International Journal of Modern Physics C
Vol. 21, No. 10 (2010) 1217–1227

DOI: 10.1142/S0129183110015786

CAN DISSIMILAR USERS CONTRIBUTE TO ACCURACY AND DIVERSITY OF PERSONALIZED RECOMMENDATION?

WEI ZENG*, MING-SHENG SHANG† and QIAN-MING ZHANG‡
Web Sciences Center, School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu 610054, P. R. China
**zwei504@gmail.com*
†*msshang@uestc.edu.cn*
‡*QM-Zhang@hotmail.com*

LINYUAN LÜ§
Department of Physics, University of Fribourg
Chemin du Musée 3, Fribourg CH-1700, Switzerland
linyuan.lue@unifr.ch

TAO ZHOU
Web Sciences Center, School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu 610054, P. R. China
Department of Modern Physics and Nonlinear Science Center
University of Science and Technology of China
Hefei 230026, P. R. China
zhutou@ustc.edu

Received 1 August 2010
Accepted 1 September 2010

Recommender systems are becoming a popular and important set of personalization techniques that assist individual users with navigating through the rapidly growing amount of information. A good recommender system should be able to not only find out the objects preferred by users, but also help users in discovering their personalized tastes. The former corresponds to high accuracy of the recommendation, while the latter to high diversity. A big challenge is to design an algorithm that provides both highly accurate and diverse recommendation. Traditional recommendation algorithms only take into account the contributions of similar users, thus, they tend to recommend popular items for users ignoring the diversity of recommendations. In this paper, we propose a recommendation algorithm by considering both the effects of similar and dissimilar users under the framework of collaborative filtering. Extensive analyses on three datasets, namely MovieLens, Netflix and Amazon, show that our method performs much better than the standard collaborative filtering algorithm for both accuracy and diversity.

§Corresponding author.

Keywords: Recommender systems; collaborative filtering; diversity; accuracy.

PACS Nos.: 89.20.Ff, 89.75.Hc, 89.65.-s.

1. Introduction

It is more and more difficult to efficiently find out the relevant items among the huge amount of available information. A possible way to solve this problem is search engines,[1–3] which return the relevant items matching the required tags input by the user. It requires the users to specify in advance what they are looking for and the results cannot satisfy the personalized interests. Another way is recommender systems,[4] which attempt to anticipate the future likes or interests based on user's historical activities. The big difference between search engine and recommender system is that the former finds what you are looking for, while the latter helps you to discover what you might like.

Up to now, many recommendation algorithms have been proposed, such as collaborative filtering,[4,5] content-based analysis,[6] spectral analysis,[7] latent semantic models,[8] heat conduction,[9] opinion diffusion,[10,11] tag-based filtering,[12] and so on. Therein one of the most successful algorithms is *Collaborative Filtering* (CF), which has been extensively studied and has already found applications in e-commerce.[13] This algorithm can be further divided into user-based CF[14] and item-based CF.[15,16] The basic assumption of user-based CF is that people who agree in the past tend to agree again in the future. Thus, for a target user, the potential evaluation on an object is estimated according to the ratings from his similar users. Different with user-based CF, the item-based CF algorithm recommends a user the objects that are similar to what he/she has collected before.

The standard CF algorithm takes only the effect of similar users into account. The risk of such approach is that, with recommendation based on overlap rather than difference, more and more users will be exposed to a narrowing band of popular objects, while the niche items that might be very relevant will be overlooked. As a result, the popular objects will be overestimated and the recommended objects will become more and more similar to each other. In a word, the recommendations drawn by traditional methods may be accurate but not diverse — this is the so-called diversity-accuracy dilemma of recommender systems.[17] To solve this problem, based on the framework of user-based CF, we propose an algorithm considering the effects of not only similar users, but also dissimilar users. Three datasets, MovieLens, Netflix and Amazon, are used to test the algorithm's performance. The results show that our method results in a considerable improvement for both accuracy and diversity.

2. Method

2.1. *Standard collaborative filtering*

A rating system can be represented by a bipartite network $G(U, O, E)$, where $U = \{u_1, u_2, \ldots, u_m\}$, $O = \{o_1, o_2, \ldots, o_n\}$ and $E = \{e_1, e_2, \ldots, e_l\}$ are the sets of users,

objects and links.[18] Denote by $A_{m\times n}$ the adjacency matrix, where the element $a_{i\alpha}$ equals 1 if user $i \in U$ has collected object $\alpha \in O$, and 0 otherwise.

The basic assumption of collaborative filtering algorithm is that people who agreed in the past tend to agree again in the future. Based on this assumption, for a target user $i \in U$, we first give a score for each uncollected object α, and those objects with the highest score will be recommended to this user. The score is estimated in the following way

$$p_{i,\alpha} = \sum_{j=1}^{m} s_{ij} \cdot a_{j\alpha} \,, \tag{1}$$

where s_{ij} is the similarity between user i and j, and $a_{i\alpha}$ is the element of adjacency matrix A.

In the framework of user-based collaborative filtering, the most important thing is to properly quantify the similarity between users. In this paper, we apply two standard indices. One is *cosine similarity*,[19] defined as:

$$s_{ij} = \frac{|\Gamma_i \cap \Gamma_j|}{\sqrt{k_i \cdot k_j}} \,, \tag{2}$$

where Γ_i denotes the set of objects that user i has collected, and $k_i = |\Gamma_i|$ is the degree of user i, namely the number of objects that user i has collected. The other is *Jaccard similarity*,[20] defined as:

$$s_{ij} = \frac{|\Gamma_i \cap \Gamma_j|}{|\Gamma_i \cup \Gamma_j|} \,. \tag{3}$$

2.2. *Collaborative filtering with both similar and dissimilar users*

The standard CF algorithm (SCF for short) considers only the effects of similar users of the target user. This will lead to an unwilling fact that all similar users will become more and more similar to each other, deviating from the soul of personalized recommendation (personalized tastes should be somehow related to diverse recommendations). Furthermore, the SCF algorithm tends to overestimate the popular objects, thus it has low ability to find the niche or unpopular objects that may be liked by the users.

Taking Fig. 1 for example, we want to recommend an object to u_3. For simplicity, we only compare o_4 and o_5. It is clear that u_3 and u_4 are similar to each other since

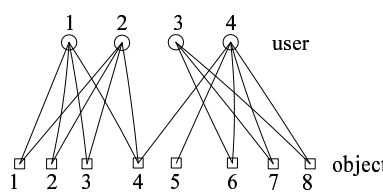


Fig. 1. A user-movie bipartite network. Circles and squares denote users and movies respectively. The solid lines denote the users' history collections.

most of their collections are the same. In addition, u_1 and u_2 have different tastes with u_3, and thus both u_1 and u_2 can be considered as dissimilar users to u_3. Because both o_4 and o_5 were collected by u_4, and according to SCF algorithm which consider only the effect of u_3's similar neighbor u_4, these two objects have the same probability to be recommended to u_3. However, o_4 is a popular object that has been collected also by u_1 and u_2 who are dissimilar to u_3, while o_5 is only selected by u_4. Therefore, if considering the negative effects of u_3's dissimilar neighbors u_1 and u_2, o_4 may be less suitable to u_3 than o_5. Aiming at solving these problems and achieving a better recommendation, we propose a method considering both the effects of similar and dissimilar users under the framework of collaborative filtering.

Given a target user, firstly we need to compute the similarity scores and dissimilarity scores between the target user and other users. We adopt the simplest method, *common neighbors*, to measure the similarity between two users, (despite its simplicity, the *common neighbors* index is shown to be very effective to quantify node similarities[21–23]) namely, two users are considered to be more similar if they have collected more common objects. The definition can be written as

$$s_{ij} = |\Gamma_i \cap \Gamma_j| \,. \tag{4}$$

Accordingly, the dissimilarity can be defined as the number of different objects that two users have collected, namely

$$d_{ij} = |\Gamma_i \cup \Gamma_j| - |\Gamma_i \cap \Gamma_j| \,. \tag{5}$$

We normalize both s_{ij} and d_{ij} in the following way:

$$\widehat{s_{ij}} = \frac{s_{ij}}{\sum_{k=1}^{m} s_{ik}} \,, \tag{6}$$

$$\widehat{d_{ij}} = \frac{d_{ij}}{\sum_{k=1}^{m} d_{ik}} \,. \tag{7}$$

According to Eq. (1), with $\widehat{s_{ij}}$ and $\widehat{d_{ij}}$, we have the positive recommendation score from similar users, p^+, and negative recommendation score, p^- from dissimilar users, as:

$$p^+_{i,\alpha} = \sum_{j=1}^{m} \widehat{s_{ij}} a_{j\alpha} \,, \tag{8}$$

and

$$p^-_{i,\alpha} = \sum_{j=1}^{m} \widehat{d_{ij}} a_{j\alpha} \,. \tag{9}$$

The final score $p_{i,j}$ is integrated in a linear way

$$p_{i,j} = p^+_{i,j} + \lambda \cdot p^-_{i,j} \,, \tag{10}$$

where λ is a free parameter. When $\lambda = 0$, our method will degenerate to SCF algorithm.

Table 1. Basic statistics of the tested datasets. The sparsity is calculated by $|E|/(|U|^*|O|$, where $|E|$ is the number of links in the bipartite network, $|U|$ and $|O|$ denote the number of users and objects respectively.

Dataset	Users	Objects	Links	Sparsity
MovieLens	943	1682	100 000	6.3×10^{-2}
Netflix	3000	2779	197 248	2.4×10^{-2}
Amazon	3607	4000	134 680	9.3×10^{-3}

3. Experiment

3.1. *Datasets*

Three datasets are used to test the algorithms:

(i) MovieLens (http://www.movieLens.org) is a movie recommendation website, which uses users' ratings to generate personalized recommendations.
(ii) Netflix (http://www.netflix.com) is an online DVD and Blu-ray Disc rental service in the US. The data we used is a random sample that consists of 3000 users who have voted at least 45 movies and 2779 movies having been voted by at least 23 users.
(iii) Amazon (http://www.amazon.com) is a multinational electronic commerce company. The original data were collected from 28 July 2005 to 27 September 2005, and the data we used is a random sample.

The basic statistics of these three datasets are shown in Table 1.

3.2. *Metrics*

To test the algorithm's performance, the observed links are randomly divided into two parts: the training set and the probe set. In this paper, we randomly remove 20% links to the probe set, and the remaining constitutes the training set.

We employ three metrics to measure the algorithm's performance: *Precision* and *Ranking Score* to measure accuracy in recovery of deleted links[24] and *Hamming distance* to measure recommendation diversity.[25] A short introduction of these three metrics is shown as follows:

Precision — This metric considers only the top-L objects of the recommendation list. For a target user i, the precision of recommendation, $P_i(L)$, is defined as

$$P_i(L) = \frac{R_i(L)}{L}\,, \tag{11}$$

where $R_i(L)$ indicates the number of relevant objects, namely the objects collected by u_i in the probe set (among the L recommended objects). Averaging over all the

individual precisions, we obtain the precision of the whole system, as

$$P(L) = \frac{1}{m} \sum_{i=1}^{m} P_i(L) \,, \tag{12}$$

where m is the number of users in the system. Clearly, higher precision means higher recommendation accuracy.

Ranking Score — Each recommendation algorithm can provide an active user an ordered list of its uncollected items. For an active user u, if the rating on item α, namely $p_{u,\alpha}$, is in the probe set (according to the training set the item α is an uncollected item for user u), we measure the position of this item α in the order list. For example, if there are 1000 uncollected items for u and α is the 30th from the top in the order list, we say the position of α is the top 30/1000, and thus the ranking score $RS_{u,\alpha} = 0.03$. A good algorithm is expected to give a smaller ranking score.

Diversity — It considers the uniqueness of different user's recommendation list. Given two users i and j, the difference between their recommendation lists can be measured by the Hamming distance,

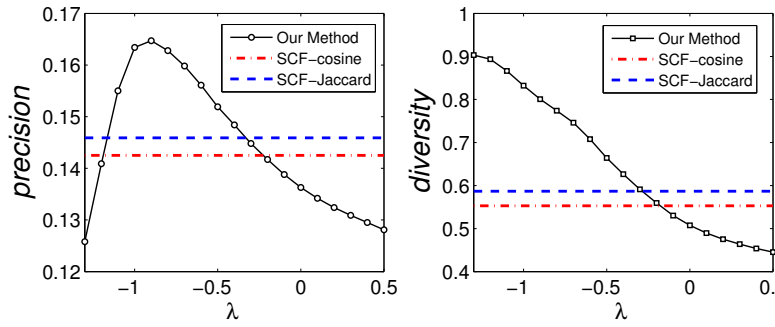
$$H_{ij}(L) = 1 - \frac{N_{ij}(L)}{L} \,, \tag{13}$$

where $N_{ij}(L)$ is the number of common objects in the top-L places of both lists. Clearly, if user i and j have the same list, $H_{ij}(L) = 0$, while if their lists are completely different, $H_{ij}(L) = 1$. Averaging $H_{ij}(L)$ over all pairs of users we obtain the mean distance $H(L)$, for which greater or lesser values mean, respectively, greater or lesser personalization of users' recommendation lists.

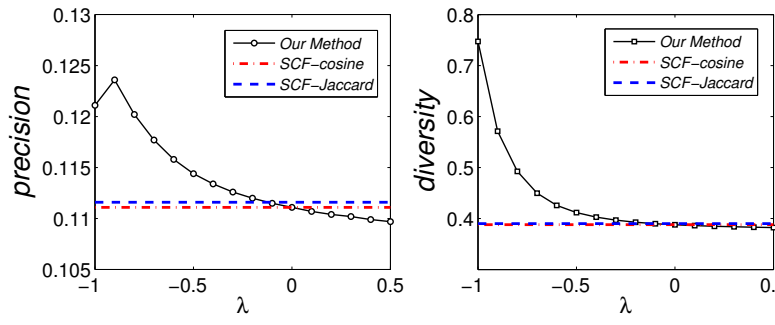
3.3. *Results*

The dependence of precision and diversity of our method on parameter λ for three datasets are shown in Fig. 2, where the length of recommendation list is 50. The main findings for $L = 20$ and $L = 100$ are similar, and thus we did not show them here. In Fig. 2, the red dotted line and the blue dashed line indicate the performances of SCF with cosine and Jaccard similarity, respectively. From Fig. 2, one can see that with a suitable parameter λ our method performs much better than SCF for both *precision* and *diversity*. Subject to the *precision* the optimal parameters λ for MovieLens, Netflix and Amazon are -0.93, -0.95 and -1.79, respectively.

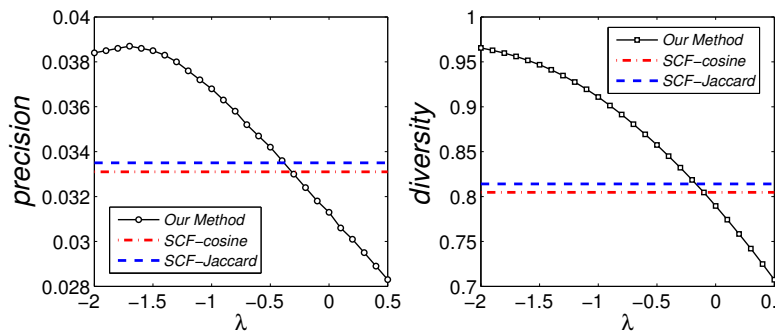
Table 2 shows the comparison of precision and diversity of SCF and the optimal case of our method. One can see that the Jaccard-based SCF performs slightly better than the cosine-based SCF, and comparing with the Jaccard-based SCF, our method achieves a considerable improvement for both precision and diversity. The comparison of Ranking Score of SCF and our method on three datasets is given in Table 3. Although the optimal value of λ leading to the lowest Ranking



(a)



(b)



(c)

Fig. 2. (Color online) Dependence of precision and diversity on parameter λ. Each number is obtained by averaging over 10 implementations with independently random divisions of the training set and probe set. (a) MovieLens, (b) Netflix, and (c) Amazon.

Score is different from that subject to the highest precision, there exists a certain negative λ corresponding to the most accurate recommendation with considerable improvement compared with SCF. For both precision and Ranking Score, the values of optimal λ are less than 0, which implies that the dissimilar users play a negative role in recommendation. This strongly supports our previous assumption that the

Table 2. Comparison of precision and diversity of SCF and our method on three datasets. For precision, we presented results of our method corresponding to the optimal values of precision and the values of optimal λ are given in brackets. For diversity, we give the values with optimal λ of precision.

	Precision	SCF-cosine	SCF-Jaccard	Our method*
MovieLens	$L = 20$	0.208	0.214	**0.245**(−0.88)
	$L = 50$	0.143	0.146	**0.165**(−0.93)
	$L = 100$	0.101	0.103	**0.114**(−0.91)
Netflix	$L = 20$	0.171	0.172	**0.202**(−0.95)
	$L = 50$	0.111	0.112	**0.124**(−0.95)
	$L = 100$	0.074	0.074	**0.080**(−0.9)
Amazon	$L = 20$	0.049	0.050	**0.059**(−1.94)
	$L = 50$	0.033	0.034	**0.039**(−1.79)
	$L = 100$	0.024	0.024	**0.027**(−1.57)
	Diversity	SCF-cosine	SCF-Jaccard	Our method*
MovieLens	$L = 20$	0.638	0.672	**0.849**(−0.88)
	$L = 50$	0.553	0.587	**0.809**(−0.93)
	$L = 100$	0.463	0.494	**0.752**(−0.91)
Netflix	$L = 20$	0.496	0.499	**0.732**(−0.95)
	$L = 50$	0.388	0.390	**0.642**(−0.95)
	$L = 100$	0.299	0.302	**0.534**(−0.9)
Amazon	$L = 20$	0.855	0.862	**0.975**(−1.94)
	$L = 50$	0.805	0.814	**0.959**(−1.79)
	$L = 100$	0.764	0.774	**0.936**(−1.57)

Table 3. Comparison of Ranking Score of SCF and our method on three datasets. The presented results of our method correspond to the optimal values of *rank score*, $\lambda^* = -0.76$, -0.65 and -0.75, for MovieLens, Netflix and Amazon respectively.

Rank Score	SCF-cosine	SCF-Jaccard	Our method*
MovieLens	0.110	0.105	**0.09**(−0.76)
Netflix	0.067	0.066	**0.062**(−0.65)
Amazon	0.129	0.128	**0.125**(−0.75)

quality of recommendations will be improved if taking into account the negative effects of dissimilar users.

As we have pointed out in Sec. 2, our method has higher ability to find the niche (unpopular) objects that may be liked by users, and thus give a more personalized recommendation to the target user. To give more evidences, we collect the top-L recommended objects for each user. Denote by n the number of distinct objects among all the selected objects. Then we rank the n objects according to their recommended times, denoting by Q_i $(i = 1, \ldots, n)$, in decreasing order. The

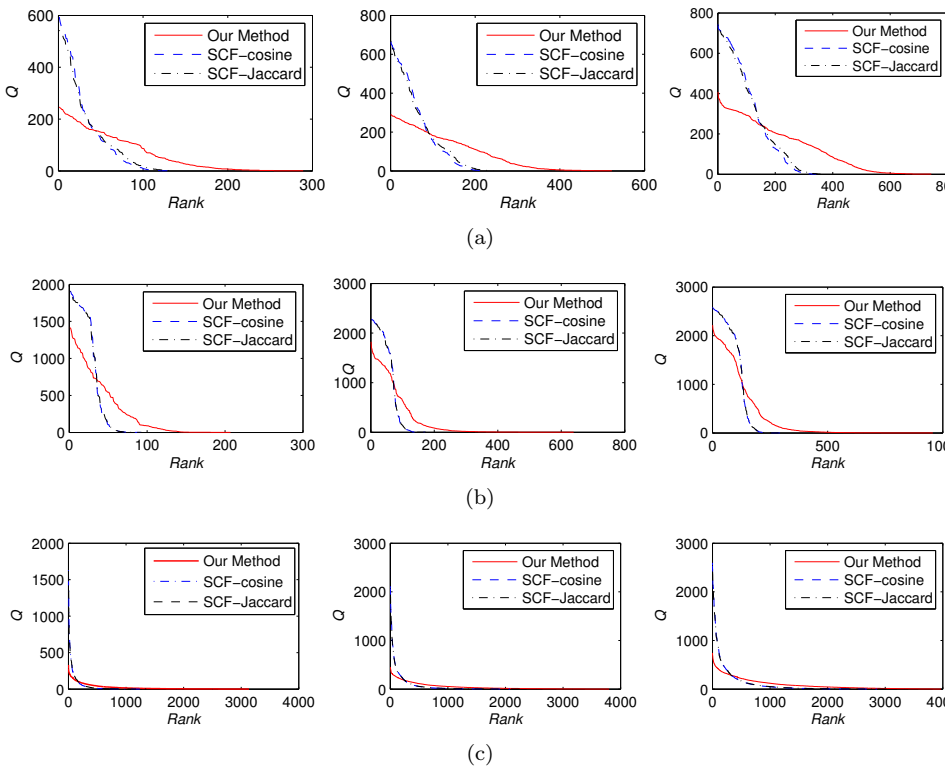


Fig. 3. (Color online) The relationship between the recommended times of objects and their ranks. From left to right, the length of recommendation lists are 20, 50 and 100 respectively. (a) MovieLens, (b) Netflix, and (c) Amazon.

relationships between the objects' recommended times and their ranks for the three datasets are shown in Fig. 3.

Two important phenomena can be obtained from Fig. 3. Firstly, our method provides much more distinct objects to the users. For example, when the length of recommendation list is 50, in Netflix, SCF can only recommend less than 200 objects, while our method increases this number to more than 500. In Amazon data, nearly 4000 objects have been recommended, namely almost every object has the chance to be recommended. This implies that our method gives a much wider horizon than SCF. Secondly, the curves for SCF are remarkably steeper than those from our method. Take the MovieLens data for example (the case $L = 50$), with SCF algorithm, some movies are recommended over six hundred times. Since there are only 943 users in this dataset, it means that this movie is recommended to more than two-thirds of users. However, with our method, more objects have probability to be recommended to the users, and thus the recommendation will be more personalized.

4. Conclusion

In this paper, we proposed a recommendation algorithm by considering both the effects of similar and dissimilar users under the framework of collaborative filtering. Three datasets were used to test the performance of the algorithm, namely MovieLens, Netflix and Amazon. The results showed that our method performs much better than the standard collaborative filtering algorithm subject to both accuracy (measured by precision and Ranking Score) and diversity. In other words, our method has higher ability to find not only the objects that the users like (high accuracy) but also the niche or unpopular objects that may be liked by users (high diversity).

Acknowledgment

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. (60973069, 90924011), and by the Sichuan Provincial Science and Technology Department (Grant No. 2010HH0002).

References

1. S. Brin and L. Page, *Comput. Netw. ISDN Syst.* **30**, 107 (1998).
2. J. Kleinberg, *J. ACM* **46**, 604 (1999).
3. N. J. Belkin, *Commun. ACM* **43**, 58 (2000).
4. G. Adomavicius and A. Tuzhilin, *IEEE Trans. Knowl. Data Eng.* **17**, 734 (2005).
5. J. B. Schafer, D. Frankowski, J. Herlocker and S. Sen, *Lect. Notes Comput. Sci.* **4321**, 291 (2007).
6. M. J. Pazzani and D. Billsus, *Lect. Notes Comput. Sci.* **4321**, 325 (2007).
7. K. Goldberg, T. Roeder, D. Gupta and C. Perkins, *Inform. Retrieval* **4**, 133 (2001).
8. T. Hofmann, *ACM Trans. Inf. Syst.* **22**, 89 (2004).
9. Y.-C. Zhang, M. Blattner and Y. K. Yu, *Phys. Rev. Lett.* **99**, 154301 (2007).
10. J. G. Liu, T. Zhou, B. H. Wang, Y.-C. Zhang and Q. Guo, *Int. J. Mod. Phys. C* **20**, 1925 (2009).
11. T. Zhou, J. Ren, M. Medo and Y.-C. Zhang, *Phys. Rev. E* **76**, 046115 (2007).
12. Z.-K. Zhang, T. Zhou and Y.-C. Zhang, *Physica A* **389**, 179 (2010).
13. Z. Huang, D. Zeng and H. C. Chen, *IEEE Intelligent Syst.* **22**, 68 (2007).
14. P. Resnick, N. Iacovou, M. Suchak, P. Bergström and J. Riedl, in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (New York, 1994), pp. 175.
15. M. Deshpande and G. Karypis, *ACM Trans. Inf. Syst.* **22**, 143 (2004).
16. G. Linden, B. Smith and J. York, *IEEE Internet Comput.* **7**, 76 (2003).
17. T. Zhou, Z. Kuscsik, J. G. Liu, M. Medo, J. R. Wakeling and Y.-C. Zhang, *Proc. Natl. Acad. Sci. USA* **107**, 4511 (2010).
18. M.-S. Shang, L. Lü, Y.-C. Zhang and T. Zhou, *EPL* **90**, 48006 (2010).
19. J. S. Breese, D. Heckerman and C. Kadie, in *Proceedings 14th Conference Uncertainty in Artificial Intelligence* (*UAI 98*) (1998), pp. 43.
20. P. Jaccard, *Bulletin de la Societe Vaudoise des Sciences Naturelles* **37**, 547 (1901).
21. D. Liben-Nowell and J. Kleinberg, *J. Am. Soc. Inf. Sci. Technol.* **58**, 1019 (2007).
22. T. Zhou, L. Lü and Y.-C. Zhang, *Eur. Phys. J. B* **71**, 623 (2009).

23. Q.-M. Zhang, M.-S. Shang and L. Lü, *Int. J. Mod. Phys. C* **21**, 813 (2010).
24. J. L. Herlocker, J. A. Konstan, L. G. Terveen and J. T. Riedl, *ACM Trans. Inf. Syst.* **22**, 5 (2004).
25. T. Zhou, L. L. Jiang, R. Q. Su and Y.-C. Zhang, *EPL* **81**, 58004 (2008).