How to project a bipartite network?

Tao Zhou[1,2,*], Jie Ren[1], Matúš Medo[1], and Yi-Cheng Zhang[1,3,†]

[1]*Department of Physics, University of Fribourg, Chemin du Muse 3, CH-1700 Fribourg, Switzerland*
[2]*Department of Modern Physics and Nonlinear Science Center,*
University of Science and Technology of China, Hefei Anhui, 230026, PR China
[3]*Information Economy and Internet Research Laboratory,*
University of Electronic Science and Technology of China, Chengdu Sichuan, 610054, PR China
(Dated: February 11, 2013)

The one-mode projecting is extensively used to compress the bipartite networks. Since the one-mode projection is always less informative than the bipartite representation, a proper weighting method is required to better retain the original information. In this article, inspired by the network-based resource-allocation dynamics, we raise a weighting method, which can be directly applied in extracting the hidden information of networks, with remarkably better performance than the widely used global ranking method as well as collaborative filtering. This work not only provides a creditable method in compressing bipartite networks, but also highlights a possible way for the better solution of a long-standing challenge in modern information science: How to do personal recommendation?

PACS numbers: 89.75.Hc, 87.23.Ge, 05.70.Ln

I. INTRODUCTION

The last few years have witnessed a tremendous activity devoted to the understanding of complex networks [1, 2, 3, 4, 5, 6, 7]. A particular class of networks is the *bipartite networks*, whose nodes are divided into two sets, X and Y, and only the connection between two nodes in different sets is allowed (as illustrated in Fig. 1a). Many systems are naturally modeled as bipartite networks [8]: Human sexual network [9] is consisted of men and women, metabolic network [10] is consisted of chemical substances and chemical reactions, etc. Two kinds of bipartite networks should be paid more attention for their particular significance in social, economic and information systems. One is the so-called *collaboration network*, which is generally defined as a networks of actors connected by a common collaboration act [11, 12]. Examples are numerous, including scientists connected by coauthoring a scientific paper [13, 14], movie actors connected by costarring the same movie [1, 15], and so on. Moreover, the concept of collaboration network is not necessarily restricted within social systems (see, for example, recent reports on technological collaboration of software [16] and urban traffic systems [17]). Although the collaboration network is usually displayed by the one-mode projection on actors (see later the definition), its fully representation is a bipartite network. The other one is named *opinion network* [18, 19], where each node in the *user-set* is connected with its collected objects in the *object-set*. For example, listeners are connected with the music groups they collected from music-sharing library (e.g. *audioscrobbler.com*) [20, 21], web-users are connected with the webs they collected in a bookmark

site (e.g. *delicious*) [22], customers are connected with the books they bought (e.g. *Amazon.com*) [23, 24].

Recently, a large amount of attention is addressed to analyzing [8, 20, 25, 26, 27] and modeling [28, 29, 30] bipartite network. However, for the convenience of directly showing the relations among a particular set of nodes, the bipartite network is usually compressed by one-mode projecting. The one-mode projection onto X (X-projection for short) means a network containing only X-nodes, where two X-nodes are connected when they have at least one common neighboring Y-node. Fig. 1b and Fig. 1c show the resulting networks of X-projection and Y-projection, respectively. The simplest way is to project the bipartite network onto an unweighted network [13, 14, 31, 32, 33], without taking into account of the frequency that a collaboration has been repeated. Although some topological properties can be qualitatively obtained from this unweighted version, the loss of information is obvious. For example, if two listeners has collected more than 100 music groups each (it is a typical number of collections, like in *audioscrobbler.com*, the average number of collected music groups per listener is 140 [20]), and only one music group is selected by both listeners, one may conclude that those two listeners probably have different music taste. On the contrary, if nearly 100 music groups belong to the overlap, those two listeners are likely to have very similar habits. However, in the unweighted listener-projection, this two cases have exactly the same graph representation.

Since the one-mode projection is always less informative than the original bipartite network, to better reflect structure of the network, one has to use the bipartite graph to quantify the weights in the projection graph. A straightforward way is to weight an edge directly by the number of times the corresponding partnership repeated [34, 35]. This simple rule is used to obtain the weights in Fig. 1b and Fig. 1c for X-projection and Y-projection, respectively. This weighted network is much more infor-

[*]Electronic address: zhutou@ustc.edu
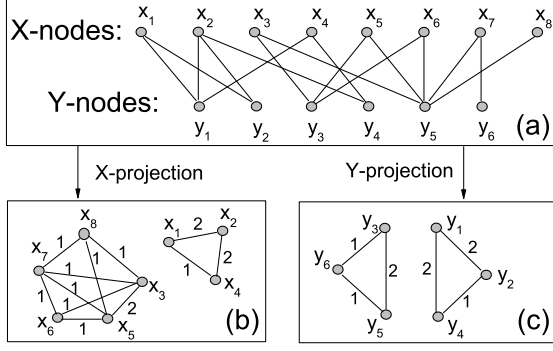[†]Electronic address: yi-cheng.zhang@unifr.ch

FIG. 1: Illustration of a bipartite network (a), as well as its X-projection (b) and Y-projection (c). The edge-weight in (b) and (c) is set as the number of common neighbors in Y and X, respectively.

mative than the unweighted one, and can be analyzed by standard techniques for unweighted graphs since its weights are all integers [36]. However, this method is also quantitatively biased. Li *et al.* [37] empirically studied the scientific collaboration networks, and pointed out that the impact of one additional collaboration paper should depend on the original weight between the two scientists. For example, one more co-authorized paper for the two authors having only co-authorized one paper before should have higher impact than for the two authors having already co-authorized 100 papers. This saturation effect can be taken into account by introducing a hyperbolic tangent function onto the simple count of collaborated times [37]. As stated by Newman that two scientists whose names appear on a paper together with many other coauthors know one another less well on average than two who were the sole authors of a paper [14], to consider this effect, he introduced the factor $1/(n-1)$ to weaken the contribution of collaborations involving many participants [38, 39], where n is the number of participants (e.g. the number of authors of a paper).

How to weight the edges is the key question of the one-mode projections and their use. However, we lack a systematic exploration of this problem, and no solid base of any weighting methods have been reported thus far. For example, one may ask the physical reason why using the hyperbolic tangent function to address the saturation effect [37] rather than other infinite possible candidates. In addition, for simplicity, the weighted adjacent matrix $\{w_{ij}\}$ is always set to be symmetrical, that is, $w_{ij} = w_{ji}$. However, as in scientific collaboration networks, different authors may assign different weights to the same co-authorized paper, and it is probably the case that the author having less publications may give a higher weight, vice versa. Therefore, a more natural weighting method may be not symmetrical. Another blemish in the prior methods is that the information contained by the edge whose adjacent X-node (Y-node) is of degree

one will be lost in Y-projection (X-projection). This information loss may be serious in some real opinion networks. For example, in the user-web network of *delicious* (http://del.icio.us), a remarkable fraction of webs have been collected only once, as well as a remarkable fraction of users have collected only one web. Therefore, both the user-projection and web-projection will squander a lot of information. Since more than half publications in *Mathematical Reviews* have only one author [31], the situation is even worse in mathematical collaboration network.

In this article, we propose a weighting method, with asymmetrical weights (i.e., $w_{ij} \neq w_{ji}$) and allowed self-connection (i.e., $w_{ii} > 0$). This method can be directly applied as a personal recommendation algorithm, which performs remarkably better than the widely used *global ranking method* (GRM) and *collaborative filtering* (CF).

II. METHOD

Without loss of generality, we discuss how to determine the edge-weight in X-projection, where the weight w_{ij} can be considered as the importance of node i in j's sense, and it is generally not equal to w_{ji}. For example, in the book-projection of a customer-book opinion network, the weight w_{ij} between two books i and j contributes to the strength of book i recommendation to a customer provided he has brought book j. In the scientific collaboration network, w_{ij} reflects how likely is j to choose i as a contributor for a new research project. More generally, we assume a certain amount of a resource (e.g. recommendation power, research fund, etc.) is associated with each X-node, and the weight w_{ij} represents the proportion of the resource j would like to distribute to i.

To derive the analytical expression of w_{ij}, we go back to the bipartite representation. Since the bipartite network itself is unweighted, the resource in an arbitrary X-node should be equally distributed to its neighbors in Y. Analogously, the resource in any Y-node should be equally distributed to its X-neighbors. As shown in Fig. 2a, the three X-nodes are initially assigned weights x, y and z. The resource-allocation process consists of two steps; first from X to Y, then back to X. The amount of resource after each step is marked in Fig. 2b and Fig. 2c, respectively. Merging these two steps into one, the final resource located in those three X-nodes, denoted by x', y' and z', can be obtained as:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} 11/18 & 1/6 & 5/18 \\ 1/9 & 5/12 & 5/12 \\ 5/18 & 5/12 & 4/9 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}. \quad (1)$$

Note that, this 3×3 matrix are column normalized, and the element in the ith row and jth column represents the fraction of resource the jth X-node transferred to the ith X-node. According to the above description, this matrix is the very weighted adjacent matrix we want.
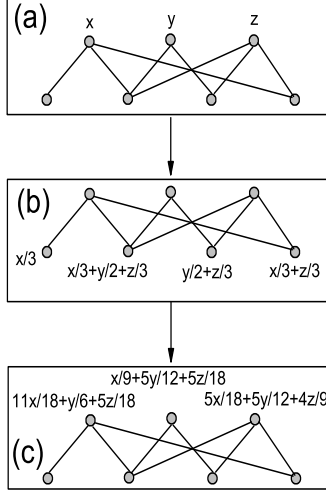
FIG. 2: Illustration of the resource-allocation process in bipartite network. The upper three are X-nodes, and the lower four are Y-nodes. The whole process consists of two steps: First, the resource flows from X to Y (a→b), and then returns to X (b→c). Different from the prior network-based resource-allocation dynamics [40], the resource here can only flow from one node-set to another node-set, without consideration of asymptotical stable flow among one node-set.

Now, consider a general bipartite network $G(X, Y, E)$, where E is the set of edges. The nodes in X and Y are denoted by $x_1, x_2, \cdots, x_n$ and $y_1, y_2, \cdots, y_m$, respectively. The initial resource located on the ith X-node is $f(x_i) \geq 0$. After the first step, all the resource in X flows to Y, and the resource located on the lth Y-node reads,

$$f(y_l) = \sum_{i=1}^{n} \frac{a_{il} f(x_i)}{k(x_i)}, \qquad (2)$$

where $k(x_i)$ is the degree of x_i, and a_{il} is an $n \times m$ adjacent matrix as

$$a_{il} = \begin{cases} 1, & x_i y_l \in E, \\ 0, & \texttt{otherwise}. \end{cases} \qquad (3)$$

In the next step, all the resource flows back to X, and the final resource located on x_i reads,

$$f'(x_i) = \sum_{l=1}^{m} a_{il} f(y_l)/k(y_l) = \sum_{l=1}^{m} \frac{a_{il}}{k(y_l)} \sum_{j=1}^{n} \frac{a_{jl} f(x_j)}{k(x_j)}. \qquad (4)$$

This can be rewritten as

$$f'(x_i) = \sum_{j=1}^{n} w_{ij} f(x_j), \qquad (5)$$

where

$$w_{ij} = \frac{1}{k(x_j)} \sum_{l=1}^{m} \frac{a_{il} a_{jl}}{k(y_l)}, \qquad (6)$$

which sums the contribution from all 2-step paths between x_i and x_j. The matrix $W = \{w_{ij}\}_{n \times n}$ represents the weighted X-projection we were looking for. The resource-allocation process can be written in the matrix form as $\overrightarrow{f'} = W \overrightarrow{f}$.

It is worthwhile to emphasize the particular characters of this weighting method. For convenience, we take the scientific collaboration network as an example, but our statements are not restricted to the collaboration networks. Firstly, the weighted matrix is not symmetrical as

$$\frac{w_{ij}}{k(x_j)} = \frac{w_{ji}}{k(x_i)}. \qquad (7)$$

This is in accordance with our daily experience - the weight of a single collaboration paper is relatively small if the scientist has already published many papers (i.e., he has large degree), vice versa. Secondly, the diagonal elements in W are nonzero, thus the information contained by the connections incident to one-degree Y-node will not be lost. Actually, the diagonal element is the maximal element in each column. Only if all x_i's Y-neighbors belongs to x_j's neighbors set, $w_{ii} = w_{ji}$. It is usually found in scientific collaboration networks, since some students coauthorize every paper with their supervisors. Therefore, the ratio $w_{ji}/w_{ii} \leq 1$ can be considered as x_i's researching independence to x_j, the smaller the ratio, the more independent the researcher is, vice versa. The independence of x_i can be approximately measured as

$$I_i = \sum_{j} \left(\frac{w_{ji}}{w_{ii}} \right)^2. \qquad (8)$$

Generally, the author who often publishes papers solely, or often publishes many papers with different coauthors is more independent. Note that, introducing the measure I_i here is just to show an example how to use the information contained by self-weight w_{ii}, without any comments whether to be more independent is better, or contrary.

III. PERSONAL RECOMMENDATION

The exponential growth of the Internet [41] and World-Wide-Web [42] confronts people with an information overload: They are facing too many data and sources to be able to find out those most relevant for him. One landmark for information filtering is the use of search engines [43], however, it can not solve this *overload problem* since it does not take into account of personalization thus returns the same results for people with far different habits. So, if user's habits are different from the mainstream, it is hard for him to find out what he likes in the
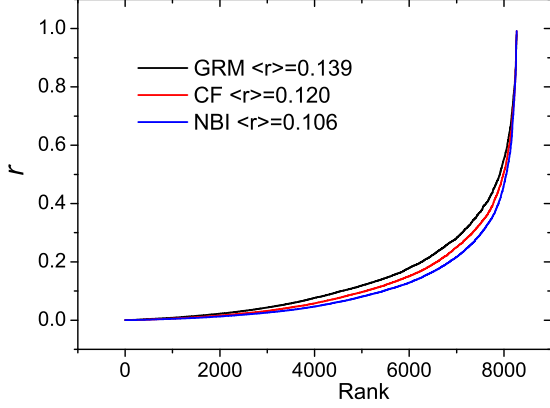
FIG. 3: (color online) The predicted position of each entry in the probe ranked in the ascending order. The black, red and blue curves, from top to bottom, represent the cases of GRM, CF and NBI, respectively. The mean values are top 13.9% (GRM), top 12.0% (CF) and top 10.6% (NBI).

countless searching results. Thus far, the most potential way to efficiently filter out the information overload is to recommend personally. That is to say, using the personal information of a user (i.e., the historical track of this user's activities) to uncover his habits and to consider them in the recommendation. For instances, Amazon.com uses one's purchase history to provide individual suggestions. If you have bought a textbook on statistical physics, Amazon may recommend you some other statistical physics books. Based on the well-developed *Web 2.0* technology [44], the recommendation systems are frequently used in web-based movie-sharing (music-sharing, book-sharing, etc.) systems, web-based selling systems, bookmark web-sites, and so on. Motivated by the significance in economy and society, recently, the design of an efficient recommendation algorithm becomes a joint focus from marketing practice [45, 46] to mathematical analysis [47], from engineering science [48, 49, 50] to physics community [51, 52, 53].

Basically, a recommendation system consists of users and objects, and each user has collected some objects. Denote the object-set as $O = \{o_1, o_2, \cdots, o_n\}$ and user-set as $U = \{u_1, u_2, \cdots, u_m\}$. If users are only allowed to collect objects (they do not rate them), the recommendation system can be fully described by an $n \times m$ adjacent matrix $\{a_{ij}\}$, where $a_{ij} = 1$ if u_j has already collected o_i, and $a_{ij} = 0$ otherwise. A reasonable assumption is that the objects you have collected are what you like, and a recommendation algorithm aims at predicting your personal opinions (to what extent you like or hate them) on those objects you have not yet collected. A more complicated case is the voting system [54, 55], where each user can give ratings to objects (e.g., in the *Yahoo Music*, the users can vote each song with 5 discrete ratings representing *Never play again, It is ok, Like it, Love it*, and

Can't get enough), and the recommendation algorithm concentrates on estimating unknown ratings for objects. These two problems are closely related, however, in this article, we focus on the former case.

Denote $k(o_i) = \sum_{j=1}^{m} a_{ij}$ the degree of object o_i. The *global ranking method* (GRM) sorts all the objects in the descending order of degree and recommends those with highest degrees. Although the lack of personalization leads to an unsatisfying performance of GRM (see numerical comparison in the next section), it is widely used since it is simple and spares computational resources. For example, the well-known *Yahoo Top 100 MTVs*, *Amazon List of Top Sellers*, as well as the board of most downloaded articles in many scientific journals, can be all considered as results of GRM.

Thus far, the widest applied personal recommendation algorithm is *collaborative filtering* (CF) [50, 54], based on a similarity measure between users. Consequently, the prediction for a particular user is made mainly using the similar users. The similarity between users u_i and u_j can be measured in the Pearson-like form

$$s_{ij} = \frac{\sum_{l=1}^{n} a_{li}a_{lj}}{\min\{k(u_i), k(u_j)\}}, \qquad (9)$$

where $k(u_i) = \sum_{l=1}^{n} a_{li}$ is the degree of user u_i. For any user-object pair $u_i - o_j$, if u_i has not yet collected o_j (i.e., $a_{ji} = 0$), by CF, the predicted score, v_{ij} (to what extent u_i likes o_j), is given as

$$v_{ij} = \frac{\sum_{l=1, l \neq i}^{m} s_{li}a_{jl}}{\sum_{l=1, l \neq i}^{m} s_{li}}. \qquad (10)$$

Two factors give rise to a high value of v_{ij}. Firstly, if the degree of o_j is larger, it will, generally, have more nonzero items in the numerator of Eq. (10). Secondly, if o_j is frequently collected by users very similar to u_i, the corresponding items will be significant. The former pays respect to the global information, and the latter reflects the personalization. For any user u_i, all the nonzero v_{ij} with $a_{ji} = 0$ are sorted in descending order, and those objects in the top are recommended.

We propose a recommendation algorithm, which is a direct application of the weighting method for bipartite networks presented above. The layout is simple: first compress the bipartite user-object network by object-projection, the resulting weighted network we label G. Then, for a given user u_i, put some resource on those objects already been collected by u_i. For simplicity, we set the initial resource located on each node of G as

$$f(o_j) = a_{ji}. \qquad (11)$$

That is to say, if the object o_j has been collected by u_i, then its initial resource is unit, otherwise it is zero. Note that, the initial configuration, which captures personal preferences, is different for different users. The initial resource can be understood as giving a unit recommending capacity to each collected object. According to

the weighted resource-allocation process discussed in the prior section, the final resource, denoted by the vector $\overrightarrow{f'}$, is $\overrightarrow{f'} = W \overrightarrow{f}$. Thus components of f' are

$$f'(o_j) = \sum_{l=1}^{n} w_{jl} f(o_l) = \sum_{l=1}^{n} w_{jl} a_{li}. \qquad (12)$$

For any user u_i, all his uncollected objects o_j ($1 \leq j \leq n$, $a_{ji} = 0$) are sorted in the descending order of $f'(o_j)$, and those objects with highest value of final resource are recommended. We call this method *network-based inference* (NBI), since it is based on the weighted network G. Note that, the calculation of Eq. (12) should be repeated m times, since the initial configurations are different for different users.

IV. NUMERICAL RESULTS

We use a benchmark data-set, namely *MovieLens*, to judge the performance of described algorithms. The MovieLens data is downloaded from the web-site of *GroupLens Research* (http://www.grouplens.org). The data consists 1682 movies (objects) and 943 users. Actually, MovieLens is a rating system, where each user votes movies in five discrete ratings 1-5. Hence we applied the coarse-graining method similar to what is used in Ref. [19]: A movie has been collected by a user iff the giving rating is at least 3. The original data contains 10^5 ratings, 85.25% of which are ≥ 3, thus the user-movie bipartite network after the coarse gaining contains 85250 edges. To test the recommendation algorithms, the data set (i.e., 85250 edges) is randomly divided into two parts: The training set contains 90% of the data, and the remaining 10% of data constitutes the probe. The training set is treated as known information, while no information in probe set is allowed to be used for prediction.

All three algorithms, GRM, CF and NBI, can provide each user an ordered queue of all its uncollected movies. For an arbitrary user u_i, if the edge $u_i - o_j$ is in the probe set (according to the training set, o_j is an uncollected movie for u_i), we measure the position of o_j in the ordered queue. For example, if there are 1500 uncollected movies for u_i, and o_j is the 30th from the top, we say the position of o_j is the top 30/1500, denoted by $r_{ij} = 0.02$. Since the probe entries are actually collected by users, a good algorithm is expected to give high recommendations to them, thus leading to small r. The mean value of the position value, averaged over entries in the probe, are 0.139, 0.120 and 0.106 by GRM, CF and NBI, respectively. Fig. 3 reports the distribution of all the position values, which are ranked from the top position ($r \to 0$) to the bottom position ($r \to 1$). Clearly, NBI is the best method and GRM performs worst.

To make this work more relevant to the real-life recommendation systems, we introduce a measure of algorithmic accuracy that depends on the length of recommendation list. The recommendation list for a user u_i, if of
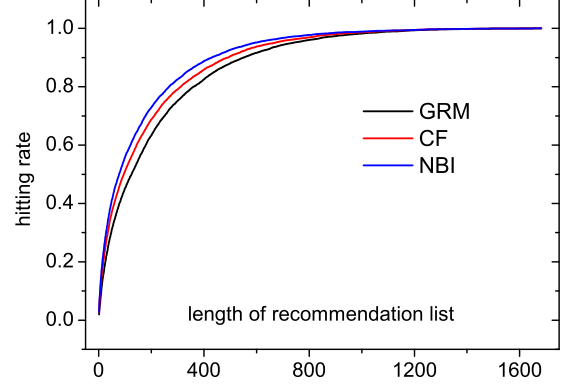

FIG. 4: The hitting rate as a function of the length of recommendation list. The black, red and blue curves, from bottom to top, represent the cases of GRM, CF and NBI, respectively.

TABLE I: The hitting rates for some typical lengths of recommendation list.

Length	GRM	CF	NBI
10	10.3%	14.1%	16.2%
20	16.9%	21.6%	24.8%
50	31.1%	37.0%	41.2%
100	45.2%	51.0%	55.9%

length L, contains L highest recommended movies resulting from the algorithm. For each incident entry $u_i - o_j$ in the probe, if o_j is in u_i's recommendation list, we say the entry $u_i - o_j$ is *hit* by the algorithm. The ratio of hit entries to the population is named *hitting rate*. For a given L, the algorithm with a higher hitting rate is better, and vice versa. If L is larger than the total number of uncollected movies for a user, the recommendation list is defined as the set of all his uncollected movies. Clearly, the hitting rate is monotonously increasing with L, with the upper bound 1 for sufficiently large L. In Fig. 4, we report the hitting rate as a function of L for different algorithms. In accordance with Fig. 3, the accuracy of the algorithms is NBI > CF > GRM. The hitting rates for some typical lengths of recommendation list are shown in Table I.

In a word, via the numerical calculation on a benchmark data set, we have demonstrated that the NBI has remarkably better performance than GRM and CF, which strongly guarantee the validity of the present weighting method.

V. CONCLUSION AND DISCUSSION

Weighting of edges is the key problem in the construction of a bipartite network projection. In this article

we proposed a weighting method based on a resource-allocation process. The present method has two prominent features. First, the weighted matrix is not symmetrical, and the node having larger degree in the bipartite network generally assigns smaller weights to its incident edges. Second, the diagonal element in the weighted matrix is positive, which makes the weighted one-mode projection more informative.

Furthermore, we proposed a personal recommendation algorithm based on this weighting method, which performs much better than the widest used global ranking method as well as the collaborative filtering. Especially, this algorithm is tune-free (i.e., does not depend on any control parameters), which is a big advantage for potential users. The main goal of this article is to raise a new weighting method, as well as provide a bridge from this method to the recommendation systems. The presented recommendation algorithm is just a rough framework, whose details have not been exhaustively explored yet. For example, the setting of the initial configuration may be oversimplified, a more complicated form, like $f(o_j) = a_{ji}k^{\beta}(o_j)$, may lead to a better performance than the presented one with $\beta = 0$. One is also encouraged to consider the asymptotical dynamics of the resource-allocation process [40], which can eventually lead to some certain iterative recommendation algorithms. Although such an algorithm require much longer CPU time, it may give more accurate prediction than the present algorithm.

If we denote $\langle k_u \rangle$ and $\langle k_o \rangle$ the average degree of users and objects in the bipartite network, the computational complexity of CF is $\mathbb{O}(m^2\langle k_u \rangle + mn\langle k_o \rangle)$, where the first term accounts for the calculation of similarity between users (see Eq. (9)), and the second term accounts for the calculation of the predicted score (see Eq. (10)). Substituting the equation $n\langle k_o \rangle = m\langle k_u \rangle$, we are left with $\mathbb{O}(m^2\langle k_u \rangle)$. The computational complexity for NBI is $\mathbb{O}(m\langle k_u^2 \rangle + mn\langle k_u \rangle)$ with two terms accounting for the calculation of the weighted matrix and the final resource distribution, respectively. Here $\langle k_u^2 \rangle$ is the second moment of the users' degree distribution in the bipartite network. Clearly, $\langle k_u^2 \rangle < n\langle k_u \rangle$, thus the resulting form is $\mathbb{O}(mn\langle k_u \rangle)$. Note that the number of users is usually much larger than the number of objects in many recommendation systems. For instance, the *EachMovie* dataset provided by the *Compaq* company contains $m = 72916$ users and $n = 1628$ movies, and the *Netflix* company provides nearly 20 thousands online movies for million users. It is also the case of music-sharing systems and online bookstores, the number of registered users is more than one magnitude larger than that of the available objects (e.g., music groups, books, etc.). Therefore, NBI runs much fast than CF. In addition, NBI requires n^2 memory to store the weighted matrix $\{w_{ij}\}$, while CF requires m^2 memory to store the similarity matrix $\{s_{ij}\}$. Hence, NBI is able to beat CF in all the three criterions of recommendation algorithm: *accuracy*, *time* and *space*. However, in some recommendation systems, as in bookmark sharing websites, the number of objects (e.g. webpages) is much larger than the number of users, thus CF may be more practicable.

[1] L. A. N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley, Proc. Natl. Acad. Sci. U.S.A. **97**, 11149 (2000).

[2] S. H. Strogatz, Nature **410**, 268 (2001).

[3] R. Albert and A. -L. Barabási, Rev. Mod. Phys. **74**, 47 (2002).

[4] S. N. Dorogovtsev and J. F. F. Mendes, Adv. Phys. **51**, 1079 (2002).

[5] M. E. J. Newman, SIAM Review **45**, 167 (2003).

[6] S. Boccaletti, *et al.*, Phys. Rep. **424**, 175 (2006).

[7] L. da F. Costa, *et al.*, Adv. Phys. **56**, 167 (2007).

[8] P. Holme, F. Liljeros, C. R. Edling, and B. J. Kim, Phys. Rev. E **68**, 056107.

[9] F. Liljeros, *et al.*, Nature **411**, 907 (2001).

[10] H. Jeong, *et al.*, Nature 407, 651 (2000).

[11] S. Wasserman, and K. Faust, *Social Network Analysis* (Cambridge Univ. Press, Cambridge, 1994).

[12] J. Scott, *Social Network Analysis* (Sage Publication, London, 2000).

[13] M. E. J. Newman, Proc. Natl. Acad. Sci. U.S.A. **98**, 404 (2001).

[14] M. E. J. Newman, Phys. Rev. E **64**, 016131 (2001).

[15] D. J. Watts and S. H. Strogatz, Nature **393**, 440 (1998).

[16] C. R. Myers, Phys. Rev. E **68**, 046116 (2003).

[17] P. -P. Zhang, *et al.*, Physica A **360**, 599 (2006).

[18] S. Maslov, and Y. -C. Zhang, Phys. Rev. Lett. **87**, 248701 (2001).

[19] M. Blattner, Y. -C. Zhang, and S. Maslov, Physica A **373**, 753 (2007).

[20] R. Lambiotte, and M. Ausloos, Phys. Rev. E **72**, 066107 (2005).

[21] P. Cano, O. Celma, M. Koppenberger, and J. M. Buldu, Chaos **16**, 013107 (2006).

[22] C. Cattuto, V. Loreto, and L. Pietronero, Proc. Natl. Acad. Sci. U.S.A. **104**, 1461 (2007).

[23] G. Linden, B. Smith, and J. York, IEEE Internet Computing **7**, 76 (2003).

[24] K. Yammine, *et al.*, Lect. Notes. Comput. Sci. **3220**, 720 (2004).

[25] R. Lambiotte, and M. Ausloos, Phys. Rev. E **72**, 066117 (2005).

[26] P. G. Lind, M. C. González, and H. J. Herrmann, Phys. Rev. E **72**, 056127 (2005).

[27] E. Estrada, and J. A. Rodríguez-Velázquez, Phys. Rev. E **72**, 046105 (2005).

[28] J. J. Ramasco, S. N. Dorogovtsev, and R. Pastor-Satorras, Phys. Rev. E **70**, 036106 (2004).

[29] J. Ohkubo, K. Tanaka, and T. Horiguchi, Phys. Rev. E **72**, 036120 (2005).

[30] M. Peltomäki, and M. Alava, J. Stat. Mech. P01010 (2006).

[31] J. W. Grossman, and P. D. F. Ion, Congressus Numerantium **108**, 129 (1995).

[32] A. -L. Barabási, *et al.*, Physica A **311**, 590 (2002).

[33] T. Zhou, *et al.*, Int. J. Mod. Phys. C **18**, 297 (2007).

[34] J. J. Ramasco, and S. A. Morris, Phys. Rev. E **73**, 016122 (2006).

[35] M. Li, *et al.*, Physica A **375**, 355 (2007).

[36] M. E. J. Newman, Phys. Rev. E **70**, 056131 (2004).

[37] M. Li, *et al.*, Physica A **350**, 643 (2005).

[38] M. E. J. Newman, Phys. Rev. E **64**, 016132 (2001).

[39] M. E. J. Newman, Proc. Natl. Acad. Sci. U.S.A. **101**, 5200 (2004).

[40] Q. Ou, *et al.*, Phys. Rev. E **75**, 021102 (2007).

[41] M. Faloutsos, P. Faloutsos, and C. Faloutsos, Comput. Comm. Rev. **29**, 251 (1999).

[42] A. Broder, *et al.*, Comput. Netw. **33**, 309 (2000).

[43] J. M. Kleinberg, J. ACM **46**, 604 (1999).

[44] B. Alexander, Educause Rev. **41**, 33 (2006).

[45] A. Ansari, S. Essegaier, and R. Kohli, J. Marketing Research **37**, 363 (2000).

[46] Y. P. Ying, F. Feinberg, and M. Wedel, J. Marketing Research **43**, 355 (2006).

[47] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, J. Comput. Syst. Sci. **63**, 42 (2001).

[48] N. J. Belkin, Comm. ACM **43**, 58 (2000).

[49] M. Montaner, B. López, and J. L. De La Rosa, Artificial Intelligence Review **19**, 285 (2003).

[50] J. L. Herlocker, J. A. Konstan, K. Terveen, and J. T. Riedl, ACM Trans. Inform. Syst. **22**, 5 (2004).

[51] P. Laureti, L. Moret, Y. -C. Zhang, and Y. -K. Yu, Europhys. Lett. **75**, 1006 (2006).

[52] Y. -K. Yu, Y. -C. Zhang, P. Laureti, and L.Moret, arXiv: cond-mat/0603620.

[53] F. E. Walter, S. Battiston, and F. Schweitzer, arXiv: nlin/0611054.

[54] J. A. Konstan, *et al.*, Commun. ACM **40**, 77 (1997).

[55] K. Glodberg, T. Roeder, D. Gupta, and C. Perkins, Information Retrieval **4**, 133 (2001).