Recommender Systems With Non-Binary Grades

Yossi Azar[*]
School of Computer Science
Tel Aviv University
Tel Aviv 69978
Israel
azar@tau.ac.il

Aviv Nisgav
School of Electrical Engineering
Tel Aviv University
Tel Aviv 69978, Israel
avivns@eng.tau.ac.il

Boaz Patt-Shamir[†]
School of Electrical Engineering
Tel Aviv University
Tel Aviv 69978, Israel
boaz@eng.tau.ac.il

ABSTRACT

We consider the interactive model of recommender systems, in which users are asked about just a few of their preferences, and in return the system outputs an approximation of all their preferences. The measure of performance is the *probe complexity* of the algorithm, defined to be the maximal number of answers any user should provide (probe complexity typically depends inversely on the number of users with similar preferences and on the quality of the desired approximation). Previous interactive recommendation algorithms assume that user preferences are binary, meaning that each object is either "liked" or "disliked" by each user. In this paper we consider the general case in which users may have a more refined scale of preference, namely more than two possible grades. We show how to reduce the non-binary case to the binary one, proving the following results. For discrete grades with s possible values, we give a simple deterministic reduction that preserves the approximation properties of the binary algorithm at the cost of increasing probe complexity by factor s. Our main result is for the general case, where we assume that user grades are arbitrary real numbers. For this case we present an algorithm that preserves the approximation properties of the binary algorithm while incurring only polylogarithmic overhead.

1. INTRODUCTION

Arguably, helping people identify objects they may desire has always been a central part of civilization, but recently, with the advent of the so-called "information age" it has taken new forms. On one hand, many more objects are available (because many objects are digital and can be obtained by a click of a button), and on the other hand, sophisticated algorithms can provide the users with relatively-intelligent advices when seeking something, where typical goals may be movies to watch, interesting articles to read, authoritative web pages, good restaurants, similar users in a social network etc.

One of the main ways to find such objects is a *recommender system*, and in particular *collaborative filtering* (see, e.g., [14, 15] for some early work). The basic idea in collaborative filtering is that the system collects data about user preferences, and tries to tailor individual recommendation to each user based past choices of that user and the choices of other users.

Much of the current research in collaborative filtering considers the following model. There is a large dataset that contains all past choices of users (be it purchase history, or, say, movie grades), and the goal is to predict the way a user would grade an object she did not examine yet.[1] The problem with this approach is that it ignores the existence of feedback in the model: Assuming that the recommender system indeed affects user choices, the dataset is biased toward objects recommended by the system, and does not reflect the "true" preference of the users. It follows that analytically, such a system makes sense only for a single recommendation to each user (this way there's no feedback), or under the implicit assumption that users actually choose objects based on some external recommendations (making the feedback effect weak enough to ignore). Obviously, these assumptions are unsatisfactory in most cases.

This gap is bridged by the *interactive recommender system* model [7, 4]. In this model, it is assumed that the system can observe the user's reaction to recommendations and act on it. More specifically, the model is that the system proposes an object to the user, and the user, in response, informs the system (perhaps implicitly) of her grade for that object. The system uses this input when making future recommendations. It is usually assumed that the system starts out with no knowledge at all about user grades. (Note that in this case, the system is more likely to propose controversial objects to the user at the beginning, because the way users grade them conveys more information.)

[*]Supported in part by the Israel Science Foundation (grant No. 1404/10).

[†]Supported in part by the Israel Science Foundation (grant 1372/09) and by Israel Ministry of Science and Technology. Research partly done while visiting MIT CSAIL.

[1]It may be interesting to note that in the Netflix Challenge [6], the algorithm is asked to predict the grade users have already gave to some objects.

The model. Assume that there is a set of n users and a set of m objects, and that each user has a grade for each object. The grades are initially unknown, and the goal is to find them, by means of asking users for their grades for certain objects (a.k.a. "probing"). The performance measure we are interested in is the *probe complexity*, defined as the maximal number of grades any user is asked to report. The strongest possible goal for a recommender algorithm is to reconstruct all user preferences. Obviously, reconstructing preferences of an esoteric user may require many more probes than reconstructing the preferences shared by many users. We therefore introduce two parameters, $0 < \alpha \leq 1$ and $D \geq 0$ and assume that the following holds: For each user u there is a set of αn other users whose preferences are at most distance D away from the preferences of u, under some appropriate metric. It is not hard to see that in order to reconstruct user preferences to within distance D, users need to examine $\Omega(m/\alpha n)$ objects on average: Intuitively, the αn similar users need to examine all m objects between them.

Previous work. The goal of the algorithms presented in [7, 4] is to find a single "good object" for the users. Since being a "good" object is a binary predicate, these algorithms effectively assume binary grades. In [2], full preference reconstruction algorithms are presented for arbitrary grades, but the algorithms are applicable only under the assumption that $D = 0$, namely reconstruction is guaranteed to work only if αn users share *exactly* the same preferences. A general reconstruction result is presented in [1], where any $0 \leq D \leq n/\log n$ is allowed. However, the latter algorithm relies strongly on the assumption that preference grades are binary. The metric used in [1] is the Hamming distance, namely the distance between two users is the number of objects on which their grades differ. An improved reconstruction algorithm is presented [8], where the presence of some Byzantine users can be tolerated.

Our Contribution. In this paper we extend this line of work by considering real-valued grades (normalized to the range $[0, 1]$). The metric we use to measure distance between preference vectors is L_1, i.e., the distance between two users is the sum, over all objects, of difference in their grades.

Our algorithms use algorithms for the binary model as a black box, so the main contribution of the paper is showing how to reduce the continuous-scale grades recommendation model to the binary one. Let us note upfront that simple discretization approaches do not work. For example, suppose we round each grade to 0 or 1 (i.e., round$(x) = \lfloor x + \frac{1}{2} \rfloor$), and apply a binary algorithm to the rounded grades. Such a rounding performs miserably in the case of a cluster of αn users whose preferences have small diameter D, but all their grades are uniformly distributed around $\frac{1}{2}$. Intuitively, the rounding splits this tight cluster into many sets, and places them very far from each other. More precisely, consider the possible parameters α and D to be supplied to the binary algorithm: If we insist on keeping all αn users as a cluster, then the binary algorithm must use distance parameter $\Omega(m)$, and if we insist on keeping D as the distance parameter, the binary algorithm must use $\alpha/2^{m-D}$ as the number of neighbors. Both alternatives give us trivial parameters, because the distance can never be greater than m, and $\alpha \geq 1/n$ always. It is easy to see that *any* rounding threshold will suffer from the same problem.

Thus, the main technical tool we introduce in this paper is a way to do the rounding while increasing D and decreasing α by only a constant factor, and increasing the probe complexity of the underlying binary algorithm by only polylogarithmic factor. Roughly speaking, we show the following result. Suppose that for each user there are at least αn users whose preferences are at most D away, where $D > \epsilon$ for some constant $\epsilon > 0$ (the case of $D = 0$ is much simpler: see [2]). Then the preferences of all users can be reconstructed, with high probability, to within $O(D)$, using $O(\frac{m}{\alpha n} \log^{9/2} n)$ probes per user. (Note that the $\frac{m}{\alpha n}$ factor is unavoidable: if αn users agree on $m - D$ objects, then each of the $m - D$ objects must be probed by at least one of the αn similar users.) The precise statement is slightly more involved: See Theorem 4.1.

As an intermediate output, our algorithm produces estimates of "inter-user distances," i.e., each user can identify (approximately) what is her distance from any other user (Theorem 4.7). We believe that this result may be of independent interest, say in the context of social networks [12].

Another result we present is a linear-reduction algorithm for multiple evenly-spaced discrete grades (Theorem 3.1). In this case, to handle $s + 1$ possible grades, the cost of the algorithm grows by a factor s, compared to the cost of a binary algorithm. From the theoretical viewpoint the linear reduction is very simple. However, we include it here because in many practical situations the grades are discrete and there are only a few of them, in which case the linear reduction is more efficient than the algorithm for the general case. More formally, assuming $m = \Theta(n)$, the linear reduction has better probe complexity than the general algorithm if the number of grades s satisfies $s \leq \log n/D$.

Related Work. Our model is closely related to models of recommender systems [5, 7, 4], in which the users grades are structured as a user-product matrix where each entry is a user opinion on an object and the goal is to produce an estimate of this matrix. Some variants of recommender systems assume partial information about the matrix, which is presented as known entries, and the task of the algorithm is to predict unknown entries based on the given data. Other variants assume that all entries are unknown, and the task of the algorithm is to instruct the users which products to try (thereby revealing some entries of the matrix) so that the algorithm can recommend a good product more effectively.

In the former model, where a partial matrix is given, it is common to assume a linear generative model for user's opinion vectors and apply algebraic techniques such as principal component analysis [9] or singular value decomposition [16]. Papadimitriou et al. [13] and Azar et al. [5] rigorously prove conditions under which SVD is effective. Other generative user models that were considered include simple Markov chain models [10, 11], where users randomly select their "type," and each type is a probability distribution over the objects.

Drineas et al. [7] were the first to propose a competitive model, where the algorithm directs the users which products

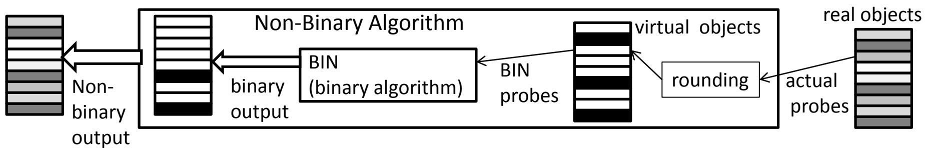

Figure 1: Framework for reducing real value algorithms to binary value algorithms

to try and the results of the tries are fed back to the algorithm. In [4] it was shown that in this model, a user sharing its preference with at least α fraction of the users ($D = 0$ in our terms), can find a product he likes in $O\left(\left\lceil \frac{m}{n} \right\rceil \log n/\alpha\right)$ tries. Later, in the same model Awerbuch et al. [2] reconstruct the users preference for all objects with similar probe complexity and show this is a lower bound.

In case users grade objects in a binary fashion (e.g. "like" or "dislike") Alon et al. [1] present an algorithm where user preference vectors are computed approximately using similar users in a competitive way. In that algorithm, if for every user there is a set of αn other users whose preferences are at most Hamming-distance $\log(n)$ away, then the number of queries by each user is $O\left(\left\lceil \frac{m}{n} \right\rceil \log^{3.5} n/\alpha\right)$.

Awerbuch et al. [3] study recommendations algorithms in an asynchronous model, where an adversarial (oblivious) schedule determines which user will make the next probe, and the algorithm may only say which object should that user probe.

Organization. In Section 2 we formalize the model, define some notation and present the overall framework. In Section 3 we present a reduction of the discrete model. In Section 4 we present our main result, an algorithm for the continuous model.

2. PRELIMINARIES

Basic concepts and notation. There are n *users* and m *objects*. Each user has a *grade* for each object (the grades are initially unknown). In the *discrete* case, there are $s+1$ grades, assumed to be the set $\left\{0, \frac{1}{s}, \frac{2}{s}, \ldots, 1\right\}$. In the *continuous* case, grades are real numbers in the interval $[0,1]$. We sometimes refer to the complete set of user grades as the *preference matrix* A, of dimension $n \times m$. A_{ij} is the grade of user i to object j, and row i, denoted A_i, is called user i's *preference vector*. The *distance* between two preference vectors A_i and $A_{i'}$ is the L_1 norm of their difference, i.e., $\text{dist}(A_i, A_{i'}) \stackrel{\text{def}}{=} \sum_{j=1}^{m} |A_{ij} - A_{i'j}|$. A matrix $A_{n\times m}$ is called (α, D)*-similar* if for each row A_i it holds that $|\{i' : \text{dist}(A_i, A_{i'}) \le D\}| \ge \alpha n$, i.e., there are at least αn row vectors whose distance from A_i is at most D.

The recommendation problem statement. The input is an (α, D)-similar matrix $A_{n\times m}$, of which only α, D, and the dimensions n and m are initially known. The output is a matrix $\hat{A}_{n\times m}$, which is an estimate of A. The *approximation factor* of the output is $\max\left\{\frac{\text{dist}(A_i, \hat{A}_i)}{D} : 1 \le i \le n\right\}$.

Computational model. Algorithms proceed in synchronous *rounds*. In each round, each user is asked to reveal its grade for at most one object (a "probe"). (In a distributed model, the results of probes are published on a public "billboard," i.e., they are available to all users.) The maximal number of grades any user provides is the *probe complexity* of the algorithm. Trivially, the recommendation problem can be solved without errors in probe complexity m. It is also not difficult to see that $\Omega(m/\alpha n)$ probe complexity is necessary to produce $O(1)$-approximation of (α, D)-similar matrix.

Binary Algorithms. We shall assume that we have at our disposal an algorithm denoted BIN that solves the recommendation problem when all grades are binary. More precisely, when run on an (α, D)-similar binary matrix of dimension $n \times m$, BIN produces, at the cost of $T_{\text{BIN}}(n, m, \alpha, D)$ probe complexity, a γ_{BIN}-approximation, for some constant $\gamma_{\text{BIN}} \ge 0$. The binary algorithms we use are typically randomized; we assume that they are high-probability Monte Carlo algorithms, namely they succeed with probability $1 - n^{-c}$ for any desired constant c. Since in our algorithms, the number of invocations of BIN is always polynomial in n, we may apply the Union Bound to deduce that w.h.p., all invocations of BIN are successful.

Let us now explain how invocations of BIN are carried out. The problem is that we cannot apply BIN directly when preferences are not binary. We use the following natural on-line reduction framework (see Figure 1). The binary algorithm is presented with the same set of users, but with different set of objects which we call *virtual objects*, where each virtual object corresponds to exactly one real object (but one real object may correspond to none or several virtual objects). Whenever BIN asks user i to probe virtual object j, the non-binary algorithm asks user i to probe the real object corresponding to j, and presents BIN with a binary value derived from the real grade by applying some rounding function that depends on the non-binary algorithm.

Our algorithms solve the continuous-scale problem using a black-box implementation of a binary algorithm. For concreteness, we use the following result from [1].

THEOREM 2.1. *Given an $(\alpha, \log n)$-similar $n \times m$ binary matrix B, algorithm Small_Radius from [1] reconstructs B with probe complexity $O\left(\frac{1}{\alpha}\left\lceil \frac{m}{n} \right\rceil \log^{7/2} n\right)$ and approximation factor $O(1)$.*

3. THE DISCRETE CASE: A LINEAR REDUCTION

In this section we present an algorithm for the discrete preference case, where there are $s+1$ possible grades. This reduction is very simple, but it might be practical for small values of s, and it serves as a gentle warm-up to our main result for the continuous model.

We assume that the grades are the set $S = \left\{0, \frac{1}{s}, \frac{2}{s}, \ldots, 1\right\}$, where $s = |S| - 1$. The algorithm works as follows (pseudo-

Algorithm 1 DISCRETE_RECONST$(A_{n\times m}, \alpha, D)$

```
(1) Define matrix $B_{n\times sm}$ for $i \in [1,n]$, $j \in [1,m]$ and
    $k \in [1,s]$ by $B_{i,(j-1)s+k} = \begin{cases} 1 & \text{If } A_{i,j} > \frac{k-1}{s} \\ 0 & \text{Otherwise} \end{cases}$
(2) Invoke BIN$(B, \alpha, sD)$, and let $\hat{B}$ denote its output.
(3) Find $l_{i,j} \in [0,s]$ that minimizes
        $\sum_{l=1}^{l_{i,j}} \left(1 - \hat{B}_{i,(j-1)s+l}\right) + \sum_{l=l_{i,j}+1}^{s} \hat{B}_{i,(j-1)s+l}$
    for $i \in [1,n]$ and $j \in [1,m]$.
(4) Let $L_{n\times m}$ be the matrix over $S$ defined by $L_{i,j} = \frac{l_{i,j}}{s}$.
    Output $L$.
```

code is provided in Algorithm 1 below.) For each real object j define s binary virtual objects $j_1, \ldots, j_s$. Whenever BIN asks user i to probe some virtual object j_ℓ, we apply the following rounding procedure. If none of the virtual objects corresponding to j was probed by user i so far, then we ask i to probe (real) object j, and obtain the value of A_{ij}. Suppose the value is such that $\frac{k-1}{s} < A_{ij} \le \frac{k}{s}$ for some $1 \le k \le s$. Then we set the values (for user i) of virtual objects $j_1, \ldots j_k$ to 1, and the values of $j_{k+1}, \ldots, j_s$ to 0 (intuitively, the grades are coded in unary). Finally, if j was already probed in the past by i, we can use the known A_{ij} value to return a virtual probe value to BIN. This concludes the description of the rounding procedure.

Note that the output of BIN is a binary matrix, which does not necessarily represent valid encoding of grades the way the input was encoded, because it is only an approximation. In the algorithm, we choose the closest (under Hamming distance) valid encoding of a preference vector.

We have the following result, assuming that BIN succeeds with probability $1 - n^{-\Omega(1)}$.

THEOREM 3.1. *With probability $1 - n^{-\Omega(1)}$, for any user i, Algorithm* DISCRETE_RECONST *reconstructs the preference vector of user i with probe complexity $T_{\text{BIN}}(n, sm, \alpha, sD)$ and approximation factor $O(\gamma_{\text{BIN}})$.*

Proof: Let i, i' be any two users. Then by Step 1 of the algorithm,

$$\begin{aligned}\text{dist}(B_i, B_{i'}) &= \sum_{j=1}^{m}\sum_{k=1}^{s} \left|B_{i,(j-1)s+k} - B_{i',(j-1)s+k}\right| \\ &= \sum_{j=1}^{m} s\left|A_{i,j} - A_{i',j}\right| \\ &= s \cdot \text{dist}(A_i, A_{i'})\ ,\end{aligned}$$

and therefore, by the properties of BIN, we get that its invocation in Step 2 yields $\hat{B}$ such that $\text{dist}(B_i, \hat{B}_i) \le \gamma_{\text{BIN}} \cdot sD$ for any user i.

Next, for any i, j, let $l^*_{i,j} = sA_{i,j}$. Let $\hat{B}^{i,j}$ denote the vector $\hat{B}_{i,(j-1)s+1}, \ldots, \hat{B}_{i,js}$, and let U_l denote the binary vector of s bits with the first $l-1$ bits set to 1. By Step 3 of the algorithm, $\text{dist}(U_{l_{i,j}}, \hat{B}^{i,j}) \le \text{dist}(U_{l^*_{i,j}}, \hat{B}^{i,j})$. Using the triangle inequality we obtain

$$\begin{aligned}\|L_i - A_i\| &= \sum_{j=1}^{m} \left|S_{l_{i,j}} - S_{l^*_{i,j}}\right| \\ &= \frac{1}{s}\sum_{j=1}^{m} \left\|U_{l_{i,j}} - U_{l^*_{i,j}}\right\| \\ &\le \frac{2}{s}\sum_{j=1}^{m} \left\|U_{l^*_{i,j}} - \hat{B}^{i,j}\right\| \\ &= \frac{2}{s}\left\|B_i - \hat{B}_i\right\| \ = \ 2\gamma D\,.\end{aligned}$$

which proves the approximation claim. The probe complexity follows directly from the fact that in Algorithm 1 probes are done only while evaluating BIN(B, α, sD) in Step 2. ∎

We note that the probe complexity of BIN is typically linear in the number of objects, and hence Algorithm DISCRETE_RECONST guarantees no degradation in approximation, but the cost grows linearly with the number of discrete grades. In particular, when compared with the general construction presented in the following section, Algorithm DISCRETE_RECONST is superior in terms of probe complexity when the number of grades $s+1$ satisfies $s = O(\log n/D)$.

4. THE CONTINUOUS CASE

In this section we present our main result: an algorithm for the case grades are arbitrary bounded real numbers. W.l.o.g., we may assume in this case that the grades are in the unit interval $[0,1]$.

The most natural idea is to reduce the continuous case to the discrete case. But similarly to the argument mentioned in the introduction, simplistic discretization approaches fail. If we use resolution of $1/s$, then discretization may end up in adding an overall m/s error. In other words, there are inputs where users are nicely concentrated in the continuous model, but after discretization they will end up with linear distance between them (if we use resolution of $O(1/m)$, then Algorithm DISCRETE_RECONST results in increasing the probe complexity by a factor of m, which is trivial to achieve).

The intuition behind our algorithm is as follows. While any discretization has bad instances in which clusters are split along discretization borders, a random discretization is likely to avoid many of these bad events. If we use many discretizations, many of them should not split many clusters. It turns out that this intuition is roughly correct, but it is not readily clear how to use this random discretizations. Our solution is to construct a metric embedding from L_1-norm in $[0,1]^m$ to Hamming distance in $\{0,1\}^{\Theta(\frac{m \log n}{D})}$ in order to use BIN to estimate the distance between users preference vectors. Specifically, we propose to use the concept of *friends detector*, defined as follows.

DEFINITION 4.1. *Given a set of users with preference vectors $A_1, \ldots, A_n$ and $0 \le D_1 \le D_2$, a (D_1, D_2)-friends detector is a predicate that takes any two users i, i', whose value is true if dist$(A_i, A_{i'}) \le D_1$ and false if dist$(A_i, A_{i'}) > D_2$.*

The result of a friends detector may be arbitrary for users whose distance is between D_1 and D_2.[2]

High-level algorithm. Conceptually, our algorithm works as follows.

(1) Invoke Procedure DISTANCE_EST(A, α, D), and obtain a good friends detector. (This step uses a reduction to BIN.)

(2) Using the friends detector and additional probes, obtain a good approximation of A.

In Step 2, users probe random real objects directly, so that each object has sufficiently large coverage by all users types, and then, each user collects the results of probes obtained in the second stage by users close to him (as identified by the friends detector computed in Step 1), and outputs their average as an estimate of his grade vector. We elaborate on the implementation of each step later in this section, but state the following corollary, which summarizes the algorithm performance for α, D-similar preference matrix. It follows from Theorem 4.9, applied with Theorem 2.1.

THEOREM 4.1. *There is a distributed algorithm that, with overwhelming probability, reconstructs for each user its preference vectors with $O(D)$ L_1-distance error. This algorithm has a probe complexity $O\left(\frac{1}{\alpha}\left\lceil\frac{m}{n}\right\rceil \cdot \left\lceil\frac{\log n}{D}\right\rceil \log^{3.5}(m+n)\right)$.*

4.1 Estimating L_1-distances between users

The core of the first stage is carried out by Algorithm DISTANCE_EST, which uses a reduction to BIN. The idea is as follows. First we choose $\Theta(\frac{m \log n}{D})$ objects at random as virtual objects. This has the consequence that the expected distance between users whose distance over all m objects is D, is reduced to distance $\Theta(\log n)$ when considering only the virtual objects. Now, for each such virtual object a *random threshold* is selected independently and uniformly from $[0,1]$: the results of probes will be rounded according to these random thresholds.

More specifically, the algorithm works as follows (see Algorithm 2 for pseudo-code). Let $K = \lceil 3cm \log n/D \rceil$ for some constant $c > 4$. The algorithm chooses K tuples $\langle t_k, j_k \rangle_{k=1}^{K}$ where t_k are i.i.d. uniform random variable from $[0,1]$, and j_k are objects chosen independently, uniformly at random from $\{1,\ldots,m\}$. (Note that K, the number of virtual objects, might be larger or smaller than the number of real objects, m.) We define a binary matrix $B_{n\times K}$, where each row i corresponds to user i (whose real preference vector is A_i), and each column k corresponds to object j_k (i.e. to column j_k in A). B is defined as follows: Entry $B_{i,k}$ is the grade of user i for object j_k, rounded using t_k as a threshold. Algorithm BIN is invoked on B using the reduction framework of Figure 1, yielding reconstructed matrix $\hat{B}$. The output $W(i,i')$, the distance estimate between users i, i', is simply the adjusted Hamming distance between rows i, i' in $\hat{B}$.

We now turn to analyze Algorithm DISTANCE_EST. We first bound the errors due to the random choices of j_k and t_k values, and then consider the errors introduced by the imperfection of BIN.

[2]We note that [12] presents an implementation of a friends detector for the much simpler case where $D_1 = 0$.

Algorithm 2 DISTANCE_EST(A, α, D)

```
(1) Let $K = \lceil 3cm \log n/D \rceil$.
(2) For each $k \in \{1,\ldots,K\}$ let $\langle t_k, j_k \rangle$ be such that $t_k$ is chosen independently uniformly at random from $[0,1]$ and $j_k$ independently uniformly at random from $\{1,\ldots,m\}$ (with repetitions).
(3) Let $B$ be a binary matrix of size $n \times K$ defined by
    $B_{i,k} = \begin{cases} 1 & \text{If } A_{i,j_k} > t_k \\ 0 & \text{Otherwise}. \end{cases}$
(4) Let $\hat{B}$ be the output of BIN$(B, \alpha/2, 4KD/m)$. (Note that $4KD/m = \Theta(\log n)$.)
(5) Let $W(i,i') = \frac{m}{K}\mathrm{dist}(\hat{B}_i, \hat{B}_{i'})$. Output $W$.
```

Below, we maintain the following convention. We consider users i_1, i_2, i_3 such that $\mathrm{dist}(A_{i_1}, A_{i_2}) \le D$ and $\mathrm{dist}(A_{i_1}, A_{i_3}) > 5\beta$ where $\beta \overset{\mathrm{def}}{=} 4D \cdot \max(\gamma_{\text{BIN}}, 1)$. We will prove that the algorithm estimates the distances between users preferences with error bounded by $O(D\gamma_{\text{BIN}})$.

LEMMA 4.2. *If $\mathrm{dist}(A_{i_1}, A_{i_2}) \le D$ then $\mathrm{dist}(B_{i_1}, B_{i_2}) \le \frac{\beta}{2} \cdot \frac{K}{m}$ with probability at least $1 - n^{-c/4}$.*

Proof: Let χ_k be a random variable taking the value 1 if t_k falls in between A_{i_1,j_k} and A_{i_2,j_k} and 0 otherwise. Clearly $\mathrm{dist}(B_{i_1}, B_{i_2}) = \sum_{k=1}^{K} \chi_k$. Since t_k is chosen uniformly from $[0,1]$ we have

$$\begin{aligned}\Pr(\chi_k = 1) &= \sum_{j=1}^{m} \Pr(j_k = j) \Pr(\chi_k = 1 \mid j_k = j) \\ &= \sum_{j=1}^{m} \frac{1}{m} |A_{i_1,j} - A_{i_2,j}| \\ &\le D/m,\end{aligned}$$

and hence $E[\mathrm{dist}(B_{i_1}, B_{i_2})] = E\left[\sum_{k=1}^{K} \chi_k\right] \le KD/m = 3c \log n$. Therefore, since χ_k are independent Bernoulli random variables, Chernoff Bound implies that

$$\begin{aligned}\Pr(\mathrm{dist}(B_{i_1}, B_{i_2}) > 2KD/m) &= \Pr\left(\sum_{k=1}^{K} \chi_k > 2KD/m\right) \\ &< \exp\left(-\frac{1}{4} c \log n\right) \\ &= n^{-c/4}.\end{aligned}$$

The lemma follows, because $\beta \ge 4D$. ∎

LEMMA 4.3. *For all users i_1, $\mathrm{dist}(\hat{B}_{i_1}, B_{i_1}) \le \beta \cdot \frac{K}{m}$ with probability at least $1 - n^{-c/4-1}$.*

Proof: The lemma holds due to fulfillment of the conditions required for the success of BIN in Step 4 of Algorithm DISTANCE_EST. Let I be the αn preference vectors in A closest to A_{i_1}. By assumption, their distance from A_{i_1} is at most D. For any $i \in I$ and $k \in \{1 \ldots K\}$, let $\chi_{i,k}$ be random variable taking the value 1 iff t_k is between A_{i_1,j_k} and A_{i,j_k}. By definition of I, $E[\chi_{i,k}] \le D/m$ for any $i \in I$. By linearity of expectation, $E\left[\sum_{i\in I}\text{dist}(B_{i_1},B_i)\right] = \sum_{i\in I}\sum_{k=1}^{K} E\left[\chi_{i,k}\right] \le \alpha n DK/m$.

For each k, consider the random variable $\frac{1}{\alpha n}\sum_{i\in I}\chi_{i,k}$. This is a set of independent random variables in $[0,1]$, so by the Chernoff Bound we have

$$\begin{aligned}
&\Pr\left(\sum_{i\in I}\text{dist}(B_{i_1},B_{i_2}) > 2\alpha n DK/m\right)\\
&= \Pr\left(\sum_{k=1}^{K}\left(\frac{1}{\alpha n}\sum_{i\in I}\chi_{i,k}\right) > 2DK/m\right)\\
&< \exp\left(-c/4\log n\right) = n^{-c/4}.
\end{aligned}$$

It follows from the Markov Inequality that there is a subset $I_B \subseteq I$ such that $|I_B| \ge \frac{1}{2}|I|$ and $\text{dist}(B_{i_1},B_i) \le 4KD/m$ for any $i \in I_B$. Finally, we apply the Union Bound and deduce that the probability such I_B exists for all n users is at least $1-n^{-c/4-1}$. The result now follows from the fact that $\hat{B}$ is obtained by applying BIN with parameters $\alpha/2$ and $4KD/m$. ∎

Combining the above two bounds yields the first property of the algorithm

LEMMA 4.4. *For any i_i, i_2 satisfying dist$(A_{i_1},A_{i_2}) \le D$ we have $W(i_1,i_2) \le \frac{5}{2}\beta$, with probability $1-n^{-\Omega(1)}$.*

Proof: Summing over objects and using the triangle inequality with Lemmas 4.2 and 4.3, we obtain: with probability $1-n^{-\Omega(1)}$, if $\text{dist}(A_{i_1},A_{i_2}) \le D$ then

$$\begin{aligned}
W(i_1,i_2) &= \frac{m}{K}\text{dist}(\hat{B}_{i_1},\hat{B}_{i_2})\\
&= \frac{m}{K}\sum_{j=1}^{K}\left|\hat{B}_{i_1,j}-\hat{B}_{i_2,j}\right|\\
&\le \frac{m}{K}\left(\sum_{j=1}^{K}\left|\hat{B}_{i_1,j}-B_{i_1,j}\right|\right.\\
&\quad+\sum_{j=1}^{K}\left|B_{i_1,j}-B_{i_2,j}\right|\\
&\quad\left.+\sum_{j=1}^{K}\left|\hat{B}_{i_2,j}-B_{i_2,j}\right|\right)\\
&\le \frac{5}{2}\beta.
\end{aligned}$$
∎

Applying similar analysis, we can bound from below the distance estimates of users whose preference vectors are far away from each other.

LEMMA 4.5. *If dist$(A_{i_1},A_{i_3}) > 5\beta$ then dist$(B_{i_1},B_{i_3}) > \frac{9}{2}\beta\cdot\frac{K}{m}$ with probability at least $1-n^{-\frac{3c}{10}}$.*

Proof: For any k, define χ_k be a random variable taking the value 1 if t_k falls in between A_{i_1,j_k} and $A_{i_3 j_k}$ and 0 otherwise. Clearly $\text{dist}(B_{i_1},B_{i_3}) = \sum_{k=1}^{K}\chi_k$. Since t_k is chosen uniformly from $[0,1]$ we have

$$\begin{aligned}
\Pr(\chi_k=1) &= \sum_{j=1}^{m}\Pr\left(j_k=j\right)\Pr\left(\chi_k=1 \mid j_k=j\right)\\
&= \sum_{j=1}^{m}\frac{1}{m}\left|A_{i_1,j}-A_{i_3,j}\right|\\
&> \frac{5\beta}{m},
\end{aligned}$$

and hence $E\left[\text{dist}(B_{i_1},B_{i_3})\right] = E\left[\sum_{k=1}^{K}\chi_k\right] > \frac{5K}{m}\beta$. Therefore, since χ_k are independent Bernoulli random variables, we can apply the Chernoff Bound to obtain

$$\begin{aligned}
\Pr\left(\text{dist}(B_{i_1},B_{i_3}) \le \frac{9K}{2m}\beta\right) &= \Pr\left(\sum_{k=1}^{K}\chi_k \le \frac{9K}{2m}\beta\right)\\
&\le \exp\left(-\frac{K\beta}{40m}\right)\\
&\le n^{-\frac{3c}{10}}.
\end{aligned}$$

The last inequality follows from the fact $\beta \ge 4D$. ∎

LEMMA 4.6. *If dist$(A_{i_1},A_{i_3}) > 5\beta$ then $W(i_1,i_3) > \frac{5}{2}\beta$ with probability $1-n^{-\Omega(1)}$.*

Proof: Similarly to the proof of Lemma 4.4, summing over columns and using the triangle inequality with Lemmas 4.5 and 4.3, we get that if $\text{dist}(A_{i_1},A_{i_3}) > 5\beta$ then with probability $1-n^{-\Omega(1)}$

$$\begin{aligned}
W(i_1,i_3) &= \frac{m}{K}\text{dist}(\hat{B}_{i_1},\hat{B}_{i_3})\\
&= \frac{m}{K}\sum_{j=1}^{K}\left|\hat{B}_{i_1,j}-\hat{B}_{i_3,j}\right|\\
&\ge \frac{m}{K}\left(\sum_{j=1}^{K}\left|B_{i_1,j}-B_{i_1,j}\right|\right.\\
&\quad-\sum_{j=1}^{K}\left|\hat{B}_{i_1,j}-B_{i_3,j}\right|\\
&\quad\left.-\sum_{j=1}^{K}\left|\hat{B}_{i_3,j}-B_{i_3,j}\right|\right)\\
&> \frac{5}{2}\beta.
\end{aligned}$$
∎

Finally, we claim that DISTANCE_EST can be used to obtain a good implementation of a friends detector (cf. Def. 4.1). The following theorem is implied by Lemmas 4.4 and 4.6

THEOREM 4.7. *Let $A_{n\times m}$ be a (α,D)-similar matrix with real values from $[0,1]$. Then, given a binary reconstruction algorithm* BIN, *a $(D, 20\lceil\gamma_{\text{BIN}}\rceil D)$-friends detector can be implemented, with probe complexity $T_{\text{BIN}}(n,K,\alpha/2,c_1\cdot\log n)$ for some constant c_1, and with success probability $1-n^{-\Omega(1)}$.*

Proof: Compute W by algorithm DISTANCE_EST and output "true" iff $W(i,i') \le \frac{5}{2}\beta$. ∎

Algorithm 3 BUILD_PREFS(A, α, D, β) By user i

```
(1) Invoke Algorithm DISTANCE_EST$(A, \alpha, D)$ to compute a $(D, 5\beta)$-friends detector.
    Let $I = \left\{i' : W(i', i) \le \frac{5}{2}\beta\right\}$ be the set of users detected as "friends". (Recall that $\beta = 4\lceil\gamma\rceil D$.)
(2) Let $\mathcal{C} = \emptyset$. Repeat $r \log n$ times:
    (2a) Probe each object $j$, independently with probability $\frac{\log(m+n)}{\alpha n}$ and post probe results on the billboard.
         Let $S_i$ be the set of objects probed by user $i$.
         For each object $j$, let $I^j = \{i' \in I : j \in S_{i'}\}$, i.e., members of $I$ that probed $j$.
    (2b) For each object $j$ do:
         (2bi) If $j \in S_i$ then set $C_j$ to $A_{i,j}$ as probed in Step 2a.
         (2bii) If $j \notin S_i$ then set $C_j = \frac{1}{|I_i^j|}\sum_{i' \in I_i^j} A_{i',j}$. If $I_i^j = \emptyset$ set $C_j = 1/2$ (this will happen with negligible probability).
    (2c) $\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$.
(3) Return $C_l \in \mathcal{C}$ which maximizes
         $|\{C_{l'} \in \mathcal{C} : \mathrm{dist}(C_l, C_{l'}) \le 30\beta\}|$.
```

4.2 Reconstructing the preference vectors

We now describe the complete algorithm, assuming that we are given a good friends detector (Definition 4.1), and our task is to reconstruct an approximation of the preference vectors.

The idea is as follows (see pseudo-code in Algorithm 3). First, the users probe an appropriately large random sample of the objects: intuitively, this is to get sufficient coverage. Next, each user looks at the results of probes of users who are deemed "close," as determined by the friends detector; for each object, the user computes the averages of the friends' grades as a candidate output, thereby obtaining a candidate vector. We prove that in expectation, the candidate vector is close to the true preference vector. However, its variance is unknown, and therefore, to get a usable approximation, we repeat the procedure: we construct many candidate vectors independently, and output the one with many close neighbors (this is akin to picking the median of single-dimension samples). For that output, we prove that with high probability, the result is a good approximation.

We now analyze the algorithm. We start by considering a single candidate.

LEMMA 4.8. *Let A be (α, D)-similar, and suppose that Algorithm* BUILD_PREFS *uses a $(D, 4\lceil\gamma_{\text{BIN}}\rceil D)$-friends detector over A. Then for any candidate vector C computed by user i in Step 2b of Algorithm* BUILD_PREFS*, $E[dist(C, A_i)] \le 20\lceil\gamma_{\text{BIN}}\rceil D$.*

Proof: Since there are at least αn users with $\mathrm{dist}(A_i, A_{i'}) \le D$, Theorem 4.7 ensures that $|I| \ge \alpha n$, and hence $E\left[I^j\right] \ge \log(m+n)$. By the Chernoff Bound $|I^j| \ge 1$ with overwhelming probability. We apply the Union Bound and deduce that, with overwhelming probability, $|I^j| \ge 1$ for all j. Theorem 4.7 also ensures that I includes only users with preference vectors at distance at most $5\beta \le 20\lceil\gamma_{\text{BIN}}\rceil D$ from user i preference vector.

Fix i. For any $i' \in I$ and any $j \in S_{i'}$ let $\eta_{i'}^j$ be the random variable $\eta_{i'}^j = |A_{i,j} - A_{i',j}|$ with expectation (over the choices by user i') $E\left[\eta_{i'}^j\right] \le \frac{5\beta}{m}$. As the objects probed by each user are chosen independently of the values in the matrix A we may conclude that the random variables $\eta_{i'}^j = |A_{i,j} - A_{i',j}|$ are independent from the random variables I^j. By the triangle inequality,

$$\begin{aligned}
E[\mathrm{dist}(C, A_i)] &= E\left[\sum_{j=1}^{m} |C_j - A_{i,j}|\right] \\
&\le E\left[\sum_{j=1}^{m} \frac{1}{|I^j|}\sum_{i' \in I^j} \eta_{i'}^j\right] \\
&= \sum_{j=1}^{m} \frac{1}{|I^j|}\sum_{i' \in I^j} E\left[\eta_{i'}^j\right] \\
&\le 5\beta\,. \quad \blacksquare
\end{aligned}$$

Lemma 4.8 bounded the expected error of the reconstruction done by BUILD_PREFS. We now summarize the overall algorithm.

THEOREM 4.9. *Let $A_{n\times m}$ be (α, D)-similar matrix with entries in $[0,1]$, and let* BIN *be a binary reconstruction algorithm with approximation ratio γ_{BIN} and probe complexity T_{BIN}. Then with probability $1 - n^{-\Omega(1)}$, Algorithm* BUILD_PREFS *reconstructs A with approximation ratio $O(\gamma_{\text{BIN}})$ and probe complexity $T_{\text{BIN}}\left(n, K, \frac{\alpha}{2}, \frac{4KD}{m}\right) + O\left(\frac{m}{n} \cdot \frac{\log^2(m+n)}{\alpha}\right)$, where $K = \lceil 3cm \log n / D\rceil = O(\frac{m}{D}\log n)$.*

Proof: For each candidate $C \in \mathcal{C}$ let y be the random variable equal to the distance of the candidate from the user's preference vector, i.e. $y = |C - A_i|$. As the distances are non-negative random variables, Markov's Inequality implies that the probability of each distance to be larger than 3 times its expectation is less than $\frac{1}{3}$. By the Chernoff Bound, the probability that at least half of the distances are higher than 3 times their expectation is at most $n^{-r/36}$. Hence, with probability $1 - n^{-r/36}$, there is a subset $\mathcal{C}^*$ with $|\mathcal{C}^*| > \frac{1}{2}|\mathcal{C}|$ such that for each candidate $C \in \mathcal{C}^*$ we have that $\mathrm{dist}(C, A_i) \le 3E[y]$. It follows there is at least one candidate with at least $\frac{1}{2}|\mathcal{C}|$ neighbors at distance at most $6E[y]$ from it. Every candidate with at least $\frac{1}{2}|\mathcal{C}|$ neighbors has at least one neighbor at distance at most $3E[y]$ from A_i. Hence, the output vector is at distance at most $3E[y] + 30\beta$ from A_i. The Theorem follows, because $E[y] \le 5\beta$ by Lemma 4.8.

The complexity bound follows from Theorem 4.7 and the fact that only $O\left(\frac{m}{n} \cdot \frac{\log n \cdot \log(m+n)}{\alpha}\right)$ probes are added by Step 2a of Algorithm BUILD_PREFS. $\blacksquare$

5. REFERENCES

[1] N. Alon, B. Awerbuch, Y. Azar, and B. Patt-Shamir. Tell me who I am: an interactive recommendation system. In *Proc. 18th Ann. ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 1–10, 2006.

[2] B. Awerbuch, Y. Azar, Z. Lotker, B. Patt-Shamir, and M. Tuttle. Collaborate with strangers to find own preferences. In *Proc. 17th ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 263–269, 2005.

[3] B. Awerbuch, A. Nisgav, and B. Patt-Shamir. Asynchronous active recommendation systems. In *Principles of distributed systems : 11th international conference (OPODIS 2007)*, volume 4878 of *LNCS*, pages 48–61, 2007.

[4] B. Awerbuch, B. Patt-Shamir, D. Peleg, and M. Tuttle. Improved recommendation systems. In *Proc. 16th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 1174–1183, 2005.

[5] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proc. 33rd ACM Symp. on Theory of Computing (STOC)*, pages 619–626, 2001.

[6] R. M. Bell and Y. Koren. Lessons from the netflix prize challenge. *SIGKDD Explor. Newsl.*, 9(2):75–79, 2007.

[7] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *Proc. 34th ACM Symp. on Theory of Computing (STOC)*, pages 82–90, 2002.

[8] S. Gilbert, R. Guerraoui, F. M. Rad, and M. Zadimoghaddam. Collaborative scoring with dishonest participants. In *Proc. 22nd Ann. ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 41–49, 2010.

[9] K. Goldberg, T. Roeder, D. Gupta, , and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval Journal*, 4(2):133–151, July 2001.

[10] J. Kleinberg and M. Sandler. Convergent algorithms for collaborative filtering. In *Proc. 4th ACM Conf. on Electronic Commerce (EC)*, pages 1–10, 2003.

[11] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Recommendation systems: A probabilistic analysis. In *Proc. 39th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 664–673, 1998.

[12] A. Nisgav and B. Patt-Shamir. Finding similar users in social networks: extended abstract. In *Proc. 21st Ann. ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, pages 169–177, 2009.

[13] C. H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Proc. 17th ACM Symp. on Principles of Database Systems (PODS)*, pages 159–168. ACM Press, 1998.

[14] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proc. 1994 ACM Conf. on Computer Supported Cooperative Work*, pages 175–186, Oct. 1994.

[15] P. Resnick and H. R. Varian. Recommender systems. *Commun. ACM*, 40(3):56–58, 1997.

[16] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proc. 2nd ACM Conf. on Electronic Commerce (EC)*, pages 158–167. ACM Press, 2000.