# Reciprocal Recommenders

Luiz Pizzato, Tomek Rej, Thomas Chung,
Kalina Yacef, Irena Koprinska, and Judy Kay

School of Information Technologies
University of Sydney, NSW 2006, Australia
`{forename.surname}@sydney.edu.au`

**Abstract.** This paper introduces *Reciprocal Recommenders,* an important class of personalised recommender systems that has received little attention until now. The applications of Reciprocal Recommenders include online systems that help users to find a job, a mentor, a business partner or even a date. The contributions of this paper are the definition of this class of recommendation system, the identification of the particular personalisation challenges for them, the proposition of some promising techniques to address these challenges. We illustrate these concepts with a case study in online dating.

## 1 Introduction

There has been considerable research on recommenders and many deployed systems, in domains such as books (Amazon.com), pharmacy products (Drugstore.com), online auctions and seller reputation modelling (eBay) [11]. The dominant model for such recommenders is to provide a *user* with recommendations of *items* likely to be of interest to the user.

In social matching, where people are recommended to each other, the quality of a match is determined by all parties involved in the match. A match is successful only if everyone's preferences are satisfied. In terms of recommender systems, this motivates the concept of the *reciprocal recommender*, where the *user* and the *item* both have preferences that are considered when making a recommendation. This is in contrast with traditional recommenders which consider only the preferences of the user.

The traditional recommendation process is based on four important classes of models:

1. *explicit* user models or profile information about the user, such as their personal attributes like age, gender and educational level;
2. *explicit* models of the items, such as genre, director and actors for a movie recommender;
3. *explicit* user models of preferences in the domain, such as movie preferences in a movie recommender;
4. *implicit* user models based on their actions, for example purchases and time spent gaining more information about particular items.

**Table 1.** Difference between traditional recommenders and reciprocal recommenders

| Traditional recommender | Reciprocal recommender |
|---|---|
| User receives recommendations and is sole decider about their use/purchase. | User is aware that success depends on the agreement of the other party involved. |
| Items are typically abundant, and even if not, there is no need to limit the number of users recommended an *item*. | Items have very limited availability. Items are represented by other users, or, as in online dating, *item*s are other users. |
| A successful transaction is defined by the user who was given the recommendation. | A successful transaction is defined by both the user given the recommendations and the recommendation *item* itself. |
| Users and products might constantly re-occur in the system, making easier to track preferences. | Users and products might only occur once, and might never appear again after a successful transaction. Therefore, the cold-start problem is significant in this domain. |

Notably, in this large body of recommender research work, there is some symmetry between the explicit models of the *users* and the *items* (classes 1 and 2 above). However, there is no such symmetry for the explicit model of the user's preferences (Class 3) and the implicit model of preferences (Class 4).

In this paper, we will use the term *item* to refer to object being recommended, even when both the *user* and the *item* models may represent people. This allows us to compare the reciprocal recommender with conventional recommenders. For reciprocal recommender systems, we can make use of additional models:

5. *explicit* models of the *item*'s preferences in the domain;
6. *implicit* models of the *item*s based upon their activity.

Many domains will benefit from a reciprocal recommender. Consider the workflow of the expert recommender system in [10] where expert-seekers contact experts after receiving recommendations. The experts are passive in this system, and may only choose to reject a seeker after being contacted. It is clearly desirable to reduce the number of rejections from experts, as it costs time and effort for the seeker to browse for and contact experts. A reciprocal recommender would help by not only finding the right expert for the job, but also an expert who is likely to accept the job. In contrast, improving the immediate satisfaction of the users by providing them with people they like, might not reflect the final satisfaction of the user if these relationships are not mutual.

The reciprocal recommender is also useful in areas apart from expert recommendation. A job recommender needs to match the qualification of a candidate to the requirements of a position, but should also consider the likelihood of a candidate accepting a job [6]. A student/tutor recommendation needs to consider both the student and tutor's needs, skills and previous experiences [14]. For an online dating website, successful recommendations require that both users be interested in one another. Table 1 highlights some other differences between traditional and reciprocal recommenders.

Because reciprocal recommenders rely on a two-sided expression of interest between two different types of users, it presents some issues raised by Terveen and McDonald [13], such as privacy, trust, relation and interpersonal attraction. However, the authors have defined a research agenda centred on the computer human interaction issues for social matching, which may or may not not have a reciprocal facet. In this paper, we define the reciprocal recommendation which, although mostly consisting of interactions between people, does not necessarily imply social matching.

In Section 3, we present a definition of reciprocal recommendation which can be used to guide future research in the area. Section 2 presents a review of existing systems and techniques that are used in the areas that require reciprocal recommenders. Although a considerable amount of work has been done in related areas, reciprocal recommenders are still much underdeveloped. Section 4 highlights some approaches that can be taken to address the problem of recommendation in these domains. In Section 5 we present a case study using online dating as one domain that requires this type of recommendation. Section 6 presents the concluding remarks.

## 2 Related work

The field of recommender systems is well established, with a large volume of literature describing different techniques to predict those items that are appealing to the users. A common criteria used to distinguish between recommenders is the technique used to generate the recommendations itself (i.e. content-based techniques, collaborative-filtering, or hybrid techniques). Although much work has been done in the areas of recommender system that could benefit from reciprocation, very few highlight the need for reciprocity.

The work of Malinowski et al. [6] builds two recommender systems for an employment website: one recommender that finds the best jobs for a person seeking a job, and one recommender that suggests the best people for a certain job. Malinowski et al. point out that it is important to combine both approaches to match the interests of both job seekers and employers. The authors describe different ways of integrating both recommenders, highlighting the fact that a one-to-one (job-seeker to job) Pareto-optimal solution would be desired.[1] Addressing the computational cost of working with a large dataset, the authors also proposes a quicker solution that takes into account a single stakeholder for which the recommendations are maximized..

Another work that shows some level of reciprocity is described by Vassileva et al. [14] in the mentoring systems iHelp. iHelp uses a multi-agent architecture to facilitate the search for mentors by students. In iHelp, agents have a model of the knowledge of each user and when students ask for assistance, they are capable of finding other users who are willing to help and whose knowledge is comprehensive in the required topic of learning. The reciprocity of iHelp comes from the fact that the user model of the student and all possible mentors are analysed before a match is selected. The knowledge deficiencies of the student

[1] A Pareto-optimal solution in the employment website domain is a combination of matches between jobs and job seekers such that no single swap between a pair of job seekers improves or maintains the satisfaction of every single person involved.

are found and paired with the knowledge strengths of possible mentors, as well as with the willingness of users to become mentors.

In Brožovský and Petříček [2], collaborative filtering is used to predict the ratings that users will give to other users when presented with their photo. This task is not necessarily reciprocal as it is mostly used by users who want to know how other people rate their appearance. Despite this, Brožovský and Petříček discuss the need for reciprocal matching algorithms as finding that "A likes B" does not imply that "B likes A".

Although reciprocity is an important issue for intrinsically reciprocal tasks such as friend recommendations on social networks and date recommendations on online dating systems, many works such as [16, 5] do not mention the need for reciprocity. These works seem to focus on the task of satisfying the immediate need of the user at hand. However, improving the immediate satisfaction of the users by providing them with people they like or believe to be their friends, might not reflect the final satisfaction of the user if these relationships are not mutual.

Work on referral systems in social networks such as [17] could also benefit from reciprocity; in particular when agreement between users (or their agents) is required. One such task is the business partner identification in social networks [15].

Little is know about the users' preference when they are new to the system. The lack of knowledge about the user restricts the power of the system to create recommendations. Several solutions were proposed to solve this problem, which is commonly referred to as the cold-start problem. Park and Chu [9] define four groups of recommendations: (1) existing items to existing users; (2) existing items to new user; (3) new items to existing users; (4) new items to new users. The cold-start problem is relevant for all groups except (1). For group (2) recommending popular items is a good baseline, while for group (3) an approach that takes into account the content of the items is needed. According to the authors, group (4) is a hard case that needs a "random" strategy. Park and Chu compared several strategies to generate the recommendations for the cold start cases and found that their method, pairwise preference regression, outperforms other known methods such as random, most popular, segmented most popular, and Vibes Affinity [7].

In contrast to the cold-start problem, controlling overspecialisation is particularly important for reciprocal recommenders in domains such as online dating, where allowing variability and a wide spread of recommendations is important. In online dating some users receive lots of attention, while at the same time many other users are being neglected. It is important for a recommender in this domain not to overload the popular users and to allow neglected users to be present in the other people's recommendations.

One strategy to ensure a balanced distribution of recommendation among users of different popularity is to recommend users among these groups. For instance popular users would be recommended to other popular users, while less popular users will be recommended to users with similar popularity scores. This could minimize the effects described in [4], which demonstrated the tendency of collaborative filtering to recommend popular items even when the starting items are not popular.

The work of Abbassi et al. [1] deals with overspecialisation by finding regions in the item space that are relatively unexplored by a user. Similarly, Onuma et al. [8] applies a graph-based method to recommend items that are not centred in the user's current interest area, but are borderline between distinct areas of interest. This ensures novelty, and provides certain variety in the recommendations, that is seen favourably by users [12].

## 3 Reciprocal Recommender

A recommender $R1$ is a system that, when given a user $u$, recommends a list of items $I$ such that the degree of preference between a user and every item in $I$ is larger than the degree of preference between the same user and every items not in $I$. This is shown in Equation (1).

$$R1(u) = \{i : P1(u,i) > P1(u,j), \forall i \in I, \forall j \notin I\} \tag{1}$$

where $P1$ represents how much a user prefers an item.

Because a reciprocal recommender needs to consider the degree of preference for the items in $I$ we can build a recommender $R2$ that gives the best users $U$ for an item $i$, such that the degree of preference of an item $i$ for every user in $U$ is larger than every user not in $U$ (see Equation 2).

$$R2(i) = \{u : P2(i,u) > P2(i,v), \forall u \in U, \forall v \notin U\} \tag{2}$$

where $P2$ represents how much a user is preferred by an item (normally represented by another user).

Therefore, the reciprocal recommender $RR$ for a user $u$ is a set of items $I$ (subset of $R1(u)$) such that $u$ is in the list of recommendations $R2(i)$ for all items $i$ in $I$ (see Equation 3).

$$RR(u) = \{i : i \in R1(u) \text{ and } u \in R2(i)\} \tag{3}$$

### 3.1 Combining recommenders

To obtain a single list of recommendations from a reciprocal recommender, we need to combine $R1$ and $R2$. Such a combination is necessary if the recommendations are to take into consideration the preferences of both user and item. Depending on the domain, we may choose different methods of combination that assign different weights or meanings to each of $R1$ and $R2$. Using the terminology of Burke [3], we apply the Cascade, Weighted and Switched methods of combination in our discussion below.

If both $R1$ and $R2$ produce unranked sets of users/items as described above, we may combining their output by filtering out the items in $R1$ that are not reciprocated in $R2$ (Equation 3). This method treats $R1$ and $R2$ equally and is simple to compute.

If we assign a numeric score to the recommendations in $R1$ and $R2$ using $P1$ and $P2$, we may produce a combined ranked output $PRR$ by calculating a weighted sum of the scores for each user/item pair (Equation 4). Using the flexibility in the choice of weights, it is possible to give focus to either the user's preference or the item's preferences. This customisability is useful in many cases. For example, if the user only wants items that are highly tailored to his/her needs and does not mind having a few unsuccessful interactions, we can give a higher weight to the user's recommendations.

$$PRR(u,i) = w_1 P1(u,i) + w_2 P2(i,u) \tag{4}$$

On the other extreme, if we need to recommend for a (possibly new) user for which we have no preference information about, we may choose to entirely rely on the item's preference for users to give the user items which will like the user. This can be used to mitigate the cold start problem.

## 4 Approaches to reciprocal recommendation

Recommender systems approaches are normally divided into two main classes: content-based and collaborative filtering. Content-based recommender systems take into account the content of the items that have been used by a user in order to find the likes and dislikes of the user and by using these preferences the system can find new and unknown items to present to the user. Conversely, collaborative filtering does not take into account the content of the items, but instead analyses their usage patterns. For instance, if a group of items used by a user $A$ was also used by other users $B$ and $C$, an item used by $B$ and $C$ and not used by $A$ is a potential good recommendation to present to $A$.

When both users and items are actively engaged in the search of each other, then it would be possible to create a reciprocal recommender using any technique and finding the overlapping pair of users and items. For instance, in a job search scenario where one side is looking for job positions and the employee is looking for suitable candidates, a system can be built that finds all positions which will consider a particular candidate suitable. The same logic can be applied for the employer: a system can find all candidates who consider the position attractive to them. A system that is based on the overlapping recommendation is described by [6].

However, when one side of the reciprocal recommender is not actively engaged with the search, group generalisations may become necessary. For instance, if a job post is advertised but the advertiser company does not actively search for employees in the website database, then a system can generalise the job preferences by using previous similar positions by the same advertiser or company, or even previous similar positions by any company. This is one way of dealing with the cold start problem. Generalisations can create more data and therefore minimise the problems associated with lack of data. However, it is unclear when these types of generalisations can be made.

Because similar people[2] might not act similarly, generalisations are harder to obtain when users are acting as individuals. Nevertheless, in many cases, building stereotypes for people is required, because without it there might not be any indication of what users might like. The lack of indication of preferences can also arise from the specific task at hand, which might require users or items to have one and only one successful interaction. For example, when someone finds a job using a job search website, it is likely that this person will stop using the website and the job position will be closed. For this type of task, scaling down the success requirements and finding intermediate levels of interest are required. For instance, the success of a job search website might not be defined as a job position being fulfilled, but rather the engagement of the website's users such that candidates submit applications and positions receive applications from candidates. In this way, success is a less strict criteria, and preferences can be more easily defined.

Collaborative filtering has been shown to work well for non-reciprocal recommenders, and can be easily applied to a reciprocal setting. However, in situations when generalisations are required by using previous positions and transactions, the set of users who obtained successful interactions might not be currently available. On the other hand, content-based techniques are better able to generalise preferences that were expressed previously by employers in different posts as these techniques are normally not bound by each individual candidate, but by the attributes of these users. In this way, a hybrid approach for a reciprocal job recommender could simply consist of two recommenders: one using collaborative filtering to recommend job positions to a candidate and one content-based to recommend candidates for a job position based on previous transactions between similar job positions and candidates.

Traditional recommenders can also be used to generate recommendations for reciprocal tasks. All successful and unsuccessful reciprocal transactions are used to train the recommender, which will recommend future items which are likely to reciprocate a transaction. However, this is only possible for systems that contain recurrent or long term users who can and will perform multiple successful transactions. For those systems where users are short-term and their expectations are to find a life-long partner or a life-long job career, the use of a combined, reciprocal recommender system is preferable.

Independently of the technique used, the combined reciprocal recommender must account for the lack of information about its users and items. The way of handling the cold-start problem is likely to be one of the major factors influencing the performance of such a system.

Figure 1 illustrates a reciprocal candidate-employer recommender that uses a recommender $R1$ for the candidate and a recommender $R2$ for the employer.

[2] Assuming we have a clear definition of similarity for people. However the concept of personal similarity is a whole problem on its own, which we will not address in this paper.

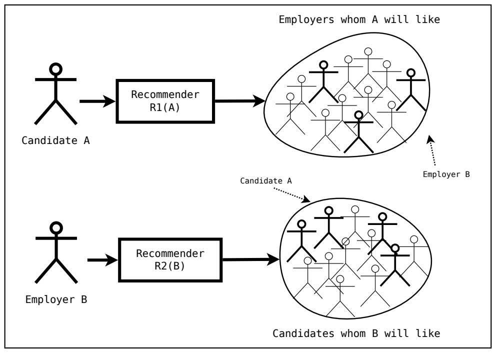

**Fig. 1.** Example of a reciprocal candidate-employer recommender. Recommendations drawn with thicker lines represent those recommendations which are reciprocal (i.e. Employer $B$ appears in the list of recommended employers for Candidate $A$ and Candidate $A$ appears in the list of recommendations for Employer $B$).

The final recommendation for a given candidate $A$ is a ranked list of employers (or positions) where the top of the list is populated by those employers whose recommenders suggest the current candidate as someone they might like (e.g. Employer $B$). Likewise, the final recommendation for a given employer $B$ is a ranked list of candidates where the first candidates are those who would like to be employed by the company (e.g. Candidate $A$).

Malinowski et al. [6] suggest seeking a Pareto-optimal solution to incorporate the needs of both user and item in a bipartite matching problem. However, we believe that for a reciprocal recommender, one must not seek a Pareto-optimal solution in domains where the recommender accuracy is low, as such a solution assumes that recommendations are mostly successful. For example, in a job recommender, a Pareto-optimal solution may lead us to recommend the "second best" jobs to a person, because we think there are better candidates who will take the job. This decision is not sensible unless we are certain that our predictions are very accurate.

## 5 Case study - Online dating

Online dating is one of the areas where a reciprocal recommender is very important. In the online dating domain, the item being recommended to a user is another user who also has the same goal when using the system: to find a date. Any recommendation given in a online dating scenario needs to be reciprocal and must take into account the needs of both users being recommended to each other. Otherwise, if a recommendation is given with only one of the users in mind, these “good” non-reciprocal recommendations will be short lived because user interactions resulting from these recommendations are likely not to develop further.

**Table 2.** Different levels of interest shown by users in dating websites

| Action | $A$ likes $B$ | $B$ likes $A$ |
|---|---|---|
| $A$ reads profile of $B$ | Possibly | Unknown |
| $A$ reads profile of $B$, $A$ sends a message to $B$ | Yes | Unknown |
| $A$ reads profile of $B$, $A$ does not send a message to $B$ | No | Unknown |
| $A$ sends a constrained message to $B$ | Yes | Unknown |
| $A$ sends a constrained message to $B$, $B$ replies positively to $A$ | Yes | Yes |
| $A$ sends a constrained message to $B$, $B$ replies negatively to $A$ | Yes | No |
| $A$ sends a unconstrained message to $B$ without previous constrained communication | Yes | Unknown |
| $A$ sends a unconstrained message to $B$ without previous constrained communication and receives a reply | Yes | Unknown* |
| $A$ sends a unconstrained message to $B$ without previous constrained communication and does not receive a reply | Yes | Unknown* |

Dating can translate into finding a life-long partner or a casual/short-term partner. Both types of users have crucial differences that have high implications for an online dating website and its recommender system. Casual and short-term relationship seekers are likely to use the website for longer periods of time and are more likely to have “successful” relationships with different people, while long-term relationship seekers are hoping to find that one person who will cause them to stop using the website.

Online dating websites (e.g. Yahoo! Personals, Match.com) allow users to create their profiles, browse and create constrained conversations[3] for free, while charging a fee for unconstrained communication such as email, chat and telephone call. Although, in theory, success is measured by the number of people who have found a partner using the website, in practice the best and easiest way of measuring success is the use of unconstrained communication. The increase in unconstrained communication is really important to online dating websites because it can lead to a real world dating scenario and also because it directly relates to most online dating websites' business models.

Online dating is an intrinsically reciprocal task which is very difficult for recommender systems for several reasons: (1) Online dating deals with a range of features that might not be represented in the website data, such as private personal expectations or experiences from previous relationships. (2) There are multiple fuzzy levels of interests between users, which are difficult to capture. (3) Users may seek communication with others whose profiles do not precisely agree with the users' explicit preferences. (4) People change preferences over time.

Difficulty (1) is beyond the scope of this research as it involves psychological and sociological issues, which we cannot currently address. The different levels of interests of difficulty (2) can be addressed during website design. Existing websites show different levels of interest similar to the ones shown in Table 2. Some of the level of interest of user $B$ toward user $A$ that are marked as unknown and are marked with an asterisk can only be determined if the exchanged messages between users are read and analysed.

[3] Constrained conversation are fixed and predefined messages that users exchange in order to show interest (or not) in each other.

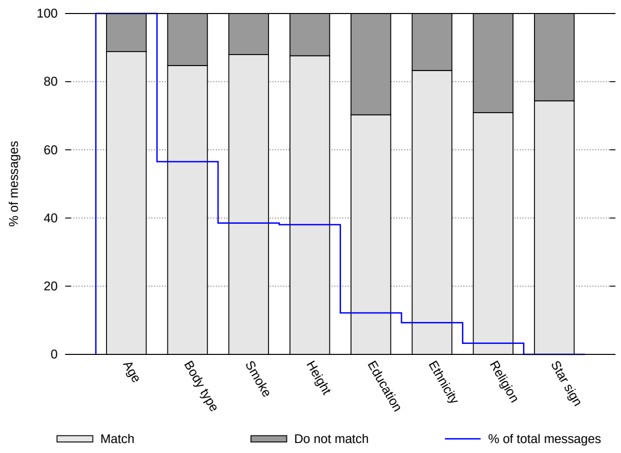

**Fig. 2.** Percentage of contacts from users which follows and does not follow the extrinsic model of preferences defined by the user

Difficulty (3) involves the creation of intrinsic models of user preference and how they differ from the extrinsic models created by the user. We observed that although people can define the characteristics of those they wish to date, users of online dating systems communicate with people whose features are different to the ones they specified. Figure 2 shows the percentage of constrained messages that were sent to a user who did not possess an important feature that was specified by the sender. For instance, 14% of messages from people with extrinsic preference were sent by users who specified education as an important attribute, 30% of those messages were sent to users who had a different level of education to the one specified in the sender's preferences. This either indicates that users cannot properly define the desired characteristics of their ideal date, or that users are willing to deviate from their ideal date when a *potentially good* date is found.

We believe that although an ideal date description is better than no description at all, the most reliable representation of someone's preferences can be inferred using the person's online date contacts. The inferred representation can be used to create reciprocal or non-reciprocal recommenders that produce lists of recommendations that match each user's own preferences.

As most content-based recommenders, such recommenders based on preference may suffer from lack of training data and over-specialisation. It is important to account for changes to user preferences (difficulty 4); such an effect can be harder to identify when a large body of previous contact is used to inferred the user's preferences. It is also important to learn how much evidence is needed to be able to provide a reliable recommendation. For instance, if user $A$ contacted $B$ and $B$ is the only user contacted by $A$, a recommender system cannot assume that $A$ only wants to contact users with the same features as $B$.

The use of reciprocal recommenders in online dating allows the creation of meaningful recommendations even when no preferences can be established for some users. For instance, in the cold start problem, if a user is new to the system and has not explicitly defined his/her preferences, nor contacted any other user (i.e. no implicit preferences), we can assume that this user does not have a preference and he/she will like any user. With this assumption, we can either create a list of recommendations for recommender $R1$ (or for recommender $R2$) that involves all users in the system. With such list, the reciprocal recommendation $RR(u)$ for a new user $u$ is all users $i$ whose preferences match user $u$. Similarly, if we only know the preferences of a user $x$ then the reciprocal recommender $RR(x)$ for this user is equivalent to his non-reciprocal recommender $R1(x)$.

It is important to highlight that the degree of preference between a user with no known preferences and all users in the database has to be kept small, but non-zero. The degree of preference must be kept small so it does not interfere with the reciprocity when preferences are known. In this way, a ranked list of recommendations should contain all users who reciprocally match each other's preferences followed by a list of users who likes the user being recommended or is liked by him/her.

Another difficulty with implementing reciprocal recommenders and in particular with dating websites is that although some users are clearly more popular than others, the website needs to be careful to balance the load of the recommendations in order not to overwhelm users and to provide a good opportunity for interaction to all the users. Popular users might have characteristics that make them popular, such as a beautiful face or a charming smile, but they will not respond positively to everyone. On the other hand, there is a large number of users who do not have the same appeal as the popular users but are more likely to engage in successful interactions with other users.

## 6 Concluding remarks

In this paper we defined Reciprocal Recommenders, a class of recommender systems applicable to a number of domains such as online dating, recommending mentors, business partners or friends. This type of recommenders differs from the traditional recommender systems and has not received sufficient attention in the recommender community. We reviewed the relevant literature highlighting the need and importance of reciprocity in certain tasks. We identified challenges and discussed promising approaches to address them. Finally, we presented a case study in the area of online dating, illustrating the important concepts.

We have already implemented different techniques for reciprocal recommendations, which we plan to evaluate using data from an existing online dating website and also from an employment website. Future work will also include measuring the impact of reciprocity in domains where reciprocity is not obvious such as in online auctions and classified advertising.

## Acknowledgements

This research was funded by the Smart Services Co-operative Research Centre.

## References

1. Z. Abbassi, S. Amer-Yahia, L. V. Lakshmanan, S. Vassilvitskii, and C. Yu. Getting recommender systems to think outside the box. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 285–288, New York, 2009.
2. L. Brožovský and V. Petříček. Recommender system for online dating service. *CoRR*, abs/cs/0703042, 2007.
3. R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
4. O. Celma and P. Cano. From hits to niches?: or how popular artists can bias music recommendation and discovery. In *NETFLIX '08: Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition*, pages 1–8, New York, NY, USA, 2008. ACM.
5. R. F. Emmerink. Ai dating: Development of a novel dating application with fuzzy inferencing. Master's thesis, Faculty of Computing Sciences and Engineering, De Montfort University, September 12 2008.
6. J. Malinowski, T. Keim, O. Wendt, and T. Weitzel. Matching people and jobs: A bilateral recommendation approach. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, volume 6, page 137c, 2006.
7. B. Nag. Vibes: A platform-centric approach to building recommender systems. *IEEE Data Eng. Bull.*, 31(2):23–31, June 2008.
8. K. Onuma, H. Tong, and C. Faloutsos. Tangent: a novel, 'surprise me', recommendation algorithm. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 657–666, 2009.
9. S. T. Park and W. Chu. Pairwise preference regression for cold-start recommendation. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 21–28, New York, NY, USA, 2009. ACM.
10. D. Richards, M. Taylor, and P. Busch. Expertise recommendation: A two-way knowledge communication channel. *Autonomic and Autonomous Systems, International Conference on*, 0:35–40, 2008.
11. J. B. Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Min. Knowl. Discov.*, 5(1-2):115–153, 2001.
12. K. Swearingen and R. Sinha. Beyond algorithms: An HCI perspective on recommender systems. In *ACM SIGIR'01 Workshop on Recommender Systems*, 2001.
13. L. Terveen and D. W. McDonald. Social matching: A framework and research agenda. *ACM Transactions on Computer-Human Interaction*, 12(3):401–434, 2005.
14. J. Vassileva, G. Mccalla, and J. Greer. Multi-agent multi-user modeling in i-help. *User Modeling and User-Adapted Interaction*, 13(1-2):179–210, 2003.
15. P. L.-K. Wong and P. Ellis. Social ties and partner identification in sino-hong kong international joint ventures. *J. Int. Bus. Stud.*, 33(2):267–289, 2002.
16. Z. Wu, S. Jiang, and Q. Huang. Friend recommendation according to appearances on photos. In *MM '09: Proceedings of the seventeen ACM international conference on Multimedia*, pages 987–988, New York, NY, USA, 2009. ACM.
17. B. Yu and M. P. Singh. Searching social networks. In *AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 65–72, New York, NY, USA, 2003. ACM.