a paradigm formula that can help different application systems in modeling according to different situations. This paper uses student borrowing records in university library modeling recommendation systems to show the implementation of the algorithm and acquire the recommendation order of student learning resources.

In Section 4, the paper proposes a new incremental weighted bipartite personalized recommendation algorithm and then gives 4 incremental equations and their derivative process. We conducted a simulation with experiment datasets that contained weights that were generated by student borrowing times and then tested the algorithm's validity.

In Section 5, we do experiments to assess the incremental algorithm by using 8000 books, 15,757 students, and 399,899 terms of borrowing records as datasets. Finally, we compare and analyze the new algorithm's recommendation precision, recall, and F1 measure to evaluate the bipartite recommendation algorithm.

2. Conventional recommendation algorithm and its incremental computation

2.1. Collaborative filtering algorithm

The collaborative filtering recommendation algorithm uses obtained data, which show user preference for items, to calculate similarities and find groups of users that have similar tastes for items. This group can help generate a recommendation collection for target users. It is obvious that we can calculate similarity in 2 ways: one is user-based and the other is item-based. The user-based method calculates similarities among users, whereas the item-based method calculates similarities among items. This mechanism does not need formal modeling and can be used for recommending items that are not easily analyzed by contents. Nevertheless, it has low accuracy for recommending new items to users, and the amount of calculation increases linearly with increasing users and items.

The vital question of incremental collaborative filtering recommendation is how to calculate similarity again in an efficient way and produce new neighbor users. The incremental algorithm of collaborative filtering recommendation [1] divides the similarity equation into 2 parts: a constant part and a changeable part. It is necessary to change the results by using changeable parts when obtaining new ratings, in order to decrease the time consumption of searching the entire datasets. In [2], the similarity calculation adopts a cosine similarity equation instead of the Pearson correlation coefficient, which is more appropriate for binary rating recommendation algorithms.

2.2. Singular value decomposition

SVD means that any matrix can be decomposed in the following format by employing SVD:

$$A = U \Sigma V^T$$

U represents the left singular vector. Σ is a diagonal matrix, and the value of the diagonal is called the singular value. V^T is a right singular vector. The singular value can be obtained by the following formula:

$$\delta_i = \sqrt{\lambda_i}$$

Each singular value is the square root of the characteristic value and is sequenced in Σ in descending order. The drop range of the singular value is extremely high. In most cases, the summation of the top 10% or even 1% of singular values can be extremely close to the total value of all singular values. Thus, deleting some of the singular values can reduce the dimension of the matrix. Instead, we can consider the k singular value and the first k column in the left singular vector as well as the first k row in the right singular vector.

A SVD incremental algorithm [3] is proposed for solving updating problems. It provides a folding-in method to calculate dynamic SVD, since the left and right singular vectors contain the information of the original matrix's rows and columns. A projection equation is also derived from [3] and is used in the folding-in method. The operation after projection is equivalent to adding new items directly; therefore, it can help save much time.

When a new user appears in the original rating matrix, the projection equation can be presented as follows:

$$\hat{d} = d^T \times U_k \times \sum_k^{-1}$$

U_k is the left singular vector, and d is a new row of users ($1 \times m$ matrix). Afterwards, $\hat{d}$ will be generated and can be added directly to the bottom of the matrix, which contains user information without any changing of other values. After that, a new $(m+1) \times k$ matrix will be generated. When a new item is entered into the rating matrix:

$$\hat{t} = t^T \times V_k \times \sum_k^{-1}$$

However, such a folding-in method still has some disadvantages. Incremental SVD characteristic vectors have nonorthogonality and this may cause errors in calculation. Moreover, the folding-in technique only considers new users or items, and it omits situations where ratings in the matrix may change. Furthermore, a sparsity problem always appears in large-scale matrixes on the internet. In order to handle missing or uncertain data in the matrix, [5] provided a new incremental SVD that can decompose the matrix with uncertain data dynamically.

2.3. Incremental computation in clustering

In the study of Gao et al. [9], an incremental clustering method was proposed according to partitional clustering. Datasets always change, and the data mining results change as well. Meanwhile, [9] introduced a method for solving flexibility and data updating problems. When the data are modified, it is solved with changes such as adding, deleting, and updating.

In [10] the authors used density-based clustering (SNN) for incremental computation. The incremental DBSCAN algorithm cannot identify the clustering of different density in one kind of parameter. ISSN solves the problem of IDBSCAN and finds the range of objects affected by updating the data. Then it acquires the affected object by using the pruning strategy in that range. ISSN has better updating properties and can find

the results of different density clusterings. However, the sampling operation of this method will inevitably generate a sampling error.

Hsu and Huang [11] claimed that a hierarchical clustering incremental computation algorithm can calculate hybrid datasets and can use the concept of hierarchy tree to handle similarity problems between data. However, the weight selection in that algorithm necessitates that users have related area knowledge.

2.4. Incremental clustering in collaborative filtering

Clustering is widely used in collaborative filtering recommendations and can help collect elements that have similar attributes. For a huge matrix with severe sparsity, dividing data into small subsets, whose elements have high similarity, is a common method. In [12], the authors promote the coclustering method to reduce dimension. This method is always used in matrixes whose rows and columns are correlated. Therefore, in collaborative filtering recommendations, coclustering is more popular.

The same problem remains—how to solve dynamic data. Wu et al. [13] reported that when a new user is added, the cluster of items that is selected by the new user will be replenished and will then predict that cluster again for recommendation. In addition, when a new item is added, a similar process will be executed. If a new rating is introduced, i.e. the user or item already existed, it will replenish ratings to the cluster of that user or item for generating a new recommendation.

3. Network structure recommendation and incremental bipartite algorithm

3.1. Network structure recommendation mechanism

The network structure recommendation algorithm does not need to consider the content's features of users and products. In this mechanism, users and products are considered as nodes, and the core idea is to find the 'user-product' bipartite network. For every target user who comes into the system, each product will have the ability to recommend other products, rank the recommended results, and recommend future products for the aimed user. Given that the recommendation algorithm is based on a network structure that fits the large-scale data recommendation system, the correlation network needs history information to make the choice.

User-product bipartite: It has x users and y products. If a user i has selected a product α , an edge will be founded, $a_{i\alpha} = 1$. The user and product node degrees concern how many products have been selected and how many users have selected them, respectively.

Resources that are able to be allocated: The products that have been selected by user i are called the resources that are able to be allocated. The products that possess resources have the ability to recommend other products and will have more resources allocated to other products that are waiting to be recommended.

Quota: $w_{\alpha\beta}$ is used to represent the resource quota that product β can allocate to product α . It achieves a recommendation algorithm of heat conduction and mass diffusion under complex circumstances in [18]. These 2 kinds of algorithms are better than traditional coordination filter algorithms, although they show some difference in resource quota expression:

$$\text{For pattern of mass diffusion: } w_{\alpha\beta} = m_{\alpha\beta} = \frac{1}{k_{\beta}} \times \sum_{i \in U} \frac{a_{i\alpha} \times a_{i\beta}}{k_i}$$

$$\text{For pattern of heat conduction: } w_{\alpha\beta} = h_{\alpha\beta} = \frac{1}{k_{\alpha}} \times \sum_{i \in U} \frac{a_{i\alpha} \times a_{i\beta}}{k_i}$$

In this expression, k represents the number of edges of one node that connect to the other nodes. The matrixes of mass diffusion and heat conduction are in a transposition relationship, i.e. $M = H^T$

Propagation matrix: It is made up by $w_{\alpha\beta}$. In the recommendation system of x users and y products:

$$W = \left\{ \begin{array}{ccc} w_{11} & w_{12} & w_{1\beta} \\ \dots & \dots & \dots \\ w_{\alpha 1} & w_{\alpha 2} & w_{\alpha\beta} \\ & \dots & \end{array} \right\} \alpha \in x, \beta \in y$$

Initial resource allocation vector: For every aimed user, we let the initial value of the product that was selected by the user be 1 and let the other be 0. In this way, an n -dimension vector that was composed of 0 and 1 could be obtained. This vector expresses its initial allocated value, and we record this vector to be f .

Final resource distribution: Use f' to represent the final resource distribution for products through the calculation of the propagation matrix, i.e. the final resource allocation value:

$$f' = w \times f$$

Recommendation strategy: Pick products that have not been selected by i for any aimed user i in the bipartite network according to the degree recommendation f' and then rank products and recommend those to the user.

3.2. Incremental algorithm of network structure recommendation mechanism

Although recommendation algorithms of heat conduction and mass diffusion could provide accurate multiple recommendations, they could not function well in a dynamic circumstance of bipartite network. An incremental computation of network structure recommendation mechanism had been proposed by Jin et al. (<http://arxiv.org/pdf/0911.4910v1.pdf>), who defined unit changes as a new edge to be added to the bipartite network. If we assume that L is the total number of edges, then the target of incremental computation changes concerns how to avoid duplication when doing overall computation to figure out $f^{(L+1)}$.

When the bipartite network adds an edge, it means that a specific user i has founded a new connection with a project α . As illustrated in Figures 1a–1d, this change results in 4 types of changes for the network. $i \rightarrow \alpha$ means that the added edge chooses α for user i . The user set is U and for every student i , $i \in U$.

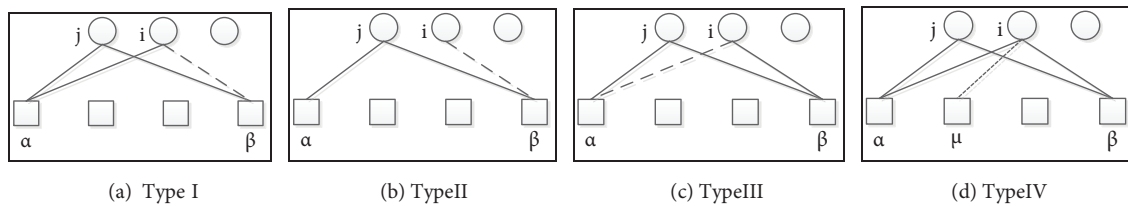


Figure 1. Four kinds of incremental bipartite types.

- 1) Initial $a_{i\alpha} = 1, a_{i\beta} = 0$; when $i \rightarrow \beta$, then $k'_\beta = k_\beta + 1, a'_{i\beta} = 1, k'_i = k_i + 1$.

$$w'_{\alpha\beta} = \frac{1}{k'_\beta} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a'_{i\alpha} \times a'_{i\beta}}{k'_i} \right) = \frac{1}{k_\beta + 1} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{1}{k_\beta + 1} \times \frac{1}{k_i + 1} \quad (1)$$

$$w_{\alpha\beta} = \frac{1}{k_{\beta}} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i} \right) = \frac{1}{k_{\beta}} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} \quad (2)$$

$$\delta_{\alpha\beta} = w'_{\alpha\beta} - w_{\alpha\beta} = \frac{-1}{k_{\beta}+1} \times w_{\alpha\beta} + \frac{1}{(k_{\beta}+1) \times (k_i+1)} \quad (3)$$

2) Initial $a_{i\alpha} = 0, a_{i\beta} = 0$; when $i \rightarrow \beta$, then $k'_{\beta} = k_{\beta} + 1, a'_{i\beta} = 1, k'_i = k_i + 1$.

$$w'_{\alpha\beta} = \frac{1}{k_{\beta}+1} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{0 \times 1}{k_i+1} \right) \quad (1)$$

$$w_{\alpha\beta} = \frac{1}{k_{\beta}} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} \quad (2)$$

$$\delta_{\alpha\beta} = \frac{1}{k_{\beta}} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} = \frac{-1}{k_{\beta}+1} \times w_{\alpha\beta} \quad (4)$$

3) Initial $a_{i\alpha} = 0, a_{i\beta} = 1$; when $i \rightarrow \alpha$, then $k'_{\beta} = k_{\beta} a_{i\alpha} = 1, k'_i = k_i + 1$.

$$w'_{\alpha\beta} = \frac{1}{k_{\beta}} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i+1} \right) \quad (1)$$

$$w_{\alpha\beta} = \frac{1}{k_{\beta}} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} \quad (2)$$

$$\delta_{\alpha\beta} = \frac{1}{k_{\beta} \times (k_i+1)} \quad (5)$$

4) Initial $\mu \neq \alpha, \mu \neq \beta, a_{i\alpha} = 1, a_{i\beta} = 1$; when $i \rightarrow \mu$, then $k'_{\beta} = k_{\beta}, k'_i = k_i + 1$.

$$w'_{\alpha\beta} = \frac{1}{k_{\beta}} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i+1} \right) \quad (1)$$

$$w_{\alpha\beta} = \frac{1}{k_{\beta}} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i} \right) \quad (2)$$

$$\delta_{\alpha\beta} = \frac{-1}{k_{\beta} \times (k_i+1) \times k_i} \quad (6)$$

There are 4 kinds of incremental expressions in the work of Jin et al. When new edges are added, the changes can be directly calculated, and this algorithm will save time from recalculating the whole propagation matrix.

3.3. Algorithm application

Algorithm application has been used in the learning resource recommendation system in order to improve the efficiency and quality of obtaining learning resources from the platform of autonomic learning for college and university students. Student learning interest can be obtained by modeling their borrowing records. This can provide more relevant or more necessary books to students according to the recommendation list. Thus, this method can enhance the efficiency of student autonomic learning.

The recommendation system, which is based on borrowing records, has users that consist of college students and items that consist of books. There will be an edge between 2 nodes if a student has borrowed a book. There will only be one edge if the student has borrowed the same book several times in the unweighted bipartite network.

Algorithm description

Input: existed propagation matrix w , new data set T_α

Output: related value in the update propagation matrix w

- 1) For each $t_{i\alpha}$ in T
- 2) Check out whether i existed in the previous data or not. If it existed, then carry out the next circulation.
- 3) Find out the set T_β of β that is related to α
- 4) For each β in T_β
- 5) If user i has borrowed book β
- 6) $\delta_{\beta\alpha} = \frac{-1}{k_\alpha+1} \times w_{\beta\alpha} + \frac{1}{(ik_\alpha+1)j \times (ik_i+1)j}$ (3)
- 7) $\delta_{\alpha\beta} = \frac{1}{k_\beta \times (k_i+1)}$ (9)
- 8) Otherwise
- 9) $\delta_{\beta\alpha} = \frac{-1}{k_\alpha+1} \times w_{\beta\alpha}$ (4)
- 10) $\delta_{\alpha\beta} = \frac{-1}{k_\beta \times (ik_i+1)j \times k_i}$ (12)
- 11) Use $\delta_{\beta\alpha}$, $\delta_{\alpha\beta}$ to update the corresponding records in database w .

4. Incremental weighted bipartite recommendation algorithm

4.1. Algorithm principle

In the weighted bipartite network there are x users and y items. If one user i selects one item α for r times, r edges exist between that user and the item ($a_{i\alpha} = r_{i\alpha}$) and we assume new borrowing records if one user i chooses an item α for s times. Such changes cause 4 types of updating rules. $i \xrightarrow{s} \alpha$ means that user i selects α for s times:

(1) Initial $a_{i\alpha}=r_{i\alpha}\neq 0$, $a_{i\beta}=r_{i\beta}$; when $i \xrightarrow{s} \beta$, then $a'_{i\alpha}=a_{i\alpha}$, $a'_{i\beta}=r_{i\beta}+s$, $k'_i=k_i+s$, $k'_\beta=k_\beta+s$.

$$\begin{aligned}
 w'_{\alpha\beta} &= \frac{1}{k'_\beta} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a'_{i\alpha} \times a'_{i\beta}}{k'_i} \right) = \frac{1}{k_\beta+s} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{r_{i\alpha} \times (r_{i\beta}+s)}{k_i+s} \right) \\
 w_{\alpha\beta} &= \frac{1}{k_\beta} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{r_{i\alpha} \times r_{i\beta}}{k_i} \right) \\
 \delta &= \frac{-s \times k_i}{k_i \times k_\beta \times (k_\beta+s)} \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} - \frac{r_{i\alpha} \times r_{i\beta}}{k_i \times k_\beta} + \frac{r_{i\alpha} \times (r_{i\beta}+s)}{(k_i+s) \times (k_\beta+s)} \\
 &= \frac{-1}{k_i \times (k_\beta+s)} \times \left(\frac{s \times k_i}{k_\beta} \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{r_{i\alpha} \times r_{i\beta} \times (k_\beta+s)}{k_\beta} \right) + \frac{r_{i\alpha} \times (k_\beta+s)}{(k_i+s) \times (k_\beta+s)} \\
 &= \frac{-1}{k_i \times (k_\beta+s)} \times \left[s \times k_i \times (k_\beta+s) + \frac{r_{i\alpha} \times r_{i\beta} \times (k_\beta+s)}{k_\beta} \right] + \frac{r_{i\alpha} \times (k_\beta+s)}{(k_i+s) \times (k_\beta+s)} \\
 &= \frac{-1}{k_i \times (k_\beta+s)} \times [s \times k_i \times w_{\alpha\beta} - r_{i\alpha} \times r_{i\beta}] + \frac{r_{i\alpha} \times (k_\beta+s)}{(k_i+s) \times (k_\beta+s)} \\
 &= \frac{-s}{k_\beta+s} \times w_{\alpha\beta} - \frac{r_{i\alpha} \times r_{i\beta}}{k_i \times (k_\beta+s)} + \frac{r_{i\alpha} \times (k_\beta+s)}{(k_i+s) \times (k_\beta+s)} \tag{7}
 \end{aligned}$$

If initial $a_{i\beta}=r_{i\beta}=0$, as shown in Figures 2a and 2b, the formula can be simplified further:

$$\delta = \frac{-s}{k_\beta+s} \times w_{\alpha\beta} + \frac{r_{i\alpha} \times s}{(ik_i+s)j \times (ik_\beta+s)j} \tag{8}$$

(2) Initial $a_{i\alpha}=r_{i\alpha}=0$, $a_{i\beta}=r_{i\beta}$; when $i \xrightarrow{s} \beta$, then $a'_{i\alpha}=0$, $a'_{i\beta}=r_{i\beta}+s$, $k'_i=k_i+s$, $k'_\beta=k_\beta+s$.

$$\begin{aligned}
 w'_{\alpha\beta} &= \frac{1}{k'_\beta+s} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} \\
 w_{\alpha\beta} &= \frac{1}{k_\beta} \times \sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} \\
 \delta &= \frac{-s}{k_\beta+s} \times w_{\alpha\beta} \tag{9}
 \end{aligned}$$

As for $a_{i\beta}=r_{i\beta}=0$, as shown in Figures 2c and 2d, since $a_{i\alpha}=0$, $a'_{i\alpha}=0$, the incremental changes of $a'_{i\beta}=r_{i\beta}+$ cannot affect δ , and the equation remain the same

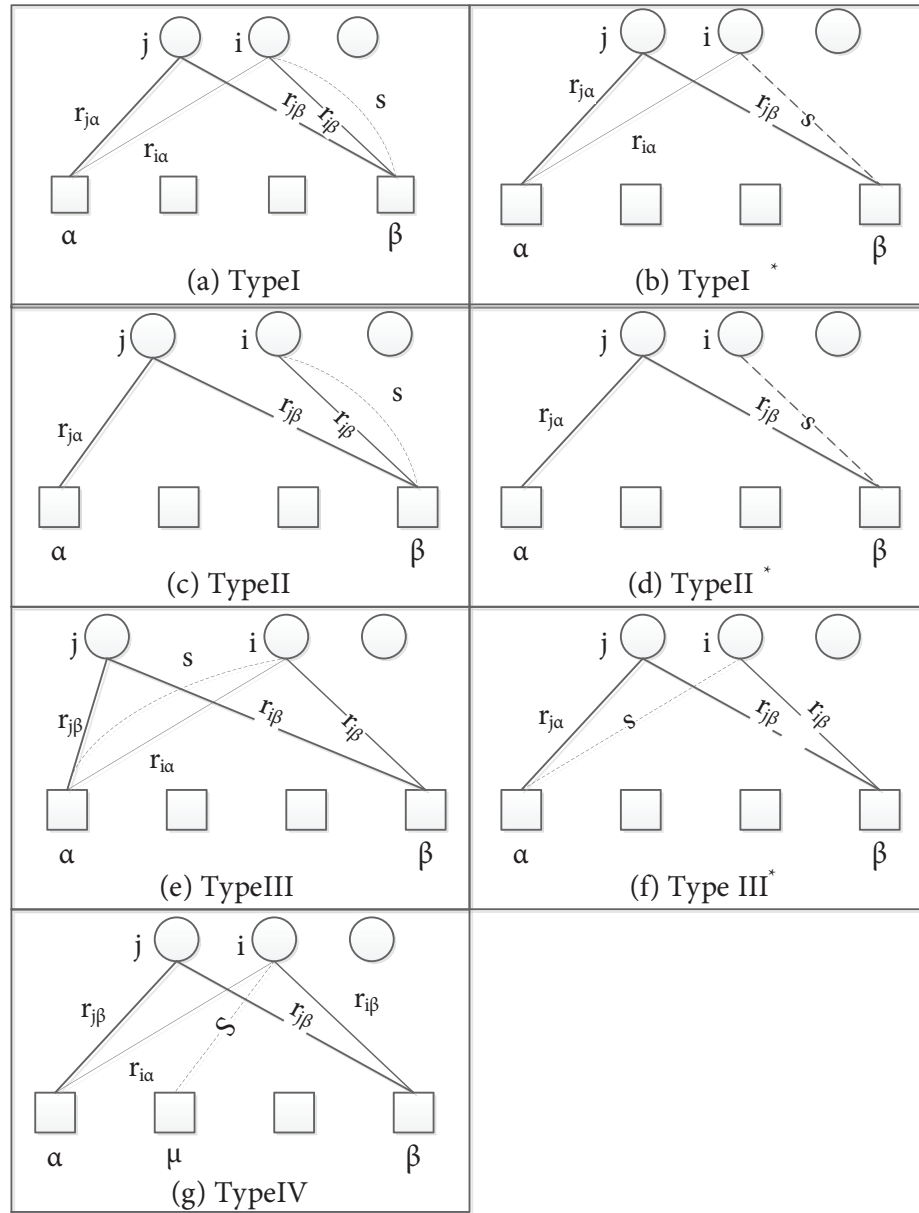


Figure 2. Four types of incremental weighted bipartite networks.

(3) Initial $a_{i\alpha}=r_{i\alpha}$, $a_{i\beta}=r_{i\beta} \neq 0$, when $i \xrightarrow{s} \alpha$, then $a'_{i\alpha}=r_{i\alpha}+s$, $a'_{i\beta}=a_{i\beta}=r_{i\beta}$, $k'_i=k_i+s$, $k'_\beta=k_\beta$

$$w'_{\alpha\beta} = \frac{1}{k'_\beta} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a'_{i\alpha} \times a'_{i\beta}}{k'_i} \right)$$

$$w_{\alpha\beta} = \frac{1}{k_\beta} \times \left(\sum_{j \in U, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i} \right)$$

$$\delta = \frac{r_{i\beta} \times (k_{i\alpha} + s)}{(k_i + s) \times k_\beta} - \frac{r_{i\alpha} \times r_{i\beta}}{k_i k_\beta} = \frac{s \times r_{i\beta} \times (k_i - r_{i\alpha})}{k_i k_\beta \times (k_i + s)} \quad (10)$$

If $a_{i\alpha} = r_{i\alpha} = 0$ at first, as shown in Figures 2e and 2f, the equation can be simplified as follows:

$$\delta = \frac{r_{i\beta} \times s}{(k_i + s) \times k_\beta} \quad (11)$$

- (4) At first $a_{i\alpha} = r_{i\alpha} \neq 0, a_{i\beta} = r_{i\beta} \neq 0, \forall \mu$, and $\mu \neq \alpha, \mu \neq \beta, a_{i\mu} = 0, i \xrightarrow{s} \mu$, then $a'_{i\alpha} = a_{i\alpha} = r_{i\alpha}, a'_{i\beta} = a_{i\beta}, a'_{i\mu} = s, k'_\beta = k_\beta, k'_i = k_i + s$ it can be shown on Figure 2(g):

$$\begin{aligned} w'_{\alpha\beta} &= \frac{1}{k'_\beta} \left(\sum_{j \in u, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a'_{i\alpha} \times a'_{i\beta}}{k'_i} \right) = \frac{1}{k_\beta} \left(\sum_{j \in u, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{r_{i\alpha} \times r_{i\beta}}{k_i + s} \right) \\ w_{\alpha\beta} &= \frac{1}{k_\beta} \left(\sum_{j \in u, j \neq i} \frac{a_{j\alpha} \times a_{j\beta}}{k_j} + \frac{a_{i\alpha} \times a_{i\beta}}{k_i} \right) \\ \delta &= \frac{-s \times r_{i\alpha} \times r_{i\beta}}{k_\beta \times (k_i + s) \times k_i} \end{aligned} \quad (12)$$

4.2. Applications

In the weighted bipartite network one student may borrow a book several times. We consider borrowing times as edge weight. Then the algorithm description is shown as follows:

Algorithm description

Input: existing propagation matrix w , new data set T_α .

Output: related value in the update propagation matrix w .

- 1) For each $t_{i\alpha}$ in T
- 2) Check out whether i existed in the previous data or not, and obtain the incremental weight s . If it existed, then carry out the next circulation.
- 3) Find out the set T_β of β that is related to α
- 4) For each β in T_β
- 5) If user i has borrowed book β
- 6) If user i has borrowed book α
- 7) $\delta_{\beta\alpha} = \frac{-s}{k_\beta + s} \times w_{\alpha\beta} - \frac{r_{i\alpha} \times r_{i\beta}}{k_i \times (k_\beta + s)} + \frac{r_{i\alpha} \times (k_\beta + s)}{(k_i + s) \times (k_\beta + s)}$ (7)
- 8) $\delta_{\alpha\beta} = \frac{s \times r_{i\beta} \times (k_i - r_{i\alpha})}{k_i k_\beta \times (k_i + s)}$ (10)
- 9) Otherwise

$$10) \delta_{\beta\alpha} = \frac{-s}{k_{\beta}+s} \times w_{\alpha\beta} + \frac{r_{i\alpha} \times s}{(k_i+s) \times (k_{\beta}+s)} \quad (8)$$

$$11) \delta_{\alpha\beta} = \frac{r_{i\beta} \times s}{(k_i+s) \times k_{\beta}} \quad (11)$$

12) Otherwise

$$13) \delta_{\beta\alpha} = \frac{-s}{k_{\beta}+s} \times w_{\beta\alpha} \quad (9)$$

$$14) \delta_{\alpha\beta} = \frac{-s \times r_{i\alpha} \times r_{i\beta}}{k_{\beta} \times (k_i+s) \times k_i} \quad (12)$$

15) Use $\delta_{\beta\alpha}, \delta_{\alpha\beta}$ to update the corresponding records in database w .

5. Experiment and results

In this section, several experiments are conducted in order to compare the performance of the 2 incremental algorithms to the original algorithm, which requires global computation of the propagation matrix.

5.1. Dataset description

According to the analysis of the algorithm application, the dataset in this paper is the collection of borrowing records from the platform of the autonomic learning for college and university students. When a student borrows books, the records will be saved in the platform. Borrowing records reflect student interests directly. A total of 8000 books were chosen randomly at the stage of testing the algorithm's validity. Each borrowing record can be considered as an edge in the bipartite network. The duplicated borrowing records can illustrate student preference for a specific book. The dataset contains 15,757 users and 399,899 borrowing records. In order to simulate the incremental process, another 100 borrowing records were chosen as incremental data. A total of 70% of the data were randomly selected as the training set in order to train the bipartite network model. The rest were selected as a testing set to test the performance of the algorithm.

5.2. Metrics

In order to better evaluate the incremental algorithm of the weighted and unweighted bipartite network recommendations, we need to consider 2 aspects of its performance. Time consumption must be considered, since the incremental algorithm only calculates a small part of the whole data. This method could relieve the computational burden significantly and save a lot of time. Therefore, it can be widely utilized for dynamic data. However, the incremental algorithm may produce errors. The algorithm should guarantee the performance of the recommendation when saving computation time. Thus, precision, recall, and F1 measure are considered for the recommendation results evaluation.

Assume: The number of book is n_b . The number of students who have borrowed the book is n_i . The book that has been recommended to a student is k and the rank that has been recommended to the student is top-N. The actual number of books that i has borrowed is $n_{ik}, n_{ik} \leq k$.

Precision: The ratio of the books on the recommendation list that were borrowed by students to all the books that were recommended is $P = \frac{n_{ik}}{K}$

Recall: The ratio of the books on the recommendation list that were borrowed by students to all the borrowed books in the system is $R = \frac{n_{ik}}{n_i}$.

F1 measure: There is a contradiction between the precision and recall rates. When K increases, the recall rate increases, whereas the precision decreases. F measure uses the same weight to combine the precision and recall; that is, $F = \frac{2PR}{P+R}$

5.3. Results and analysis

The experiments compared the performance of the weighted bipartite network recommendation to the unweighted bipartite network recommendation. In addition, the experiments showed the efficiency and performance of their incremental algorithm. All the experiments were conducted on a standard personal computer with Inter Core i7-3632 QM and 8 G RAM.

Figure 3 shows the recommendation results of the weighted bipartite network recommendation and the unweight network recommendation. The F1 measure curve should be used for evaluation. When K is less than 55, the F1 measure of the weighted bipartite network recommendation is higher than that of the unweighted bipartite network recommendation. This means that the weighted bipartite recommendation has better performance. In real application, users always focus merely on the top of the recommendation list. Therefore, a longer recommendation list is meaningless. From the observation of the recommendation result, we can see that books with higher weight are recommended at the top of the recommendation list. Thus, the weighted bipartite network recommendation algorithm performs better.

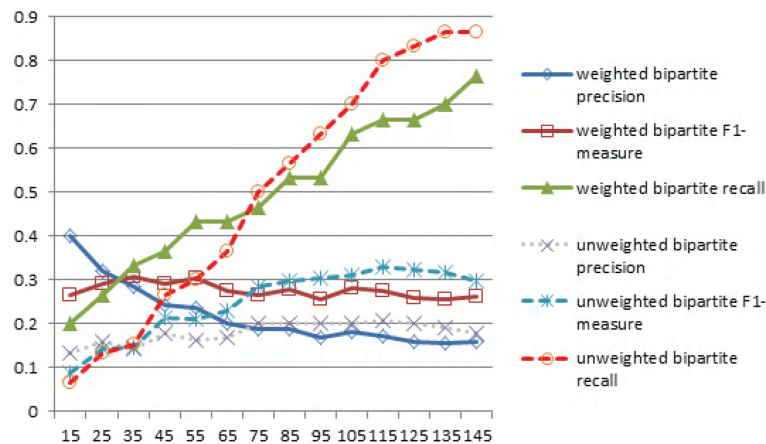


Figure 3. Comparison of weighted bipartite network recommendations and unweighted bipartite network recommendations.

Figure 4 shows the comparison of the incremental unweighted bipartite recommendation and the unweighted bipartite recommendation. The graphs of the 2 algorithms almost completely overlap. There is a difference between them when K equals 45. This result demonstrates that the incremental algorithm of the unweighted bipartite recommendation only has a slight impact on the recommendation's performance. The incremental equation not only improves computation efficiency, but also provides an error analysis. The value of the type IV equation in unweighted bipartite recommendation is relatively small and will lead to a greater burden in computation. Figure 5 shows the experiment result of the unweighted bipartite recommendation, which ignores type IV. Clearly, the neglect of type IV will cause more errors. When K equals 45 and 95, several differences show up between the 2 algorithms. Although the incremental algorithm causes small errors, it improves the efficiency of computation significantly. Table 1 shows the time consumption of these 3 situations. It demonstrates that the incremental algorithm can save a lot of time. Less computation time could

update the data quickly and would be suitable for the recommendation system. The time consumption of these 2 incremental algorithms is only 40.0% and 14.69% of the original global computation's time consumption, respectively.

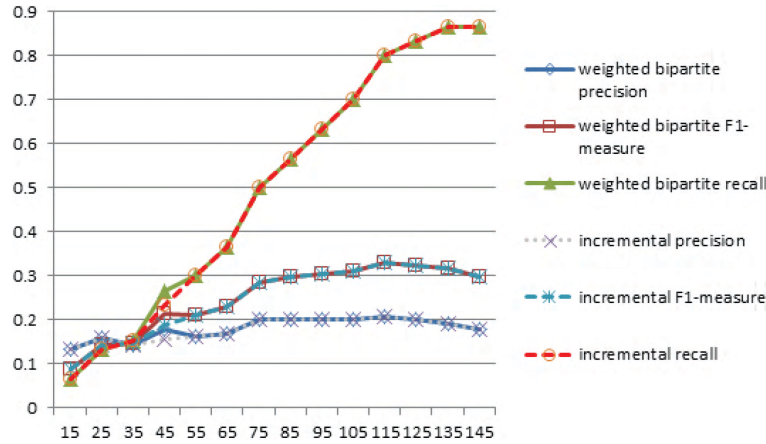


Figure 4. Comparison of incremental unweighted bipartite recommendations and unweighted bipartite recommendations.

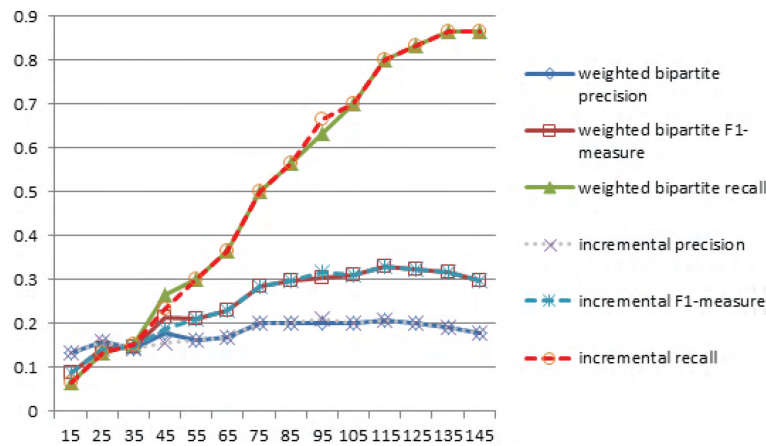


Figure 5. Comparison of incremental unweighted bipartite recommendations and unweighted bipartite recommendations while ignoring type IV.

Table 1. Comparison of time consumption.

	Original global computation (min)	Incremental computation (min)	Incremental computation, ignore type IV (min)
Unweighted	4281	1712	629

As for the weighted bipartite network recommendation, Figure 6 shows the performance of the incremental algorithm. When K equals 145, there is an error between the incremental algorithm and the original weighted bipartite recommendation. This also shows that the incremental algorithm has some impact on recommendation performance. By analyzing the last type of incremental equation in the weighted bipartite recommendation, we can find that new weight will have some influence on the value of the equation. However, in real application, new weights are not very big in a specific time duration. Type IV also brings a great burden in computation.

Figure 7 shows the comparison of weighted bipartite recommendation and incremental weighted bipartite recommendation directly, without type IV. When K equals 85 and 145, the recommendation results of these 2 algorithms show some differences. The incremental algorithm affects the recommendation's performance; however, it reduces time consumption in the weighted bipartite recommendation. Table 2 compares their time consumption. The time consumption of those 2 incremental algorithms is only 10.52% and 4.3% of the original global computation, respectively.

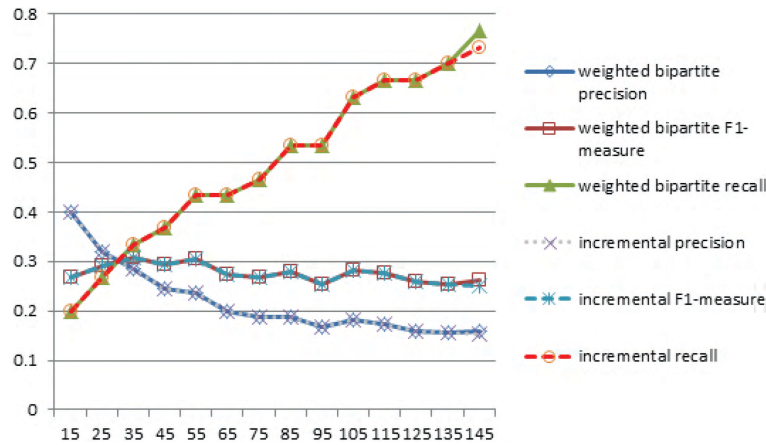


Figure 6. Comparison of incremental weighted bipartite recommendations and weighted bipartite recommendations.

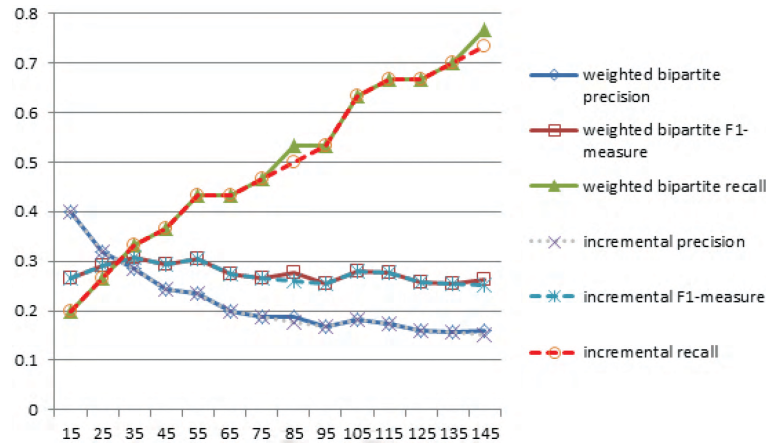


Figure 7. Comparison of incremental weighted bipartite recommendations and weighted bipartite recommendations while ignoring type IV.

Table 2. Comparison of time consumption.

	Original global computation (min)	Incremental computation (min)	Incremental computation, ignore type IV (min)
Weighted	6978	734	300

In summation, the incremental algorithm can save much time for computation; however, it will affect the recommendation's precision. In order to avoid the accumulation effect of error in incremental computation, especially in neglecting type IV situation, the system should redo a global computation in an appropriate period of time.

6. Conclusion

This paper proposed a type of personalized recommendation algorithm based on an incremental weighted bipartite algorithm that can allow the difference in ratings or the times of the same choice to effectively affect the rank of the recommendation. This paper also discussed the incremental factor method of the calculated bipartite recommendation mechanism of the network structure. It also gave 4 types of paradigm expression and the incremental factors' detailed reduction. This makes it more convenient for different application systems to model and improve the algorithm in the actual situation.

The paper examined borrowing records from a university library. It used them for a modeling recommendation system and gave algorithm implementation as well as acquiring a personalized recommendation order. Additionally, 8000 books and 15,757 students were selected for a bipartite and weighted bipartite incremental algorithm simulation. This simulation tested the system's efficiency by using precision, recall rate, and F1 measure. Finally, we used several incremental datasets to obtain their response time for testing the algorithm's performance in a real application.

In the future, we will conduct research on the computation complexity of the bipartite recommendation and its incremental algorithm. The user-item matrix will generate an $N \times M$ propagation matrix. How to use matrix decomposition for dimension reduction or other techniques to decrease computation complexity remains a challenge.

Acknowledgments

This work was supported by the National Key Project of the Scientific and Technical Supporting Programs of China (Grant No. 2013BAH10F01, 2013BAH07F02, 2014BAH26F02); the Research Fund for the Doctoral Program of Higher Education (Grant No. 20110005120007); the Beijing Higher Education Young Elite Teacher Project (Grant No. YETP0445); the Co-construction Program with the Beijing Municipal Commission of Education; and the Engineering Research Center of Information Networks, Ministry of Education.

References

- [1] Papagelis M, Rousidis I, Plexousakis D, Theoharopoulos E. Incremental collaborative filtering for highly-scalable recommendation algorithms. In: ISMIS 2005 International Symposium on Methodologies for Intelligent Systems; 25–28 May 2005; Saratoga Springs, NY, USA. Berlin, Germany: Springer-Verlag. pp. 553–561.
- [2] Miranda C, Jorge AM. Incremental collaborative filtering for binary ratings. In: IEEE 2008 International Conference on Web Intelligence; 9–12 December 2008; Sydney, Australia. New York, NY, USA: IEEE. pp. 389–392.
- [3] Berry MW, Dumais ST, O'Brien GW. Using linear algebra for intelligent information retrieval. SIAM Rev 1995; 37: 573–595.
- [4] Brand M. Incremental singular value decomposition of uncertain data with missing values. In: ECCV 2002 European Conference on Computer Vision; 28–31 May 2002; Copenhagen, Denmark. Berlin, Germany: Springer-Verlag. pp. 707–720.
- [5] Chandramouli B, Levandoski JJ, Eldawy A, Mokbel MF. Streamrec: a real-time recommender system. In: ACM 2011 SIGMOD Management of Data International Conference; 12–16 June 2011; Athens, Greece. New York, NY, USA: ACM. pp. 1243–1245.
- [6] Yang X, Zhang Z, Wang K. Scalable collaborative filtering using incremental update and local link prediction. In: ACM 2012 Information and Knowledge Management International Conference; 29 October–2 November 2012; Maui, HI, USA. New York, NY, USA: ACM. pp. 2371–2374.

- [7] Hwang W, Lee H, Kim S, Lee M. On using category experts for improving the performance and accuracy in recommender systems. In: ACM 2012 Information and Knowledge Management International Conference; 29 October–2 November 2012; Maui, HI, USA. New York, NY, USA: ACM. pp. 2355–2358.
- [8] Liu NN, Zhao M, Xiang E, Yang Q. Online evolutionary collaborative filtering. In: ACM 2010 Recommender Systems Conference; 26–30 September 2010; Barcelona, Spain. New York, NY, USA: ACM. pp. 95–102.
- [9] Gao X, Feng Y, Feng X. Incremental clustering algorithm based on k-medoids. *Jisuanji Gongcheng* 2005; 31:181–183 (in Chinese with abstract in English).
- [10] Huanliang S, Fei Q, Junling L, Yeli Z. IncSNN: an incremental clustering algorithm based on density. *Jisuanji Yanjiu yu Fazhan* 2006; 43: 213–309 (in Chinese with abstract in English).
- [11] Hsu C, Huang Y. Incremental clustering of mixed data based on distance hierarchy. *Expert Sys Appl* 2008; 35: 1177–1185.
- [12] George T, Merugu S. A scalable collaborative filtering framework based on co-clustering. In: IEEE 2005 Data Mining International Conference; 27–30 November 2005; Houston, USA. New York, NY, USA: IEEE. pp. 625–628.
- [13] Wu H, Wang Y, Wang Z, Wang X, Du S. Two-phase collaborative filtering algorithm based on co-clustering. *Ruan Jian Xue Bao* 2010; 21: 1042–1054 (in Chinese with abstract in English).
- [14] Khoshneshin M, Nick Street W. Incremental collaborative filtering via evolutionary co-clustering. In: ACM 2010 Recommender Systems Conference; 26–30 September 2010; Barcelona, Spain. New York, NY, USA: ACM. pp. 325–328.
- [15] Zhou T, Su R, Liu R, Jiang L, Wang B, Zhang Y. Accurate and diverse recommendations via eliminating redundant correlations. *New J Phys* 2009; 11: 123008.
- [16] Zhou T, Ren J, Medo M, Zhang Y. Bipartite network projection and personal recommendation. *Phys Rev E* 2007; 76: 046115.
- [17] Liu J, Shang M, Chen D. Personal recommendation based on weighted bipartite networks. In: IEEE 2009 Fuzzy Systems and Knowledge Discovery International Conference; 14–16 August 2009; Tianjin, China. New York, NY, USA: IEEE. pp. 134–137.
- [18] Zhang Y, Blattner M, Yu Y. Heat conduction process on community networks as a recommendation model. *Phys Rev Lett* 2007; 99: 154301.