Maximizing Aggregate Recommendation Diversity: A Graph-Theoretic Approach

Gediminas Adomavicius
Department of Information and Decision Sciences
University of Minnesota
gedas@umn.edu

YoungOk Kwon
Department of Information and Decision Sciences
University of Minnesota
kwonx052@umn.edu

ABSTRACT
Recommender systems are being used to help users find relevant items from a large set of alternatives in many online applications. Most existing recommendation techniques have focused on improving recommendation accuracy; however, diversity of recommendations has also been increasingly recognized in research literature as an important aspect of recommendation quality. This paper proposes a graph-theoretic approach for maximizing aggregate recommendation diversity based on maximum flow or maximum bipartite matching computations. The proposed approach is evaluated using real-world movie rating datasets and demonstrates substantial improvements in both diversity and accuracy, as compared to the recommendation re-ranking approaches, which have been introduced in prior literature for the purpose of diversity improvement.

Keywords
Recommendation diversity, aggregate diversity, collaborative filtering, graph-based algorithms.

1. INTRODUCTION AND MOTIVATION
Many recommendation techniques have been developed over the past decade, and major efforts in both academia and industry have been made to improve recommendation accuracy, as exemplified by the recent Netflix Prize competition. However, it has been increasingly noted that it is not sufficient to have accuracy as the sole criteria in measuring recommendation quality, and we should consider other important dimensions, such as diversity, novelty, serendipity, confidence, trust, to generate recommendations that are not only accurate but also useful to users [19,29,34].

In this paper, we focus on the *aggregate diversity* of recommendations, which has recently attracted attention in research literature due to its impact on the shifts in product variety and sales concentration patterns [11,12,15,31]. As observed by Brynjolfsson et al. [12], recommender systems can play a key role in increasing both "long tail" and "superstar" effects in real-world e-commerce applications. In particular, the "long tail" literature argues that recommendations on the Internet help to increase users' awareness of niche products and create a long tail in the distribution of product sales [6,11,15,31]. For example, one study, using data from online clothing retailer, demonstrates that recommendations would increase sales of the items in the long tail, resulting in the improvement in aggregate diversity [11]. In contrast, the "superstar" literature indicates that recommender systems may promote the so-called "rich get richer" phenomenon, where users are recommended more popular/bestselling items than idiosyncratic/personalized ones. One explanation for this is that the niche products often have limited historical data and, thus, are more difficult to recommend to users, whereas popular products typically have more ratings and, thus, can be recommended to more users [15,26,36].

Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011), held in conjunction with ACM RecSys 2011. October 23, 2011, Chicago, Illinois, USA.

More diverse recommendations, presumably leading to more sales of long-tail items, could be beneficial for both individual users and some business models [10,11,18]. Exposing individual consumers to more long-tail recommendations can intensify this effect. Thus, more consumers would be attracted to the companies that carry a large selection of long tail items and have long tail strategies, such as providing more diverse recommendations [12]. Also, some business models (e.g., Netflix), can benefit from recommendation diversity, because more diverse recommendations would encourage users to rent more long-tail movies, which are less costly to license and acquire from distributors than new releases or extremely popular movies of big studios [18].

Taking into consideration the potential benefits of aggregate diversity (hereinafter simply diversity) to individual users and businesses, several studies have explored new methods that can increase the diversity of recommendations [2,3,23,27,32]. In particular, considering that recommender systems typically compute recommendations to users in two phases – (Phase 1) estimating ratings of items that the users have not consumed yet and (Phase 2) generating top-*N* items for each user – the prior work can be divided into two lines of research. One line of research [23,27,32] aims to enhance the estimation phase (mainly for long tail items), and the other focuses on finding the best set of recommendations in the recommendation generation phase [2,3]. The approach proposed in this paper fits within the latter line of research and, therefore, has the flexibility of being used in conjunction with *any* available rating estimation algorithms, as illustrated by our empirical evaluation. In contrast to simple recommendation re-ranking heuristics for diversity improvement proposed in [2,3], we develop a more sophisticated and systematic graph-based approach for direct diversity maximization, while maintaining acceptable levels of accuracy.

Our empirical results, using real-world rating datasets, show that the proposed graph-based approach consistently outperforms the recommendation re-ranking approach from prior literature in terms of both accuracy and diversity. The paper also discusses the scalability of the proposed approach in terms of its theoretical computational complexity as well as its empirical runtime based on real-world rating datasets.

2. RELATED WORK
In this section, we briefly discuss two widely used recommendation techniques that are used in conjunction with our proposed approach in our empirical experiments as well as two important dimensions in the evaluation of recommendation quality: accuracy and diversity. We also discuss a simple recommendation re-ranking approach from prior literature, which has been shown to improve the aggregate diversity of recommendations with only a small loss of accuracy, and which

we will use as one of the baseline comparison techniques.

2.1 Recommendation Algorithms

Let U be the set of users and I be the set of items available in the recommender system. Then, the usefulness or utility of any item i to any user u can be denoted as $R(u,i)$, which usually is represented by a rating (on a numeric, ordinal, or binary scale) that indicates how much a particular user likes a particular item [1]. Thus, the job of a recommender system in the rating estimation phase (Phase 1) is to use known ratings as well as other information that might be available (e.g., content attributes of items or demographic attributes of users) to estimate ratings for items that the users have not yet consumed. For clarity, we use $R(u,i)$ to denote the actual rating that user u gave to item i, and $R^*(u,i)$ for the system-estimated rating for item i that user u has not rated before. Given all of the unknown item predictions for each user, in generating top-*N* recommendations (Phase 2) the system selects the most relevant items, i.e., items that maximize a user's utility, according to a certain ranking criterion. More formally, item i_x is ranked ahead of item i_y, if $rank(i_x) < rank(i_y)$, where $rank\colon I \rightarrow \mathbf{R}$ is a function representing some ranking criterion. Most recommender systems rank the candidate items by their *predicted rating value* and recommend to each user the N most highly predicted items (where N is a relatively small positive integer) because users are typically interested in (or have time for) only a limited number of recommendations. We refer to this as the *standard ranking approach* and can formally define the corresponding ranking function as $rank_{\text{Standard}}(i)=R^*(u,i)^{-1}$. While the standard ranking approach exhibits good recommendation accuracy, its performance in terms of recommendation diversity is poor [2,3], which further emphasizes the need for different recommendation approaches for diversity improvement.

Among a large number of recommendation techniques that have been developed over the past decade, collaborative filtering (CF) techniques represent most widely used and well-performing algorithms; we use two representative CF techniques for Phase 1 (i.e., rating estimation) in this paper: neighborhood-based CF and matrix factorization CF techniques.

Neighborhood-based CF techniques. The basic idea of neighborhood-based CF techniques is, given a target user, to find the user's neighbors who share similar rating patterns, and then to use their ratings to predict the unknown ratings of the target user [1,9]. There are many variations of computational methods to identify a user's neighbors (i.e., by computing the similarity between users) and aggregate the neighbors' ratings for the user. In our experiments, we use a popular cosine similarity measure for calculating similarity between users, and the final rating prediction for a specific item to a user is made as an adjusted weighted sum of the ratings of the user's closest 50 neighbors on this item. The neighborhood CF techniques can be user- or item-based, depending on whether the similarity is computed between users or items [33]; we use both variations in this paper.

Matrix factorization CF techniques. Matrix factorization CF techniques have recently gained popularity because of their effectiveness in the Netflix Prize competition in terms of predictive accuracy. In contrast to heuristic-based techniques (such as the neighborhood-based CF techniques mentioned above), the matrix factorization CF techniques use the existing ratings to learn a model with k latent variables for users and items. In other words, this technique models and estimates each user's preferences for k latent features as the user-factors vector and each item's importance weights for the k latent features as the item-factors vector [16,24]. Then, the predicted rating of item i for user u can be computed as an inner product of the user-factors vector for user u and the item-factors vector for item i. Typically, the model-based techniques have been shown to generate more accurate recommendations than heuristic-based techniques. While a number of variations for the matrix factorization technique have been developed, in this paper we use its basic version, as proposed by Funk [16].

2.2 Recommendation Accuracy and Diversity

Recommendation Accuracy. The goal of this work is to generate good top-*N* recommendation lists in terms of accuracy and diversity and, accordingly, we chose to evaluate the accuracy of top-*N* recommendation lists using one of the most popular decision-support metrics, *precision* [19]. Simply put, precision is measured as a proportion of "relevant" items among the recommended items across all users. Note that the decision-support metrics, such as precision, typically work with binary outcomes; therefore, here the notion of "relevance" is used to convert a numeric rating scale (e.g., 1-5) into binary scale (i.e., relevant vs. irrelevant).

More specifically, in our empirical data ratings are provided on a 5-point (or 5-star) scale, and the natural assumption is that users provide higher ratings to the items that are more relevant to them. As a consequence, in our experiments, we treat items with ratings 4 and 5 as relevant, and items with ratings 1, 2, 3 as irrelevant, or, more precisely, we choose the threshold between relevant or and irrelevant items as 3.5 (denoted by T_H). The list of N items recommended for user u should include only items predicted to be relevant and can be formally defined as $L_N(u) = \{i_1, i_2, \ldots, i_N\}$, where $R^*(u, i_k) \geq T_H$ for all $k \in \{1, 2, \ldots, N\}$. The precision of such top-*N* recommendation lists, often referred to as *precision-in-top-N*, is calculated as the percentage of truly "relevant" items, denoted by $correct(L_N(u)) = \{i \in L_N(u) \mid R(u, i) \geq T_H\}$ among the items recommended across all users, and can be formalized as:

$$precision\text{-}in\text{-}top\text{-}N = \sum_{u\in U} |correct(L_N(u))| \Big/ \sum_{u\in U} |L_N(u)|.$$

In real-world settings, obviously a recommender system has to be able to recommend items that users have not yet rated (the ratings for those items typically become available to the system only after item consumption), i.e., the true precision of the generated recommendation lists is not known at the time of recommendation. However, using two popular real-world datasets (details on datasets are provided in Section 4), different popular CF recommendation algorithms discussed above, and standard cross-validation techniques from machine learning and data mining, we show that, not surprisingly, precision is highly correlated with average predicted rating value of recommended items using for all recommendation algorithms, as indicated in Fig. 1. In other words, recommending items with higher predicted rating values results in higher precision (i.e., higher likelihood that the user would actually like the item), which provides further empirical support for using the standard ranking approach if the goal is just to maximize recommendation accuracy. An important consequence of this relationship is that we can use the average predicted rating value of top-*N* recommendation lists, which can always be computed at the time of recommendation, as a simple proxy for the precision metric. In addition, this metric is extremely simple to compute and easily scales to large-scale real-world applications. We refer to this metric as *prediction-in-top-N* and formally define it as follows:

$$prediction\text{-}in\text{-}top\text{-}N = \sum_{u \in U} \sum_{i \in L_N(u)} R^*(u,i) \Big/ \sum_{u \in U} |L_N(u)|$$

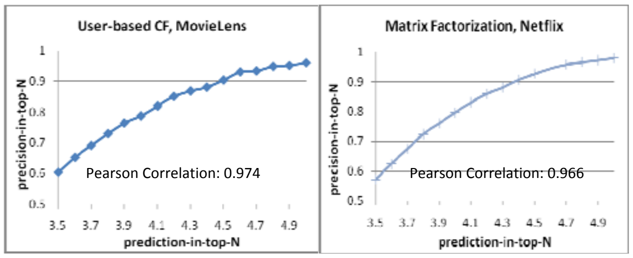


Figure 1. Precision versus Average Predicted Rating Value

Recommendation Diversity. As discussed earlier, accurate recommendations are not always useful to users. For example, recommending only popular items (e.g., blockbuster movies that many users tend to like) could obtain high accuracy, but also can lead to a decline of other aspects of recommendations, including recommendation diversity. The inherent tradeoff between accuracy and diversity has been observed in previous studies [2,3,30,38,39], therefore, indicating that maintaining accuracy while improving diversity constitutes a difficult task.

The diversity of recommendations has been assessed at either individual or aggregate level. The majority of previous studies have focused on individual diversity [8,21,30,35,37,38,39]. For example, the individual diversity of recommendations for a user can be measured by calculating an average dissimilarity between all pairs of items recommended to a user (e.g., based on item attributes). In contrast, the aggregate diversity of recommendations across all users has been relatively less studied, and the recent interest in the impact of recommender systems on product variety and sales concentration patterns [11,12,15,31] has sparked a renewed interest in this topic.

Several metrics can be used to evaluate various aspects of aggregate diversity, including absolute long-tail metrics that measure the change in the absolute number of items recommended (e.g., recommendation frequency of items above a certain popularity rank), relative long-tail metrics to measure the relative share of recommendations above or below a certain popularity rank percentile, and the slope of the log-linear relationship between item popularity rank and recommendations (or sales) that can indicate the relative importance of the head versus the tail of the distribution [12]. In the recommender systems literature, both absolute and relative long-tail metrics have been used to measure the aggregate diversity of recommendations [2,3,19,23,27,37]. In this paper, we use a simple absolute long-tail metric which measures aggregate diversity using the total number of distinct items among the top-*N* items recommended across all users, referred to as the *diversity-in-top-N* [2,3]. More formally:

$$diversity\text{-}in\text{-}top\text{-}N = \left| \bigcup_{u \in U} L_N(u) \right|,$$

and prior research has shown that this simple and easy-to-compute metric exhibits high correlation with more sophisticated, distributional diversity metrics [3], i.e., is able to properly capture the same diversity dynamics as some of the relative long-tail metrics on several real-world rating datasets. This diversity metric could also potentially be viewed as a crude indicator of the system's level of personalization, because high diversity implies that each user gets very different and unique set of recommendations (potentially indicating a high level of personalization), whereas low diversity indicates that mostly the same items (possibly bestsellers) are recommended to all users (i.e., low level of personalization).

Although the approach proposed in this paper aims to improve aggregate recommendation diversity, their accuracy is also given the proper attention in the paper, because diverse but inaccurate recommendations may not provide significant value to the users.

2.3 Re-Ranking Approaches for Diversity

Several prior studies have explored improving aggregate diversity of recommendations [2,3,23,27,32]. As discussed earlier, one line of research proposes new methods for predicting unknown ratings, mainly for long-tail items. For example, Park and Tuzhilin [32] propose new clustering methods to improve predictive accuracy of long-tail items that have only few ratings, which can also increase the recommendation of long-tail items. In addition, Levy and Bosteels [27] design long-tail music recommender systems, simply by removing popular artists (i.e., with more than 10,000 listeners) in the rating prediction phase. Also, a local scoring model, proposed by Kim et al. [23], was developed to alleviate the scalability and sparsity problems by suggesting a more efficient way to select the best neighbors for neighborhood-based recommendation techniques; however, as a by-product, it is shown to improve aggregate recommendation diversity.

In contrast to these studies, another line of research proposes new approaches for improving top-*N* item selection *after* the rating estimation is performed. In particular, Adomavicius and Kwon [2,3] propose a heuristic approach for recommendation re-ranking, which has been shown to improve aggregate diversity with a negligible accuracy loss and represents an important baseline for comparison with our proposed diversity maximization approaches. Typical recommender systems recommend to users those items that have the highest predicted ratings, i.e., using the standard recommendation ranking criterion $rank_{\text{Standard}}$, as discussed earlier. While the standard ranking approach is used to maximize the accuracy of recommendations, as was illustrated by Fig. 1, Adomavicius and Kwon [2,3] showed that changing the ranking of items (i.e., not following the standard ranking approach) can help with other aspects of recommendation quality, in particular, with recommendation diversity. As a result, they proposed several alternative re-ranking approaches, and showed that all of them can provide substantial improvements in recommendation diversity with only negligible accuracy loss. In our experiments, as a baseline for comparison, we specifically use the ranking approach based on the reverse predicted rating value. This is a personalized yet simple and highly-scalable ranking approach that can be formally defined as $rank_{\text{RevPred}}(i) = R^*(u,i)$.

While this re-ranking approach can significantly improve recommendation diversity, as might be expected, this improvement comes at the expense of recommendation accuracy, since not the most highly predicted items are recommended. Adomavicius and Kwon [2,3] demonstrate that the balance between diversity and accuracy can be achieved by parameterizing any ranking function with "ranking threshold" $T_R \in [T_H, T_{\max}]$ (where $T_{\max}$ is the largest rating on the rating scale). That is, the ranking threshold enables to specify the level of acceptable accuracy loss while still extracting a significant portion of diversity improvement. The parameterized version $rank_{\text{RevPred}}(i, T_R)$ of ranking function $rank_{\text{RevPred}}(i)$ can be implemented as:

$$rank_{\text{RevPred}}(i,T_R) = \begin{cases} rank_{\text{RevPred}}(i), & if\ R^*(u,i) \in [T_R, T_{\max}] \\ \alpha_u + rank_{\text{Standard}}(i), & if\ R^*(u,i) \in [T_H, T_R) \end{cases},$$

where $\alpha_u = \max_{i \in I_u^*(T_R)} rank_{\text{RevPred}}(i)$, and $I_u^*(T_R) = \{i \in I \mid R^*(u,i) \ge T_R\}$.

In particular, items with predicted ratings from [T_{R}, $T_{\max}$] would be ranked ahead of items with predicted ratings [T_{H}, T_{R}), as ensured by α_u in the above definition. Increasing the ranking threshold T_{R} towards $T_{\max}$ would enable choosing the most highly predicted items (i.e., more accuracy and less diversity – similar to the standard ranking approach), while decreasing the ranking threshold T_{R} towards T_{H} makes $rank_{\text{RevPred}}(i,\ T_{\text{R}})$ increasingly more similar to the pure ranking function $rank_{\text{RevPred}}(i)$, i.e., more diversity with some accuracy loss. Thus, choosing $T_{\text{R}} \in [T_{\text{H}}, T_{\max}]$ values in-between the two extremes allows setting the desired balance between accuracy and diversity. In our experiments, we are able to explore the accuracy-diversity tradeoff of the re-ranking approach, by varying this ranking threshold T_{R}.

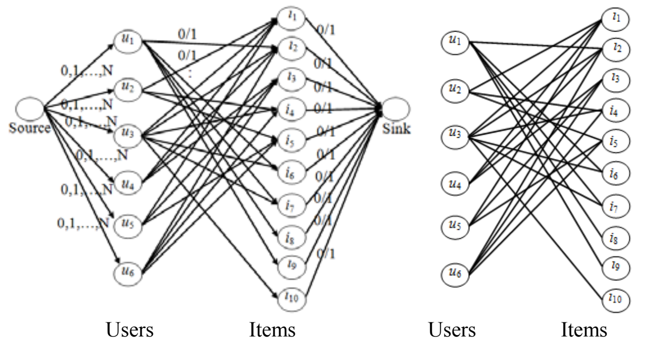
In terms of computational complexity, the re-ranking approach is implemented as a simple sorting algorithm. Assuming there are m users and n items, the worst case situation for this algorithm occurs when all n items are available to every user for recommendation. Then, the heuristic-based ranking does the job of sorting n items, $O(n\log n)$, for m users, and its complexity would be $O(mn\log n)$.

3. PROPOSED APPROACH

While the recommendation re-ranking approach can obtain a certain level of diversity gains at the expense of a small loss in accuracy, in this section we propose a graph-based approach that can obtain maximum possible diversity.

Graph-based algorithms have been previously used in recommender systems [4,22,28], mostly for the purpose of improving predictive accuracy of CF techniques. We formulate our problem of diversity maximization as a well-known *max-flow problem* in graph theory [5,14]. One simple version of the general maximum flow problem, which has been extensively studied in operations research and combinatorial optimization, can be defined as follows. Assuming that V is the set of vertices (or nodes), and E is the set of directed edges, each of which connects two vertices, let $G = (V, E)$ be a directed graph with a single source node $s \in V$ and a single sink node $t \in V$. Each directed edge $e \in E$ has capacity $c(e) \in \mathbf{R}$ associated with it. Also, the amount of actual flow between two vertices is denoted by $f(e) \in \mathbf{R}$. The flow of an edge cannot exceed its capacity, and the sum of the flows entering a vertex must equal the sum of the flows exiting a vertex, except for the source and the sink vertices. The maximum flow problem is to find the largest possible amount of flow passing from the source to the sink for a given graph G.

Translating the top-N recommendation setting into a graph-theoretic framework, let users and items be represented as vertices, and an edge from user u to item i exists if and only if item i is predicted to be relevant for user u, i.e., $R^*(u, i) \ge T_{\text{H}}$ or, in other words, when the item is available to the user for recommendation. Each edge has capacity $c(e) = 1$ and can be assigned an integer flow of 1 if item i is actually recommended to user u as part of top-N recommendations, and the flow of 0 otherwise. As described in the example in Fig. 2a, we augment this directed graph by adding a source node and connecting it by directed edges to each of the user vertices. Let the capacity of each of these "source" edges be N and, again, only integer flows of 0, 1, …, or N are permitted on each of these edges. Furthermore, we also augment this graph by adding a sink node and connecting each item vertex by a directed edge to this node. Let the capacity of each of these "sink" edges be 1, and again only integer flows (i.e., 0 or 1) are permitted for these edges.


Source
0,1,…,N
0/1
Sink
u_1 u_2 u_3 u_4 u_5 u_6
i_1 i_2 i_3 i_4 i_5 i_6 i_7 i_8 i_9 i_{10}
Users
Items
Users
Items

(a) Max Flow Problem (b) Max Bipartite Matching Problem

Figure 2. Top-N Recommendation Task as a Graph Theory Problem

As can be easily seen from this construction, because of the specified capacity constraints, i.e., "source" edges not allowing flows larger than N through each user node and "sink" edges not allowing flows larger than 1 through each item node, the maximum flow value in this graph will be equal to the maximum possible number of edges from users to items that can have flow of 1 assigned to them. In other words, the max-flow value will be equal to the largest possible number of recommendations than can be made from among the available (highly predicted) items, where no user can be recommended more than N items, and no item can be counted more than once, which is precisely the definition of the *diversity-in-top-N* metric.

Note that, while finding the maximum flow will indeed find the recommendations that yield maximum diversity, since the recommendation of each item is counted only once (i.e., restricted to only one user), as part of the max-flow solution some users may have fewer than N recommendations. The remaining recommendations for these users can be filled arbitrarily, as they cannot further increase the maximum diversity. We employ the standard ranking approach for the not-yet-recommended items for each user (i.e., the remaining items with the highest predicted ratings), for the purpose of achieving better accuracy.

The maximum flow problem represents a simple and intuitive metaphor for computing top-N recommendations with maximum possible aggregate diversity, and there are many efficient (polynomial-time) algorithms for finding the maximum flow in a given graph [5,14]. Note, however, that the flow graph constructed for the diversity maximization problem is a highly specialized graph, and it may be possible to find even more effective graph-based algorithms for this problem, as compared to general-purpose max-flow algorithms.

To illustrate this, let's consider the simplest top-N recommendation setting, i.e., where $N = 1$. Since each user can be recommended only one item, *all* edges in our max-flow problem would become single-unit capacity edges, implying that the max flow in this graph will correspond to the largest possible set of edges from users to items, where no user and no item can be part of more than one such edge. Because there are no edges between two different users or between two different items (i.e., we have a bipartite user-item graph), for top-1 recommendation settings the maximum flow problem is, thus, equivalent to the more specialized *maximum bipartite matching* problem which, furthermore, has more efficient algorithmic solutions. Thus, while the max-flow approach represents a general, intuitive approach for achieving maximum diversity by implementing a single-source and single-sink flow network, we follow the equivalent yet more efficient maximum bipartite matching approach (as illustrated in Fig. 2b) and also show how it can be extended from top-1 to the more general top-N settings.

As summarized in Fig. 3, our max-flow/matching optimization approach consists of two steps: (1) find maximum diversity by solving the maximum bipartite matching problem and (2) complete top-N recommendations by applying the standard ranking approach. Since the maximum diversity in Step 1 can be obtained at some expense of accuracy, one can control the balance between accuracy and diversity with the simple parameterization of a "flow-rating threshold" $T_F \in [T_H, T_{max}]$. This allows pre-processing of the data, specifically, to include only higher predicted items (i.e., above T_F) among the items that can be recommended for the maximum diversity in Step 1. Similarly to how the ranking threshold was used in re-ranking approaches (Section 2.3), here the lowest T_F value provides the best diversity but a relatively lower accuracy, whereas higher values of T_F lower the diversity but provide a certain level of accuracy. Then, in Step 2, the highest predicted remaining items are used to complete top-N recommendation lists.

More formally, let $G = (U, I; E)$ be a bipartite graph, where vertices represent users U and items I, and edges E represent the possible recommendations of items for users. A subset of edges M (i.e., $M \subseteq E$) is a *matching*, if all edges in M are pairwise non-adjacent, i.e., any two edges in M share neither a user vertex nor an item vertex. A vertex is *matched* if it is adjacent to an edge that is in the matching (otherwise, the vertex is unmatched). The *maximum matching* of a bipartite graph is a matching with the largest possible number of edges. The maximum bipartite matching algorithm (for top-1 recommendations) in Step 1 starts with matching $M = \varnothing$ and iteratively adds edges to M, until all users are matched or no new additional edge can be added. The edges to be iteratively added to M can be found by finding an *augmenting path* for M, which is a simple path (i.e., a sequence of alternating user and item vertices with no loops) that starts at an unmatched user and ends at an unmatched item, and its edges belong alternately to $E \backslash M$ and M. In other words, $P=(v_1, v_2, \ldots, v_{2n-1}, v_{2n})$ is an augmenting path where $v_{odd} \in U$, $v_{even} \in I$, v_1 is an unmatched user, v_{2n} is an unmatched item, $(v_{2k-1}, v_{2k}) \notin M$ where $k = \{1, \ldots, n\}$, and $(v_{2k+1}, v_{2k}) \in M$ where $k = \{1, 2, \ldots, n-1\}$. Let *edges*($P$) comprise the set of all edges of the augmenting path P. The key property of augmenting paths is that the symmetric set difference of M and *edges*(P), denoted as $M \Delta$ *edges*(P), always results in a matching with cardinality one more than the cardinality of M [5,14], i.e., if $M' = M \Delta$ *edges*(P), then $|M'| = |M| + 1$.

Thus, the notion of augmenting paths allows to find the maximum bipartite matching, by starting with matching $M = \varnothing$ and iteratively increasing its size one-by-one with each augmenting path, which we use in our algorithm for diversity maximization (Fig. 3). In particular, we adopt Hopcroft-Karp algorithm [20], which finds a maximal set of augmenting paths during every iteration, i.e., multiple augmenting paths in parallel for all unmatched vertices, thereby achieving a significant reduction in time complexity. This is a well-known technique and we encapsulate it in our algorithm by calling *Find_AugmentingPaths* subroutine (lines 6, 15 in Fig. 3); the implementation details for this subroutine can be found in [13].

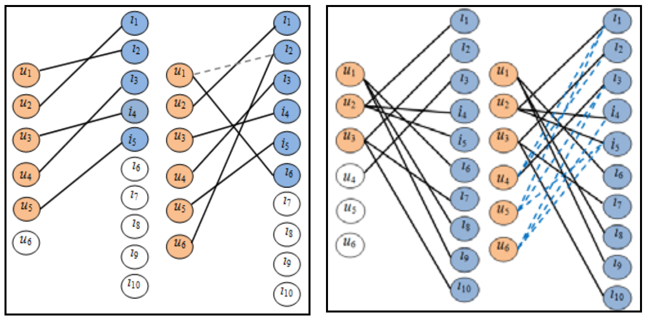
```
[Step 1] Find Maximum Diversity
// set of edges- items available for recommendation
1  $E := \{(u,i) \mid u \in U, i \in I, R^*(u, i) \in [T_F, T_{max}]\}$
2  $G := (U, I ; E)$               // bipartite graph with users, items, and edges
// initialize a set of unmatched users /items
3  $CU := U$ ; $CI := \{i \in I \mid u \in U, (u,i) \in E\}$
4  $M := \varnothing$                          // set of matched edges $M \subseteq E$
Maximum Bipartite Matching (Top-1 Task)
5  // find augmenting paths starting from unmatched user $v_1$ and ending with
   // unmatched item $v_{2n}$
6  $P$ := Find_AugmentingPaths($G$, $CU$, $CI$, $M$)
   // until all users are matched or no augmenting path exists
7  while ($CU \neq \varnothing$ and $P \neq \varnothing$)
8    for each $(v_1, v_2, \ldots, v_{2n-1}, v_{2n}) \in P$ do
9      edges:=$\{(v_{2k-1}, v_{2k}) \mid k:=1..n\} \cup \{(v_{2k+1}, v_{2k}) \mid k:=1..n-1\}$
       // flip the matched and unmatched edges
10     $M := M \Delta$ edges         // symmetric difference
11
12     Remove $v_1$ from $CU$     // one matching per user
13     Remove $v_{2n}$ from $CI$    // one matching per item
14   end for
15   $P$ := Find_AugmentingPaths ($G$, $CU$, $CI$, $M$)
16 end while
Extended Version for Top-N Recommendation Task
5  $\forall u \in U$, $uCnt[u]$:=0                    // num. of matches for each user
6  $P$ := Find_AugmentingPaths($G$, $CU$, $CI$, $M$)
7  while ($CU \neq \varnothing$ and $P \neq \varnothing$)
8    for each $(v_1, v_2, \ldots, v_{2n-1}, v_{2n}) \in P$ do
9      edges:=$\{(v_{2k-1}, v_{2k}) \mid k:=1..n\} \cup \{(v_{2k+1}, v_{2k}) \mid k:=1..n-1\}$
10     $M := M \Delta$ edges
11     $uCnt[v_1]$ := uCnt[$v_1$]+1             // N matchings per user
12     Remove $v_1$ from $CU$ if $uCnt[v_1] == N$
13     Remove $v_{2n}$ from $CI$                  // one matching per item
14   end for
15   $P$ := Find_AugmentingPaths ($G$, $CU$, $CI$, $M$)
16 end while

[Step 2] Complete Top-N Recommendations
17 for each $(u, i) \in M$ do
18   Add $i$ to $LN(u)$       // assign matchings as recommendations
19 end for
20 for each $u \in CU$ do   // fill the remaining items according to rank$_{Standard}$
21   Sort items $\{i \in I \mid R^*(u, i) \in [T_H, T_{max}]$ and $i \notin LN(u)\}$
22   Add top $(N - |LN(u)|)$ most highly predicted items to $LN(u)$
23 end for
```

Figure 3. Bipartite Matching Approach to Diversity Maximization.

The original bipartite matching algorithm for top-1 recommendations matches a user to only one item and excludes the matched user for the subsequent iterations, i.e., the user is removed from candidate user list CU (line 12 of Fig. 3). An extended version for top-N recommendations relaxes this rule by waiting to remove the user from CU until the same user is matched to N items. We also make the extended algorithm more efficient by allowing a single user to find up to N item matches in the first iteration (and not just a single match per iteration), which significantly reduces the number of subsequent iterations. However, similarly as with the max-flow approach, since an item can be recommended to only one user, some users may get fewer than N recommendations. Thus, in Step 2, for accuracy considerations, the most highly predicted items among remaining candidate items are chosen to fill the remaining top-N recommendations for all users. Note that this does not affect diversity (which is already guaranteed to be maximum).

Using the same example in Fig. 2, we illustrate the first step for top-1 recommendations, how the maximum bipartite matching algorithm can obtain the maximum diversity (Fig. 4a). This algorithm performs two iterations: (Iteration 1) finds all possible 1-edge augmenting paths between unmatched users and unmatched items, i.e., direct paths without any intermediate vertices; and (Iteration 2) finds multi-edge augmenting paths, each of which increases the cardinality of matching by one unit via alternating non-matched and matched edges in the paths. After the first iteration of Fig. 4a, the first five users are matched to one of their candidate items, but user u_6 is still unmatched because all her candidate items (i_2, i_3, i_4, i_5) are already matched to other users. The second iteration finds an augmenting path from unmatched user u_6 to unmatched item i_6, i.e., $P = (u_6, i_2, u_1, i_6)$ and $(u_6, i_2) \notin M$, $(u_1, i_2) \in M$, $(u_1, i_6) \notin M$. As a result, user u_6 is then matched to item i_2, and user u_1, previously matched to item i_2, is now matched to new item i_6, which leads to the maximum cardinality for this example (i.e., max aggregate diversity of 6 items), and the iterations for searching augmenting paths stop.



[Step1] Iteration 1 Iteration 2 [Step1] [Step2]

(a) Top-1 task (b) Top-3 task

Figure 4. Illustration of Graph-Based Approach.

On the other hand, in case of top-3 recommendations for the same example (Fig. 4b), while maximum diversity (i.e., 10) is reached in Step 1, three users (u_4, u_5, u_6) are matched to fewer than 3 items. Thus, as shown in Step 2 of Fig. 4b, the remaining top-3 recommendations are filled with the most highly predicted items among the items available for users.

Note that the sequence in which users and/or items are chosen to be evaluated in Fig. 3 may have implications on the runtime of the algorithm. E.g., finding more augmenting paths (and, therefore, a larger matching) in the first iteration may reduce the total number of iterations needed to reach the maximum matching. We found that applying a simple heuristic of first choosing users for matching who have the smallest number of remaining candidate items leads to substantial runtime improvements, because of the smaller likelihood that the items matched to those users can be replaced by other items, thus, reducing the number of iterations.

As mentioned earlier, for Step 1, we adopt the Hopcroft-Karp algorithm [20], which is known to be among the most efficient algorithms for maximum bipartite matching, having complexity of $O(E\sqrt{V})$, where E is the number of edges in the graph and V is the number of vertices on the left side of the graph (i.e., the number of users in our case) [13]. In a bipartite graph with m user vertices, n item vertices, and a maximum of mn edges, the complexity of the Hopcroft-Karp algorithm would be $O(mn\sqrt{m})$, and by adding the standard ranking approach for Step 2, the total complexity of the max flow based approach for top-1 recommendation tasks would be $O(mn\log n + mn\sqrt{m})$. For top-N recommendation tasks, we allow multiple edges from a single user vertex. We propose an efficient extension of bipartite matching algorithm for top-N recommendations, as discussed earlier; however, in the worst case, the top-N recommendation task can be treated as top-1 task with Nm users and, correspondingly, Nmn edges. Even this worst-case extension for top-N recommendations does not change the complexity, i.e., $O(Nmn\sqrt{Nm}) = O(mn\sqrt{m})$, assuming N (i.e., the number of recommendations provided to each user) is a relatively small, bounded constant. Therefore, this graph-based approach is more complex than the re-ranking heuristic, which had worst case complexity of $O(mn\log n)$, as mentioned earlier.

4. EMPIRICAL RESULTS

In our experimental evaluation, we used two movie rating datasets: MovieLens (data file available at grouplens.org) and Netflix (used for Netflix Prize competition). Each dataset is pre-processed to include users and movies with significant rating histories, which makes it possible to have a large number of highly predicted items available for recommendations to each user, thus, potentially making the diversity maximization task more challenging. The basic statistical information of the resulting datasets is as follows. MovieLens dataset has 775,176 ratings with 2,830 users and 1,919 items (i.e., 14.27% sparsity), and Netflix dataset has 1,067,999 ratings with 3,333 users and 2,091 items (i.e., 15.32% sparsity). For each dataset, we learn from all of the known ratings and predict the unknown ratings (85.73% of the whole user-item matrix in the MovieLens dataset and 84.68% in the Netflix dataset). As discussed earlier, we use three popular collaborative filtering techniques (user-based, item-based, and matrix factorization CF techniques), and top-N (N=1, 5, 10) items are recommended for each user.

We predict unknown ratings based on all known ratings, where a relatively large number of highly-predicted (i.e., with the predicted rating value above 3.5) candidate items are available for all users (typically around 500-800 items for each user). Fig. 5 presents a number of representative results obtained from the empirical evaluation, which shows not only the accuracy and diversity capabilities of the proposed approach in terms of top-N recommendation, but also compares it with two baseline techniques that re-rank the candidate items by their reverse predicted rating values [3] and at random, as well as with the standard recommendation technique. As expected, the standard recommendation technique (i.e., recommending items with highest predicted ratings) represents the most accurate, but very non-diverse set of recommendations. In Fig. 5, the representative accuracy-diversity curves for the baseline random and re-ranking techniques and for graph-based approach were obtained by using different ranking and flow-rating thresholds (3.5, 3.6, …, 5).

One notable finding is that, while the simple re-ranking technique shows the same or slightly better results than the random approach, the proposed graph-based approach is able to obtain substantial diversity improvements at the given level of accuracy, compared to the two baseline techniques, across all experiments including different datasets, different recommendation techniques, and different number of recommendations (N = 1, 5, 10).

Another notable result is that, as N increases, significant diversity improvements can be obtained with increasingly smaller sacrifices to recommendation accuracy. For example, in top-1 recommendation tasks, the graph-based approach was able to obtain the maximum possible diversity with a decrease of about 0.5 (on scale 1-5) in an average prediction. However, for top-5 tasks the accuracy decrease needed for maximum diversity was about 0.1, and for top-10 tasks only about 0.05. Table 1 further illustrates this point by showing the diversity gains of random, re-ranking, and graph-based approaches at three different accuracy loss levels (0.1 for top-1 tasks, 0.05 for top-5 tasks, 0.01 for top-10 tasks). In summary, the proposed graph-based approach was able to consistently provide substantial diversity improvements for all traditional recommendation algorithms (user-based, item-based, and matrix factorization CF) on different real-world recommendation datasets.

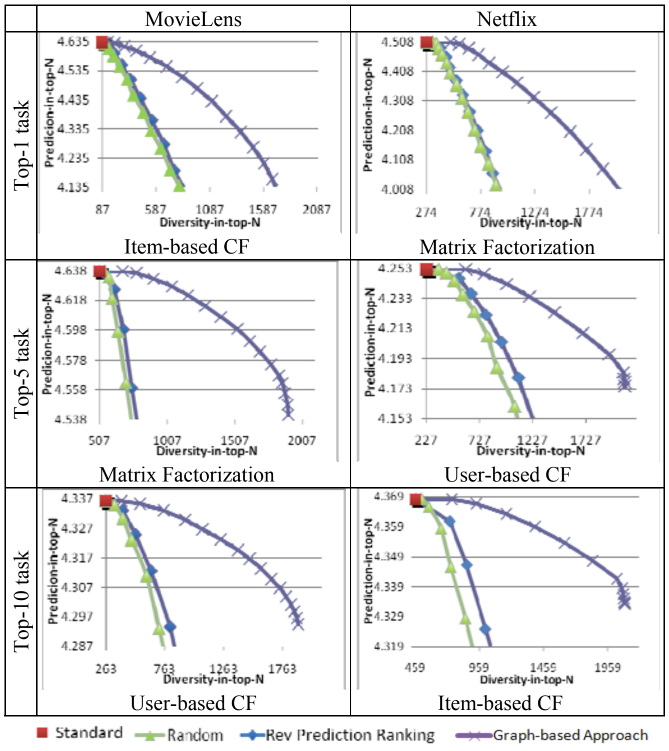


Figure 5. Performance of Ranking and Optimization Approaches

As discussed earlier, the performance improvements for the proposed technique come at the cost of computational complexity, which can become an issue as the data size increases. To complement the earlier discussion on the theoretical computational complexity, here we report how the data size affects the actual runtime. We vary the size of data by changing the number of candidate items that are available for recommendation to each user. For example, for the datasets used in our experiments we treated all items that were predicted above rating threshold T_H = 3.5 as potential candidates for recommendation. By increasing this threshold we can eliminate some candidate items across all users, thus, obtaining smaller datasets. Following this approach, we generated six datasets D_1, ..., D_6 of increasing size from MovieLens dataset by using different rating thresholds (D_1 for $T_H = 4.5$, D_2 for 4.3, D_3 for 4.1, …, D_6 for 3.5), as indicated in Fig. 6a.

We measured the runtime of the two algorithms (i.e., the proposed approach and the re-ranking approach) on the same computer. The obtained results are consistent across different recommendation algorithms, different datasets, and top-N tasks (for different N values). Fig. 6b illustrates the general trends by presenting the example runtimes of the simple recommendation re-ranking and the graph-based approach on the MovieLens dataset, for generating diverse top-1 recommendations using item-based CF technique. As expected, as the data size increases, the simple re-ranking heuristic demonstrates good scalability, while the more complex graph-based approach requires increasingly more time (while also generating better recommendation outcomes, as discussed earlier). We do observe that, for our medium-size recommendation setting (with approx. 3,000 users and 2,000 items), the proposed approach demonstrated good computational performance; even running it on the largest dataset (D_6) took less than 1.5 minutes.

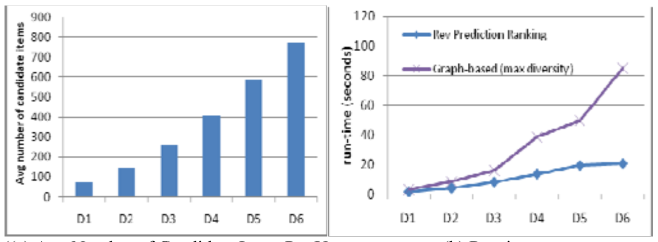
Table 1. Diversity Gains at the Given Accuracy Level

	User-based CF	Item-based CF	MF
Top-1 recommendation task (Accuracy level: Standard – 0.1)			
Standard	98, acc=4.40	87, acc=4.63	247, acc=4.73
Random	368.7 (276.2%)	272.6 (213.3%)	390.3 (58.0%)
Re-Ranking	409.1 (317.4%)	300.8 (245.7%)	412.3 (66.9%)
Graph-Based	826.0 (742.9%)	748.6 (760.4%)	927.5 (275.5%)
Top-5 recommendation task (Accuracy level: Standard – 0.05)			
Standard	190, acc=4.36	200, acc=4.57	507, acc=4.64
Random	581.6 (206.1%)	385.1 (92.6%)	659.3 (30.0%)
Re-Ranking	648.1 (241.1%)	424.5 (112.3%)	698.1 (37.7%)
Graph-Based	1562.9 (722.6%)	1415.9 (607.9%)	1647.7 (225.0%)
Top-10 recommendation task (Accuracy level: Standard – 0.01)			
Standard	263, acc=4.34	279, acc=4.53	667, acc=4.58
Random	448.6 (70.6%)	354.9 (27.2%)	745.0 (11.7%)
Re-Ranking	497.4 (89.1%)	385.7 (38.3%)	794.3 (19.1%)
Graph-Based	1107.0 (320.9%)	978.7 (250.8%)	1408.2 (111.1%)

(a) MovieLens data

	User-based CF	Item-based CF	MF
Top-1 recommendation task (Accuracy level: Standard – 0.1)			
Standard	67, acc=4.31	142, acc=4.51	274, acc=4.51
Random	416.6 (521.8%)	379.1 (167.0%)	484.1 (76.7%)
Re-Ranking	417.5 (523.1%)	420.2 (195.9%)	505.4 (84.5%)
Graph-Based	764.5 (1041.1%)	842.6 (493.4%)	967.8 (253.2%)
Top-5 recommendation task (Accuracy level: Standard – 0.05)			
Standard	227, acc=4.25	335, acc=4.42	561, acc=4.43
Random	822.5 (262.3%)	671.5 (100.4%)	904.8 (61.3%)
Re-Ranking	943.4 (315.6%)	754.4 (125.2%)	966.8 (72.3%)
Graph-Based	1829.1 (705.8%)	1795.8 (436.1%)	2000.9 (256.7%)
Top-10 recommendation task (Accuracy level: Standard – 0.01)			
Standard	341, acc=4.21	459, acc=4.37	771, acc=4.39
Random	736.1 (115.9%)	655.7 (42.9%)	1046.8 (35.8%)
Re-Ranking	876.6 (157.1%)	750.0 (63.4%)	1193.2 (54.8%)
Graph-Based	1528.9 (348.3%)	1426.3 (210.7%)	2456.1 (218.6%)

(b) Netflix data



((a) Avg Number of Candidate Items Per User (b) Runtime

MovieLens dataset, Item-based CF, Top-1 recommendation task

Figure 6. Different Datasets and Algorithmic Runtime

5. CONCLUSIONS AND FUTURE WORK

Recommendation diversity recently has attracted attention as an important aspect in evaluating the quality of recommendations. Traditional recommender systems typically recommend the top-N most highly predicted items for each user, thereby providing good predictive accuracy, but performing poorly with respect to recommendation diversity. This paper extends prior work by developing a more sophisticated graph-theoretic approach that models the diversity maximization problem as a network flow maximization or bipartite matching maximization problems and provides significant advantages over the recommendation re-ranking approaches in terms of the accuracy/diversity tradeoff.

The proposed optimization approaches have been designed specifically for the diversity-in-top-*N* metric, i.e., the number of distinct items among top-*N* recommendations. The extension of these approaches to more sophisticated diversity metrics, including relative long-tail metrics such as Gini coefficient [17] and the long-tail shape parameter such as the slope of the log-linear relationship between popularity and recommendations, represent a promising direction for future research. Another interesting and important direction would be to investigate whether the use of the diversity-maximizing recommendation algorithms can truly lead to an increase in sales diversity and user satisfaction. In particular, as discussed in recent research [12,25], it would be valuable to examine the impact of recommendations on long-tail phenomena in different categories of users and products and possibly propose different algorithms based on the appropriate categorization. We believe that this work provides insights into developing new recommendation techniques that can consider multiple aspects of recommendation quality, going beyond using just the accuracy measures.

6. ACKNOWLEDGMENTS

The research reported in this paper was supported in part by the US National Science Foundation CAREER Grant IIS-0546443.

7. REFERENCES

[1] Adomavicius, G., A. Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE TKDE* 17:6 734-749.

[2] Adomavicius, G., Y. Kwon. 2009. Toward More Diverse Recommendations: Item Re-Ranking Methods for Recommender Systems. *Proc. of the 19th Workshop on Information Technologies and Systems*.

[3] Adomavicius, G., Y. Kwon. 2011. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering*. Forthcoming.

[4] Aggarwal, C.C., J.L. Wolf, K.L. Wu, P.S. Yu. 1999. Horting Hatches An Egg: A New Graph-Theoretic Approach to Collaborative Filtering. *Proc. of the 5th ACM SIGKDD Conf. on Knowledge Discovery and Data Mining* (*KDD'99*). 201-212.

[5] Ahuja, R. K., T. L. Magnanti, J. B. Orlin, 1993. *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice-Hall.

[6] Anderson, C. 2006. *The Long Tail*. New York: Hyperion.

[7] Balabanovic, M., Y. Shoham. 1997. Fab: Content-Based, Collaborative Recommendation. *Comm. of the ACM* 40:3 66-72.

[8] Bradley, K., B. Smyth. 2001. Improving Recommendation Diversity. *Proc. of the 12th Irish Conf. on Artif. Intelligence and Cognitive Sci.*

[9] Breese, S., D. Heckerman, C. Kadie. 1998. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proc. of the 14th Conf. on Uncertainty in Artificial Intelligence*.

[10] Brynjolfsson, E., M.D. Smith, Y.J. Hu. 2003. Consumer Surplus in the Digital Economy: Estimating the value of increased product variety at online booksellers. *Management Sci.* 49:11 1580-1596.

[11] Brynjolfsson, E., Y.J. Hu, D. Simester. 2007. Goodbye Pareto Principle, Hello Long Tail: The Effect of Search Costs on the Concentration of Product Sales. *NET Institute*.

[12] Brynjolfsson, E., Y.J. Hu, M.D. Smith. 2010. Long Tails vs. Superstars: The Effect of Information Technology on Product Variety and Sales Concentration Patterns. *Information Systems Research* 21:4 736-347.

[13] Burkard R., M. Dell'Amico, S. Martello. 2009. Assignment Problems. *Society for Industrial and Applied Mathematics* (*SIAM*).

[14] Cormen, T.H., C.E. Leiserson, R.L. Rivest, C. Stein. 2001. *Introduction to Algorithms*,MIT Press.

[15] Fleder, D., K. Hosanagar. 2009. Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity, *Management Science* 55:5 697-712.

[16] Funk, S. 2006. Netflix Update: Try This At Home. http://sifter.org/˜simon/journal/20061211.html.

[17] Gini, C. 1921. Measurement of Inequality and Incomes. *Economic Journal* 31 124-126.

[18] Goldstein, D.G., D.C. Goldstein. 2006. Profiting from the Long Tail. *Harvard Business Review*, Jun 2010.

[19] Herlocker, J.L., J.A. Konstan, L.G. Terveen, J. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems* 22:1 5-53.

[20] Hopcroft, J.E., R.M. Karp. 1973. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs, *SIAM J. on Computing* 2:4 225-231.

[21] Hu, R., P. Pu. 2011. Enhancing Recommendation Diversity with Organization Interfaces. *Proc. of the 16th Int'l Conf. on Intelligent User Interfaces* (*IUI '11*). 347-350.

[22] Huang, Z., D. Zeng, H. Chen. 2007. Analyzing Consumer-product Graphs: Empirical Findings and Applications in Recommender Systems. *Management Science* 53:7 1146-1164.

[23] Kim, H.K., J.K. Kim, Y. Ryu. 2010. A Local Scoring Model for Recommendation. *Proc. of the 20th Workshop on Information Technologies and Systems* (*WITS'10*).

[24] Koren, Y., R. Bell, C. Volinsky. 2009. Matrix Factorization Techniques For Recommender Systems. *IEEE Computer Society,* 42 30-37.

[25] Lee, J., J.N. Lee, H. Shin. 2011. The Long Tail or the Short Tail: The Category-Specific Impact of eWOM on Sales Distributions. *Decision Support Systems* 51:3 466-479.

[26] Leonard, D. 2010. Tech Entrepreneur Peter Gabriel Knows What You Want. *Business Week*, April.

[27] Levy, M., K. Bosteels. 2010. Music Recommendation and the Long Tail. Workshop on Music Recommendation and Discovery, *ACM Intl. Conf. on Recommender Systems*.

[28] Liu, J., M. Shang, D. Chen. 2009. Personal Recommendation Based on Weighted Bipartite Networks. *Proc. of the 6th Intl. Conf. on Fuzzy Systems and Knowledge Discovery* 134-137.

[29] McNee, S.M., J. Riedl, J.A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics have hurt Recommender Systems. *Conf. on Human Factors in Computing Systems* 1097-1101.

[30] McSherry, D. 2002. Diversity-Conscious Retrieval. *Proc. of the 6th European Conf. on Advances in Case-Based Reasoning* 219-233.

[31] Oestreicher-Singer, G., A. Sundararajan. 2011. Recommendation Networks and the Long Tail of Electronic. *MIS Quarterly*. Forthcoming.

[32] Park, Y.J., A. Tuzhilin. 2008. The Long Tail of Recommender Systems and How to Leverage It. *Proc. of the 2nd ACM Conf. on Recommender Systems* 11-18.

[33] Sarwar, B., G. Karypis, J.A. Konstan, J. Riedl. 2001. Item-Based Collaborative Filtering Recommendation Algorithms. *Proc. of the 10th Intl. World Wide Web Conf.*

[34] Shani G., A. Gunawardana. 2011. Evaluating Recommendation Systems, in P. B. Kantor, F. Ricci, L. Rokach, B. Shapira (Eds.), *Recommender Systems Handbook: A Complete Guide for Research Scientists and Practitioners*, Chapter 8, Springer.

[35] Smyth, B., P. McClave. 2001. Similarity vs. Diversity. *Proc. of the 4th Intl. Conf. on Case-Based Reasoning*.

[36] Thompson, C. 2008. If You Liked This, You're Sure to Love That. *The New York Times*. Nov 2008. http://www.nytimes.com/ 2008/11/23/ magazine/23Netflix-t.html.

[37] Zhang, M. 2009. Enhancing Diversity in Top-*N* Recommendation. *Proc. of the 3rd ACM Conf. on Recommender Systems* 397-400.

[38] Zhang, M., N. Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. *Proc. of the 2nd ACM Conf. on Recommender Systems* 123-130.

[39] Ziegler, C.N., S.M. McNee, J.A. Konstan, G. Lausen. 2005. Improving Recommendation Lists Through Topic Diversification, *Proc. of the 14th Intl. World Wide Web Conf.* 22-32.