Post Processing Recommender Systems for Diversity

Arda Antikacioglu
Department of Mathematical Sciences
Carnegie Mellon University
aantikac@andrew.cmu.edu

R Ravi
Tepper School of Business
Carnegie Mellon University
ravi@cmu.edu

ABSTRACT

Collaborative filtering is a broad and powerful framework for building recommendation systems that has seen widespread adoption. Over the past decade, the propensity of such systems for favoring popular products and thus creating echo chambers have been observed. This has given rise to an active area of research that seeks to diversify recommendations generated by such algorithms.[2, 11, 37].

We address the problem of increasing diversity in recommendation systems that are based on collaborative filtering that use past ratings to predict a rating quality for potential recommendations. Following our earlier work, [7], we formulate recommendation system design as a subgraph selection problem from a candidate super-graph of potential recommendations where both diversity and rating quality are explicitly optimized: (1) On the modeling side, we define a new flexible notion of diversity that allows a system designer to prescribe the number of recommendations each item should receive, and smoothly penalizes deviations from this distribution. (2) On the algorithmic side, we show that minimum-cost network flow methods yield fast algorithms in theory and practice for designing recommendation subgraphs that optimize this notion of diversity. (3) On the empirical side, we show the effectiveness of our new model and method to increase diversity while maintaining high rating quality in standard rating data sets from Netflix and MovieLens.

KEYWORDS

diversity, recommender systems, network flow, discrepancy, subgraph selection

1 MOTIVATION

Collaborative filtering has long been a favored approach in recommender systems since its recommendations are derived mainly from the record of interactions between users and items. However, a key concern of CF systems is the filter bubble, the idea that recommendation systems that focus solely on accuracy lead to echo chambers that amplify "rich-get-richer" effects among the recommended items [10, 14, 28, 40]. This problem stems from the way these systems are designed since they can only make confident recommendations on items that have had a lot of engagement, and hence increase their importance. This is the main motivation to diversify the recommendations of such CF systems.

Recent ethical concerns about algorithms have focused on similar issues about algorithmic results inherently being biased [24, 26]. One approach to counter the status quo, also advocated by Karger [24] is to explicitly design algorithms that do not discriminate by designing an appropriate objective function that will increase diversity in CF recommendations.

The importance of diversifying recommendations for the sake of the user arises from their intrinsic appreciation for novelty and serendipity, a view that is supported by psychological studies [34]. Conversely, research in recommendation systems [25] has shown that focusing solely on ratings hurts user satisfaction. This has led to a subfield of recommendation systems that focuses on improving diversity for the benefit of the user [37, 39]. A third motivation is the business need for diversifying recommendations: long-tail catalogs that are frequent in the internet [8, 9] as well as media distributors with all-you-can-play business models [18] require that the recommendations influence users to consume diverse content by driving more traffic to different portions of the site.

Roadmap: In this paper , we address the non-diverse nature of CF recommendations, the needs of a long-tail business to shape traffic on its own site, and diversifying recommendations for the benefit of the users. We define a notion of diversity conducive to all these needs based on the degree-properties of the graph defined by the recommendations (Section 2). After reviewing related work (Section 3), we show that the design problem for this notion can be solved efficiently both in theory and practice using network flow techniques (Section 4). We validate our method by showing how to adapt standard collaborative filtering algorithms with an efficient post-processing step to optimize for our measure of diversity by sacrificing very little on the recommendation quality on standard data sets (Section 5).

2 A NEW GRAPH OPTIMIZATION PROBLEM

We model all the user-item recommendations provided by a CF system as a bipartite graph, and the choice of recommendations actually given to the users as a subgraph selection problem in this graph. The constraints on the number of items that can be recommended to a user put

KDD'17, August 13-17, 2017, Halifax, NS, Canada
 978-1-4503-4887-4/17/08...$15.00
DOI: 10.1145/3097983.3098173

bounds on the out-degree of the user nodes. Following our earlier work [7], we model the problem of achieving diversity among the items as specifying target in-degree values for each item and then finding a subgraph that satisfies these constraints as closely as possible. We develop the resulting graph optimization problem next.

We start our discussion by reviewing the well known b-matching problem on bipartite graphs [30]. In the b-matching problem, an underlying bipartite graph $G = (L, R)$ with edge set E is given, along with a nonnegative weight g on the edges, and two vectors of non-negative integers $(c_1, \ldots, c_l)$ and $(a_1, \ldots, a_r)$ (degree bounds) such that $\sum_{i=1}^{l} c_i = \sum_{i=1}^{r} a_i$. The goal is to find a maximum g-weight (or minimum g-cost) subgraph H where the degree of vertex $u_i \in L$ in H is c_i for every $1 \leq i \leq l$ and the degree of vertex $v_j \in R$ in H is a_j for every $1 \leq j \leq r$. This problem generalizes the well-known maximum weight perfect matching problem, which can be obtained as a special case if we set all target degrees to 1. Like the perfect matching problem in bipartite graphs, the b-matching problem can be solved by a reduction to a network flow model by adding a source with arcs to L, a sink with arcs from R and using the degree constraints as capacities on these respective arcs.

Assume that the degree bounds are given. The vector $(c_1, \ldots, c_l)$ will be taken as a vector of hard constraints that must be met exactly, based on display constraints for the users. In other words, we consider a subgraph H to be a feasible solution if and only if $\deg_H^+(u_i) = c_i$ for all $u_j \in L$. The vector $(a_1, \ldots, a_r)$ will be specified by the recommendation system designer to reflect the motivations described above (increase the coverage of items in CF results, increase the novelty to users on average, or shape traffic to some items). However, this target degree bounds may be unattainable (i.e. there is no feasible b-matching for these degree bounds). To handle this potential infeasibility, we incorporate them in the objective. We call the vector $(a_1, \ldots, a_r)$, the target degree distribution. We now define the objective for a given feasible solution H,

$$D(H) = \sum_{v_j \in R} |deg_H^-(v_j) - a_j|$$

which is simply the sum of the violations (in both directions) of the degree constraints for R. We call this objective the discrepancy between H and the degree distribution $(a_1, \ldots, a_r)$, and we name the problem of meeting the hard constraints $(c_1, \ldots, c_l)$ while minimizing this objective the MIN-DISCREPANCY problem.

The MIN-DISCREPANCY problem defined above generalizes the b-matching problem, and has objective value zero iff there is a feasible b-matching for the given degree bounds. Like the weighted b-matching problem mentioned above, we can adapt our problem to the weighted setting where each edge has a real-valued weight. In this setting, the objective to maximize the total weight of the chosen edges, among graphs that have the minimal discrepancy possible from the given targets. We call this variant of the problem the MAX-WEIGHT-MIN-DISCREPANCY problem. **Post-processing a CF Recommender** We now show how we can apply the graph optimization problem defined above to post-processing the results of a CF recommendation system. As input to a CF system, we have a set of items I, a set of users U, and list of known ratings given by each user to different subsets of the items. The CF system outputs a relevance function $rel : U \times I \to [a, b]$ that takes pairs of users and items to a predicted recommendation quality in some interval on the real line. Without any extra information on the problem domain, CF systems employ user-based filtering, item-based filtering, matrix factorization, or other methods to arrive at these predicted rating qualities. For the rest of this paper, this rating function will be considered to be given as a black-box since its implementation details have no consequence on our model, even though we will experiment with various options in our empirical tests.

2.1 Summary of Contributions

(1) Following our earlier work [7], we model the problem of post-processing recommendations from a CF system to increase diversity as a maximum-weight degree-constrained subgraph selection problem to minimize the discrepancy from a target distribution.

(2) We demonstrate that the problem of finding maximum-weight min-discrepancy subgraph can be reduced to the problem of finding minimum cost flows. In particular, this shows that the discrepancy between a recommendation system and *any* desired indegree distribution can be minimized in polynomial time. The abundance of fast solvers [23] for this problem makes our model not just theoretically interesting, but also practically feasible. Moreover, we prove that aggregate diversity maximization can be implemented special case of the discrepancy minimization problem. This generalizes the work of Adomavicius and Kwon on maximizing aggregate diversity [2] while simultaneously maximizing recommendation quality and matching the same asymptotic runtimes.

(3) We conduct experiments on standard datasets such as MovieLens-1m, and Netflix Prize data. By feeding our discrepancy minimizer as a post-processing step on the undiversified recommendation networks created by standard collaborative filtering algorithms, we measure the trade-off our algorithm makes between discrepancy and recommendation quality under a variety of parameter settings. We compare against baselines and other diversification approaches, and find that our diversifier makes more relevant recommendations despite achieving higher diversity gains, as measured not only by our discrepancy measure, but also by standard sales diversity metrics such as the Gini index or aggregate diversity.

3 RELATED WORK

First we review related work on various collaborative filtering approaches, and then discuss various extant notions

of diversity already considered in the recommender system literature.

3.1 Collaborative Filtering

Collaborative filtering is the most versatile and widely accepted way of building recommender systems. The main idea behind collaborative filtering is to exploit the similarities between different users or between different items using user feedback. While there are many different methods for doing this, we constrain our evaluation to three representative approaches.

Matrix factorization approaches assume the existence of D latent features which describe both users and items, and seeks to find two rank D matrices whose product approximates the matrix of all known rankings. The advantage is that D is typically much smaller than the number of users or items. In our work, we experiment with a version of this approach due to Hu [22].

Another popular approach is neighborhood based recommenders, which can either be user-based or item-based. These approaches define a distance between pairs of users and pairs of items respectively, using measures like cosine similarity or Pearson correlation [13]. The user-based approach then predicts the unknown rating from user u to item i by taking a distance weighted linear combination of the ratings of similar users on item i. The item-based approach operates similarly, but instead takes a weighted combination of the ratings of user u on items similar to item i. We use the implementations of these methods in RankSys [36] in our experiments.

Finally, we consider a graph based recommender strategy due to Cooper et. al. [12]. This method considers a bipartite graph of known user and item interactions, ignoring all rating information. In this graph, a random walk of length 2 from a user u corresponds to the selection of a user similar to u, and a random walk of length 3 corresponds to sampling an item liked by a similar user. Recommendations for a user u are ranked according to how many random walks of length 3 starting at that user terminate at a given item. Since this method is both simple to state and implement on small to medium sized datasets, we use our own implementation of this method in our empirical comparisons. While less commonly used than the first two types of recommenders we discussed, this approach is still representative of a large class of recommendation strategies such as UserRank [16] or ItemRank [19].

3.2 Sales Diversity

The Need for Sales Diversity: As mentioned above, the need for system-level diversification in recommenders is a business related one. Since the internet enables businesses with low inventory costs, focusing on making more recommendations in the long tail can be an effective retail strategy. This view is most clearly expressed by Anderson, who advocates selling "selling less of more" [5]. Interestingly, recommender systems rarely capitalize on this opportunity, and often compound the problems observed with popularity bias. Indeed, Zhou et. al. find that YouTube's recommendation module leads to an increase in popularity for the most popular items [40]. Similarly, Celma et. al. report similar findings for music recommendations on Last.fm [10]. Hosanagar and Fleder show that this popularity bias can lead to subpar pairings between users and items, potentially hurting customer satisfaction [14], and McNee reports that a focus on accuracy alone has hurt the user experience of recommender systems [25]. Since recommendations have an outsized impact on customer behavior [2, 31], businesses have a need to control the distribution of recommendations that they surface in their recommenders.

Metrics for Sales Diversity: There are several well-established metrics for measuring sales diversity, and we focus our attention on three. The most popular among these is the aggregate diversity, which is the total number of objects that have been recommended to at least one user. Under this name, this measure has been used notably by Adomavicius and Kwon [2, 3] and by Castells and Vargas [35]. It has also been used as a measure of system-wide diversity under the name of coverage [1, 17]. While easy to understand and measure, the aggregate diversity leaves a lot be desired as a measure of distributional equality. In particular, aggregate diversity treats an item which was recommended once as well-covered as an item which was recommended thousands of times. For example, imagine a system that recommends each item in a set of n items twice. This network will have the same aggregate diversity as a network which recommends one of the items n times, and every other item only once, even though this system is much more biased than the first.

An example of a more nuanced metric is provided by the Gini index. This measure is most popularly used in economics, as a quantization of wealth or income inequality. The Gini Index can be adapted for the recommendation setting by considering the number of recommendations an item gets as its "wealth" in the system. The Gini index defines the most equitable distribution to be the one where every item is recommended an equal number of times. Given the actually realized distribution of recommendations, it aggregates the difference between the number of recommendations the bottom nth percentile gets in the system and the number of recommendations they would have obtained under the uniform distribution where n ranges from 0 to 100%. The measure we propose is a particularly good proxy for the Gini index, since both measure a notion of distance from the uniform distribution. Since recommender systems produce distributions even more unequal than the typical wealth distributions within a country, this metric has found widespread acceptance in the recommendation community [17, 21, 29, 32].

Finally, we consider the entropy of the distribution of recommendations. Entropy has its roots in physics and information theory, where it is used to measure the amount of information contained in a stochastic process. For every item, we can define a probability of being surfaced by the recommender by counting what fraction of recommendations (made to any user) point to this item. As with the Gini index, optimal entropy is achieved if and only if the recommendation

distribution is uniform. While less common than either aggregate diversity or the Gini index, the entropy of the recommender system has also been used by many researchers [32, 33].

We measure the diversification performance of our methods and the baselines we test in our experimental section by all three of these metrics - aggregate diversity, Gini index and entropy.

Approaches for Increasing Sales Diversity: Attempts at increasing sales diversity fall into two approaches: optimization and reranking. The optimization approach has been taken up most notably by Adomavicius and Kwon [2], who consider heuristic and exact algorithms for improving aggregate diversity. Their flow based solution is approximate, while their exact solution to this problem relies on integer programming and has exponential complexity. Our work in this chapter subsumes these approaches by giving an exact polynomial algorithm for aggregate diversity maximization. To the best of our knowledge, neither the Gini index nor the entropy of the degree distribution can be optimized in an exact sense.

The reranking based approaches are by far the more popular choice in increasing sales diversity. Here, we consider three different approaches by Castells and Vargas, spread across two different papers. The first two approaches model popularity complement (PC) and free discovery (FD) as a function of the probability of an item being known to a user under a probabilistic discovery model [37]. The third approach, using the Bayes Rule (AB), adjusts the prediction function using a Bayesian approach [38].

4 ALGORITHMS

In this section, we prove that discrepancy from a target distribution can be minimized efficiently by reducing this problem to one invocation of a minimum cost flow problem. This result holds regardless of the target in-degree distribution and the required out-degree distribution.

4.1 Construction of the Flow Network

Let $G = (L, R, E)$ be the input bipartite graph which contains candidate recommendations. We construct a flow network out of G such that the min-cost feasible flow will have cost equal to the min-discrepancy. Our network will have $|V| + 2$ nodes: two special sink nodes t_1 and t_2, as well as a copy of each node in G (See Figure 1). We set the supply of each node u_i to c_i (its specified out-degree), and the demand of the sink t_2 to $\sum_{j=1}^{r} a_j = \sum_{i=1}^{l} c_i$. Next, for each arc $(u_i, v_j) \in G$, we create an arc (u_i, v_j) in the flow network, with unit capacity and zero cost. For each node v_j, we create an arc to each sink. We add the arc (v_j, t_1) of capacity a_j (its target out-degree) and zero cost, and the arc (v_j, t_2) of infinite capacity and cost 2. We finally add to our network an arc (t_1, t_2) between the two sinks, with infinite capacity and zero cost. Our assumptions ensure that total supply, $\sum_{i=1}^{l} c_i$ meets total demand $\sum_{j=1}^{r} a_j$, and that a feasible flow exists since each node u_i in L can send as much as $deg_G^+(u_i) \geq c_i$ flow to the sink t_2 via any c_i different neighbors. Note that there are $|E| + 2|R| + 1$ arcs in total in our flow network. The complete flow network constructed this way is shown in Figure 1.

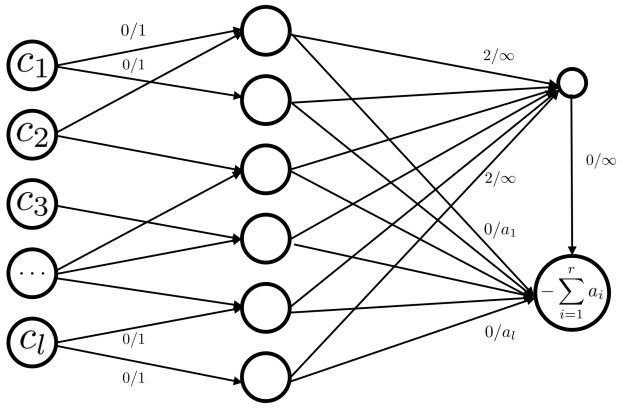

Figure 1: The network flow model for the MIN-DISCREPANCY problem with nodes labelled with their supply and arcs labeled with their cost/capacity. Unlabelled nodes have zero supply.

Our main theorem shows that the minimum cost of a flow in this network is the same as the minimum discrepancy a subgraph of G has from our target in-degree distribution.

THEOREM 4.1. *Suppose $G = (V, E)$ satisfies $deg_G^+(u_i) \geq c_i$ for all $u_i \in L$ and the degree distributions satisfy $\sum_{i=1}^{l} c_i = \sum_{i=1}^{r} a_i$. Then the minimum cost flow in the network constructed above has the same cost as the value of the MIN-DISCREPANCY problem and can be computed in $O(|E||V|^2 \log(|V|))$ time.*

PROOF. Consider a minimum cost flow in this network. Since the network's capacities and supplies are all integral, we may assume that this minimum cost flow is integral as well [4]. This means each edge crossing from L to R is either fully used or unused because it is unit capacity.

We let H be a subgraph of G defined by taking the edges of the form (u_i, v_j) where (u_i, v_j) is used in the flow. Since each such edge is either used or unused, and the supply of node u_i is c_i, we will satisfy the constraints of the form $\deg_H^+(u_i) = c_i$. To see that the cost of this flow is the same as the cost of our objective, note that we can partition the vertices in R into two halves: P for the vertices satisfying their degree requirement $\deg_H^-(v_j) \geq a_j$, and N for the vertices not satisfying their degree requirement. We can now write our objective as follows.

$$\sum_{v_j \in R} |\deg_H^-(v_j) - a_j| = \sum_{v_j \in P} \left(\deg_H^-(v_j) - a_j\right) + \sum_{v_j \in N} \left(a_j - \deg_H^-(v_j)\right)$$

However, note that our flow is feasible. Therefore, the total number of edges recommended is $\sum_{i=1}^{l} c_i$. It now follows

that $\sum_{v_j \in R}(\deg^-_H(v_j) - a_j) = \sum_{v_j \in R} \deg^-_H(v_j) - \sum_{i=1}^{l} c_i = 0$ from our assumption that $\sum_{i=1}^{l} c_i = \sum_{j=1}^{r} a_j$. Adding this to the expression above gives the following.

$$\sum_{v_j \in R} |\deg^-_H(v_j) - a_j| = 2 \sum_{v_j \in P} \left(\deg^-_H(v_j) - a_j\right)$$

In our formulation, we only pay for the flow going through a node v_j if the flow is in excess of a_j. Since we pay 2 units of cost for each unit of this type of flow, and don't pay for anything else, our objective matches that of the flow problem.

By reversing our reduction, we can show that every subgraph H with the desired properties induces a flow with the same cost as well. Therefore, the minimum discrepancy problem can be solved by a single invocation of a minimum cost flow algorithm, on a network with $|L| + |R| + 2$ nodes and $2|R| + |E| + 1$ edges with capacity bounded by $|V|$. This can be solved in $O(|E||V|^2 \log(|V|))$ time, using capacity scaling or other competitive methods [27]. □

Aggregate Diversity. Recall that aggregate diversity is the total number of items recommended by a recommender system. Aggregate diversity does not correspond to discrepancy from any target distribution, however it can be maximized by our model as well.

THEOREM 4.2. *Suppose $\sum_{i=1}^{l} c_i \geq r = |R|$. Aggregate diversity is maximized by the minimum cost flow solution in the network constructed for the MIN-DISCREPANCY$(G, \{c_i\}_{i=1}^{l}, \{1\}_{j=1}^{r})$ problem.*

PROOF. The sufficiency of our condition is obvious as it is needed to make sure that the supply of the nodes in L can be absorbed by the sink node. Now suppose that a recommender system achieves aggregate diversity r. A total of $\sum_{i=1}^{l} c_i$ units of flow make it to the sink, and each has to travel through an arc of cost 0 or 1. Since there are r different items in R, and each can send 1 unit of flow without cost, this solution has cost $\left(\sum_{i=1}^{l} c_i\right) - r$. Conversely, suppose some solution obtains cost $\left(\sum_{i=1}^{l} c_i\right) - r$. The only way the cost can be reduced below $\left(\sum_{i=1}^{l} c_i\right)$ is achievable is through the use of 0 cost arcs. Since each such arc has capacity 1, at least r such arcs must be used in the solution. This implies that a solution of aggregate diversity r exists. □

4.2 Incorporating Recommendation Relevance

4.2.1 Cumulative Gain. Note that we have had to assign zero costs to all the edges crossing between the two sides of our bipartition in order for our reduction to work. Recommendation strengths can be taken into account by our flow based methods, and we can find the graph that has the highest total recommendation quality given a discrepancy value using an extra pass with a flow solver. This can be accomplished as follows: first, we solve the regular discrepancy problem, and finding the lowest discrepancy value OPT attainable by the underlying G. Knowing this value, we can now fix the flow between t_1 and t_2 in the original flow network to $lc - OPT$, where lc is the total out-degree of the subgraph from L. This constrains the flow solver to choose subgraphs where exactly OPT of the recommendations go over the charged edges. We then keep all the other capacities the same and add new nonzero weights reflecting recommendation quality while removing all other costs. In a second pass, we find the highest weight flow in this network, which corresponds to the recommender graph with OPT discrepancy with the highest total recommendation quality. Therefore, we can solve the MAX-WEIGHT-MIN-DISCREPANCY problem with only two calls to a minimum cost flow solver. We call this approach the two-pass method[1]. Maximizing average recommendation quality in this fashion corresponds to the finding the recommendation subgraph with the highest cumulative gain.

4.2.2 Bicriteria Optimization. In the construction above, we needed to make an extra pass with a flow solver in order to find a solution with a high level of relevance. If we used these cost settings along with the cost settings we used in 4.1, we would no longer be optimizing only for ratings, or only for discrepancy. Instead, this results in a bicriteria objective of the form $\text{discrepancy}(H) - \mu \cdot \text{rel}(H)$, where μ can be any real number, and where relevance of a solution graph denotes the average relevance of the recommendations in H as predicted by the underlying CF recommender. We call this approach the weighted method, and demonstrate that while it is strictly worse than the two-pass method in theory, it yields acceptable results in practice while saving an extra pass of flow minimization. We discuss the performance differences in our experimental section.

4.3 Greedy Algorithm

In this section we describe an alternative approach to solving the discrepancy minimization problem that does not require the use minimum cost flow solvers, which can be efficient in practice, but do not guarantee linear runtimes. Our greedy algorithm constructs the solution subgraph iteratively, making a discrepancy reducing recommendation whenever possible. If such an edge is not available, then we choose from all available recommendations. Our choice of recommendation is conditioned on the quality of this recommendation, as measured by our black-box relevance function rel.

Since the greedy algorithm considers all discrepancy reducing recommendations for a user at the same time, a large number of candidate recommendations may lead to the greedy algorithm making subpar selections, since almost every recommendation we consider early on will likely be a discrepancy reducing edge. In order to moderate this effect, we include a parameter $q > 1$ which we use to reweigh the relevance scores. The larger the q is, the more our greedy algorithm prefers making relevant recommendations. On the other hand, if we

[1]This follows the goal-programming methodology for two-objective functions, popular in Operations Research.

pick a q which is too large, then we overprioritize high relevance values, and the greedy algorithm effectively turns into the standard recommendation approach. To balance these concerns, we run the greedy algorithm with different settings of q, and select the solution with the highest predicted rating quality which has discrepancy at most 10% higher than the best solution we generate.

5 EXPERIMENTS

In this section, we put our model to the test. Our findings are summarized below, and we discuss each point further in the following subsections.

(1) Our fast models perform well at optimizing for pre-existing notions of diversity such as aggregate diversity and the Gini index despite these measures not being explicitly referenced in our model. Conversely, we show that optimizing directly only for aggregate diversity (either by using heuristics or solving to optimality) does not yield results that are diverse by the other measures (See Table 1)

(2) Normalized discrepancy can often be reduced by more than 50%, at the cost of only a 15-30% change in average recommendation quality. Both the two-pass method, and the weighted method performed well in producing a smooth trade-off between recommendation quality and discrepancy reduction, with large gains in discrepancy being made for minimal recommendation quality loss (See Figure 3). The two-pass method is optimal, but the weighted-model provides a good approximation of the two-pass method's output with less computational overhead.

(3) Sales diversity maximization problems become easier as the display constraints are relaxed since there are more opportunities for the system to make unconventional recommendations. We show that the advantage our optimization based approach has over competing approaches gets bigger as display constraints are tightened, which is desirable for applications on mobile platforms where screen real estate is scarce (see Figure 2).

(4) Using the uniform target distribution can lead the optimizer to pick subgraphs where degree constraints are violated by large margins at certain nodes. To remedy this, we advocate the use of target distributions that move towards resembling the underlying degree distribution rather than the uniform distribution.

Experimental Setup and Datasets. All of our experiments were conducted on a desktop computer with an Intel i5 processor clocked at 2.7GHz, and with 16GB of memory. We used three rating datasets to generate the graphs we fed to our flow solvers: MovieLens-1m, MovieLens-10m [20] and the Netflix Prize dataset.

We pre-process the datasets to ensure that every user and every item has an adequate amount of data on which to base predictions. This pre-processing leaves the MovieLens-1m data with 5800 users and 3600 items, the MovieLens-10m dataset with 67000 users and 9000 items, and the Netflix dataset with 8000 users and 5000 items. The use of these datasets is standard in the recommender systems literature. In this work, we consider the rating data to be triples of the form $(user, item, rating)$, and discard any extra information.

We will henceforth report results only using the MovieLens data due to space constraints since the results from the Netflix data are similar. We used version 0.4.4 of the RankSys project to generate recommendations using standard collaborative filtering approaches [36]. The resulting network flow problems were optimized using a modified version of the MCFSimplex solver due to Bertolini and Frangioni [15]. Our choice of MCFSimplex was motivated by its open-source status and efficiency, but any other minimum cost flow solver such as CPLEX or Gurobi which accepts flow problems in the standard DIMACS format can also be used by our algorithms. Our discrepancy minimization code is available at Github.

Quality Evaluation. To evaluate the quality of our method, we employ a modified version of k-fold cross-validation. In particular, for each user in our datasets who has an high enough number of observed ratings, we divide the rating set into 10 equal sized subsets, and place each subset in one of 10 test sets. When creating the test sets, we filter out the items which received a rating of 1 or 2 and keep the items which received a rating of 3 or higher in order to ensure the relevance of our selections. We then define the precision of our recommendation list to be the number of items we recommend among all of our top-c recommendations which are also included in the test set. This provides an underestimate of the relevance of our recommendations, as there might be items which are relevant, but for which we have no record of the user liking. A simpler version of our hold-out method is utilized in other works [12, 38] where only a single random split is made. Using a 10-fold split of the test data enables us to run a signed rank test, and test whether the improvements made by our algorithms are statistically significant.

Our methodology stands in contrast with the methodology used by Adomavicius and Kwon to evaluate the effectiveness of their aggregate diversity maximization framework [2]. They use a metric called prediction-in-top-c, which measures the average predicted relevance of the c recommended items for each user. We believe that using predicted ratings for relevance evaluation purposes is flawed since these predictions are approximate in the first place. Furthermore, using the relevance values used by the recommender make comparisons across different recommenders difficult, as each recommender has its own scale.

Supergraph Generation. All of our optimization problems require that a supergraph of candidate recommendations be given. For each dataset we used, we generated 240 supergraphs in total. This is the result of using 10 training sets, 4 different recommender approaches, and 6 different quality thresholds enforced by picking the top 50, 100, 200, 300, 400 and 500 recommendations for each user. We use k to denote the number of candidate recommendations in the supergraph. The matrix factorization model we utilize [22] comes with three parameter settings: a regularization parameter λ, a confidence parameter α and the number of latent factors [22].

While the authors report an α value of 40 is suitable for most applications, we set a lower value of $\alpha = 30$ in order

to obtain more diverse candidate recommendation lists. The regularization parameter λ was tuned with cross-validation as recommended by the authors, and the model was trained with 50 latent factors. For the neighborhood based methods, we consider neighborhoods of size 100 in both the item-based and user-based cases. For these recommenders we opted to use the inverted neighborhood policy approach described in [38] in order to obtain more diverse candidate recommendation lists. Instead of using the top 100 most similar items to an item i as the neighborhood, this approach uses the items which have item i in their top 100 neighborhoods. We also used Jaccard similarity in order to measure similarity between pairs of users and items in the neighborhood based methods.

The authors of the random walk recommender we implemented consider a parameter setting α which raises every element of the transition matrix to the power α and find that predictive accuracy is maximized for $\alpha = 1.5$ [12]. Since they conduct their experimental validation on the same datasets as ours, we also use this parameter in our tests. In our tables, we shorten the names of these recommenders as MF for the matrix factorization model, IB and UB for the (item- and user-) neighborhood based approaches, and RW for the random walk approach.

5.1 Comparison To Other Methods and Metrics

In this section we compare our discrepancy minimization framework to other similar approaches. In particular, we test 6 different approaches to diversifying the recommendation lists.

- **Top (TOP)**: The standard method considers the unmodified output of the underlying recommender, and makes the top-k recommendations for each user. This is the undiversified solution but provides the highest rating quality.
- **Two pass (GOL)**: The two-pass method first finds the lowest discrepancy value achievable with the given graph for the current target degrees, and then in a second pass, finds the highest rating solution which achieves this minimum.
- **Aggregate Diversity (AGG)**: The aggregate diversity maximizing method is also optimized using our own flow-based framework, by running our min-cost flow algorithms with the setting of $a_i = 1$ as described in Theorem 4.2.
- **PC Reranking (PC), FD Reranking (FD) and Bayes Rule Reranking (AB)**: These diversifiers are due Vargas and Castells [37, 38], and were mentioned in our related work section.
- **Greedy (GRD)**: This is an implementation of the greedy heuristic described in Section 4.3.

We evaluate these different approaches on the following metrics, all measured for the top-n recommendation task on both the MovieLens and Netflix data.

- **D@n:** Discrepancy from the uniform distribution, normalized to fit in the $[0, 1]$ range by dividing by the maximum discrepancy achievable, i.e. $2\sum_{i=1}^{l} c_i$.
- **A@n:** The fraction of items which received a recommendation.

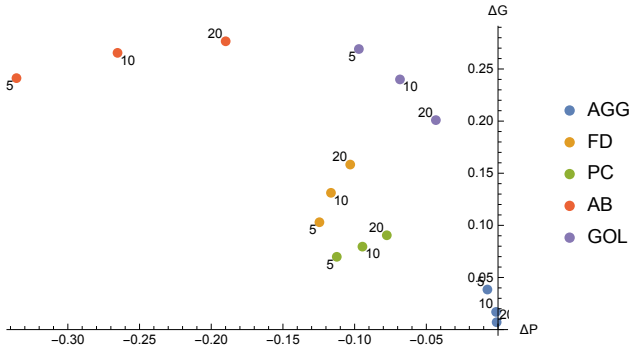

Figure 2: The change in the Gini index and the precision of the diversified solutions when the recommendation list length is varied. Values to the right and towards the top are better.

- **G@n:** The Gini index of the degree distribution of items.
- **E@n:** The entropy of the probability distribution formed by normalizing this degree distribution.
- **P@n:** Precision, measured as the fraction of items in the recommendation list which are part of the test set.

Table 1 summarizes our results for the MovieLens-1m dataset. The results from the other datasets can be found in the full version of this paper [6].

The first thing to notice in this table is that the undiversified recommendation lists perform very poorly with respect to all distributional measures. This is true with respect to even the simplest measure, aggregate diversity. The Random Walk Recommender in particular surfaces only 7% of the items in the MovieLens catalog. Other recommenders do not do particularly better, and only surface 15-20% of items to the users via top-10 recommendation lists.

5.2 Effect of Recommendation List Length

With more computer usage shifting from devices with larger displays like desktops and laptops to mobile devices like phones and tablets, display constraints that govern the number of recommendations we can make to user have gotten tighter. Therefore, it has become increasingly important for diversification approaches to be effective even when there is space for only a few recommendations to be made. In Figure 2, we fix the underlying subgraph to be the graph generated by our MF recommender with 200 candidate recommendations for each user, and measure the change in Gini index against the precision loss of the different diversifiers with display constraints set to $c = 5, 10$ and 20. We note that our diversifier performs better as the display constraints get tighter.

For the top-5 recommendation task, our method effectively outperformed all other diversifiers. Considering all of these factors, we conclude that an optimization based framework works better in applications where display constraints are

		k=50					k=250					k=500				
		P@10	A@10	G@10	E@10	D@10	P@10	A@10	G@10	E@10	D@10	P@10	A@10	G@10	E@10	D@10
MF	TOP	**0.433**	0.263	0.924	5.974	0.823	**0.433**	0.263	0.924	5.974	0.823	**0.433**	0.263	0.924	5.974	0.823
	AGG	***0.433***	**0.429**	0.919	6.024	0.816	*0.432*	**0.651**	0.906	6.094	0.804	*0.432*	**0.758**	0.896	6.146	0.796
	FD	0.340	0.385	0.826	6.819	0.712	0.316	0.478	0.792	7.000	0.658	0.330	0.528	0.802	6.952	0.659
	PC	0.362	0.331	0.858	6.619	0.754	0.335	0.347	0.843	6.715	0.736	0.321	0.356	0.836	6.756	0.728
	AB	0.257	**0.429**	**0.767**	**7.081**	0.661	0.167	0.649	0.648	**7.504**	0.503	0.204	0.735	0.660	7.472	0.507
	GOL	0.414	0.423	0.855	6.451	**0.658**	0.358	0.639	**0.646**	7.189	**0.414**	0.331	0.754	**0.483**	**7.581**	**0.282**
	GRD	0.269	0.418	0.815	6.863	0.670	0.125	0.620	0.603	7.583	0.449	0.083	0.734	0.455	7.852	0.324
IB	TOP	**0.410**	0.257	0.953	5.450	0.861	**0.410**	0.257	0.953	5.450	0.861	**0.410**	0.257	0.953	5.450	0.861
	AGG	*0.409*	**0.431**	0.947	5.496	0.855	*0.409*	**0.653**	0.933	5.573	0.843	*0.409*	**0.813**	0.918	5.640	0.833
	FD	0.366	0.352	0.906	6.202	0.795	0.321	0.465	0.842	6.730	0.708	0.296	0.633	0.781	7.035	0.622
	PC	0.368	0.317	0.921	6.028	0.819	0.338	0.332	0.902	6.261	0.796	0.323	0.348	0.889	6.383	0.779
	AB	0.282	0.429	**0.861**	**6.610**	0.742	0.196	0.641	**0.735**	**7.231**	0.591	0.214	0.782	0.695	**7.299**	0.528
	GOL	*0.404*	0.380	0.911	5.834	**0.735**	0.380	0.573	0.763	6.611	**0.524**	0.358	0.733	**0.605**	7.160	**0.375**
	GRD	0.258	0.383	0.898	6.215	0.753	0.135	0.556	0.747	7.163	0.568	0.090	0.695	0.610	7.577	0.437
UB	TOP	**0.412**	0.146	0.971	5.002	0.903	**0.412**	0.146	0.971	5.002	0.903	**0.412**	0.146	0.971	5.002	0.903
	AGG	***0.412***	**0.308**	0.967	5.062	0.895	***0.412***	**0.585**	0.952	5.176	0.878	***0.412***	**0.777**	0.936	5.262	0.866
	FD	0.380	0.282	0.902	6.242	0.797	0.335	0.498	0.840	6.674	0.686	0.333	0.619	0.833	6.630	0.665
	PC	0.391	0.240	0.922	6.029	0.833	0.360	0.275	0.903	6.252	0.809	0.350	0.279	0.898	6.300	0.802
	AB	0.291	0.307	**0.861**	**6.592**	0.761	0.195	0.566	0.747	**7.158**	0.593	0.210	0.696	0.703	7.240	0.528
	GOL	**0.412**	0.306	0.928	5.530	**0.760**	0.382	0.582	**0.732**	6.618	**0.486**	0.342	0.772	**0.507**	**7.358**	**0.297**
	GRD	0.265	0.303	0.900	6.210	0.766	0.136	0.566	0.703	7.302	0.519	0.091	0.746	0.519	7.757	0.353
RW	TOP	0.303	0.072	0.992	3.710	0.970	0.303	0.072	0.992	3.710	0.970	0.303	0.072	0.992	3.710	0.970
	AGG	*0.304*	**0.262**	0.989	3.786	0.960	*0.302*	**0.515**	0.976	3.909	0.943	0.297	**0.670**	0.967	3.965	0.936
	FD	0.343	0.232	0.964	5.203	0.898	0.289	0.466	0.911	5.968	0.780	0.284	0.523	0.908	5.792	0.761
	PC	**0.351**	0.205	0.967	5.147	0.905	**0.339**	0.364	0.933	5.819	0.829	**0.348**	0.404	0.929	5.842	0.819
	AB	0.287	0.261	**0.957**	**5.409**	0.889	0.209	0.488	**0.881**	**6.429**	0.766	0.183	0.550	0.846	**6.625**	0.705
	GOL	*0.319*	0.228	0.980	4.122	**0.887**	*0.302*	0.489	0.896	5.177	**0.689**	0.248	0.633	**0.780**	5.980	**0.536**
	GRD	0.213	0.238	0.966	5.129	0.889	0.130	0.478	0.856	6.627	0.708	0.094	0.613	0.743	7.206	0.574

Table 1: Comparison of different diversifiers systems on various diversification metrics for the 10 item recommendation task. The underlying dataset is the MovieLens-1m dataset and the candidate recommendations were generated using Matrix Factorization (MF), Item-Based (IB), User-Based (UB), and Random Walk Recommenders (RW). The bolded entries show the best values in each metric (ignoring the greedy algorithm), and italicized values show a statistically insignificant change from the baseline with $p < 0.01$.

particularly tight. The reranking approaches make recommendations for each user independently, whereas our optimization framework makes all of these recommendations at once, while optimizing explicitly for a diversity metric. Therefore, our two-pass method is better able to coordinate a small number of recommendations to make large gains in diversity while also keeping precision high.

5.3 Trading Off Discrepancy and Precision

In this section we explore the discrepancy/precision tradeoffs made by our different models. Throughout the section, we consider the discrepancy from the uniform target distribution. This target indegree distribution sets $a_j = c \cdot l/r$ for each $v_j \in R$ for a fixed c which represents the display constraints. The discrepancy from this target can be thought of as an extreme measure of diversity, since we are measuring distance from the most equitable distribution.

For the graph in Figure 3, we increase the number of candidate recommendations for each user from 100 to 500 in increments of 100, and show how this affects the recommendation quality of our solution as well as the lowest discrepancy achievable. We plot the normalized discrepancy to the uniform target on the x-axis against precision in the y-axis. Discrepancy improves towards the left, and recommendation quality improves towards the top. The trajectory of the curves we plot in in this figure Figure 3 is representative of the trajectories found using different datasets and different recommenders, and our full results can be found in [6].

We consider 4 different approaches to reducing discrepancy. The first is our two-pass method, which optimizes for discrepancy first, then for average predicted relevance across the system. We also consider the weighted method with the settings $\mu = 1$, and $\mu = 1/2$ and $\mu = 0.01$. Recall that the weighted method optimizes the objective discrepancy$(H) - \mu \cdot \text{rel}(H)$, where rel$(H)$ denotes the average recommendation quality in the solution graph H we produce. Therefore, the weighted method does not optimize for discrepancy. Instead, it find a solution where the cost of reducing discrepancy by μ units is the same as reducing aggregate predicted relevance by 1 unit. When run on the same input graph, the predicted relevance of the two-pass method is always lower than the predicted relevance of the $\mu = 1$ model, and the predicted relevance of the weighted method with μ is always lower than the predicted relevance of the weighted method with $\mu' > \mu$. The discrepancies produced by these algorithms on the same graph are also ordered in the same way.

From this figure, we can immediately notice certain features. First, all three algorithms produce highly clustered data with initial normalized discrepancy from the uniform

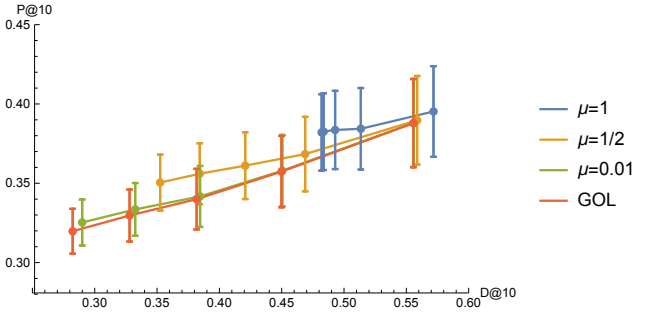

Figure 3: Precision discrepancy trade-off in the MovieLens-1m dataset using the MF recommender. In each series, the number of edges in the input graph increases towards the left.

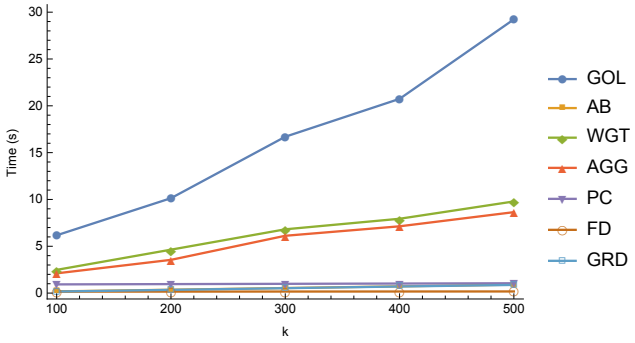

Figure 4: Time to optimize the top-10 recommendation task in MovieLens-1m based graphs in seconds ($|L|$=5800,$|R|$=3600)

distribution always being over 0.8. Second, the fall-off in discrepancy happens first quickly, then slowly as more edges are included. Therefore, significant gains are possible even while including a small number of candidate recommendations.

One notable difference between the weighted methods and the two-pass method is that in the weighted method, the discrepancy improvements start slowing down as more and more candidate edges are included in the supergraph. As mentioned above, the bicriteria objectives improve when discrepancy gains can be made which offset the fall in predicted relevance. As we enlarge the candidate set of recommendations, we enable discrepancy increases with edges that are less and less able to make up for the corresponding relevance losses. Therefore, the solution graph stops changing though lower discrepancies are possible. Where this limiting point lies depends on the structure of the graph and the distribution of relevance values assigned by the underlying recommender and is not easy to predict. However, for suitably low values of μ, the weighted method can adequately approximate the output of the two-pass method, achieving essentially the same trade-off curves.

5.4 Resource Use

Since our methods are based on minimum cost flow, the problems that result from our reduction can be solved efficiently. The graph in Figure 4 shows the cost of optimizing for uniform discrepancy with the two-pass method and the weighted method in the MovieLens-1m graphs. We increase the number of candidate recommendations from 100 to 500, and report the average runtime across the different recommenders. The labels for this plot are identical to the labels we have been using so far, with the exception of WGT, which denotes a run of the weighted method. We did not find significant runtime differences between different settings of μ for this model, and present the results for a representative run with the setting $\mu = 1$. While our methods do not run as efficiently as reranking methods, they provide much better diversification, and even instances with tens of millions of candidate recommendations can be run on a desktop.

6 CONCLUSIONS

We have proposed a new way of measuring how equitably a recommender system distributes its recommendations called discrepancy, and showed that it can be optimized for in polynomial time using network flow techniques. We validated the effectiveness and the efficiency of our method by conducting extensive tests on MovieLens and Netflix datasets, and showed it to improve diversity across a variety of measures. Our work demonstrates that distributional diversity measures like discrepancy can be efficiently optimized to allow information designers to have more control over their recommender systems.

REFERENCES

[1] Panagiotis Adamopoulos and Alexander Tuzhilin. 2011. On Unexpectedness in Recommender Systems: Or How to Expect the Unexpected. In *DiveRS@ RecSys*. 11–18.
[2] Gediminas Adomavicius and YoungOk Kwon. 2011. Maximizing aggregate recommendation diversity: A graph-theoretic approach. In *Proc. of the 1st Int. Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011)*. Citeseer, 3–10.
[3] Gediminas Adomavicius and YoungOk Kwon. 2012. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Trans. on Knowl. and Data Eng.* 24, 5 (2012), 896–911.
[4] Ravindra Ahuja, Thomas Magnanti, and James Orlin. 1993. *Network flows: theory, algorithms, and applications*. Prentice Hall.
[5] Chris Anderson. 2006. *The Long Tail: Why the Future of Business Is Selling Less of More*. Hyperion.
[6] Arda Antikacioglu and R. Ravi. 2017. Network Flow Based Post Processing for Sales Diversity. (2017). arXiv:1702.05446
[7] Arda Antikacioglu, R Ravi, and Srinath Sridhar. 2015. Recommendation Subgraphs for Web Discovery. In *Proc. of the 24th Int. Conf. on World Wide Web*. Int. World Wide Web Conf.s Steering Committee, 77–87.
[8] Erik Brynjolfsson, Yu Hu, and Duncan Simester. 2011. Goodbye pareto principle, hello long tail: The effect of search costs on the concentration of product sales. *Management Science* 57, 8 (2011), 1373–1386.
[9] Erik Brynjolfsson, Yu Hu, and Michael D Smith. 2003. Consumer surplus in the digital economy: Estimating the value of increased product variety at online booksellers. *Management Science* 49, 11 (2003), 1580–1596.
[10] Òscar Celma and Pedro Cano. 2008. From hits to niches?: or how popular artists can bias music recommendation and discovery. In *Proc. of the 2nd KDD Workshop on Large-Scale Recommender Systems*. ACM, 5.
[11] Fabian Christoffel, Bibek Paudel, Chris Newell, and Abraham Bernstein. 2015. Blockbusters and Wallflowers: Accurate, Diverse,

and Scalable Recommendations with Random Walks. In *Proc. of the 9th ACM Conf. on Rec. Systems*. ACM, 163–170.

[12] Colin Cooper, Sang Hyuk Lee, Tomasz Radzik, and Yiannis Siantos. 2014. Random walks in recommender systems: Exact computation and simulations. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 811–816.

[13] Christian Desrosiers and George Karypis. 2011. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*. Springer, 107–144.

[14] Daniel Fleder and Kartik Hosanagar. 2009. Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management Science* 55, 5 (2009), 697–712.

[15] Antonio Frangioni and Luis Perez Sanchez. 2010. Searching the best (formulation, solver, configuration) for structured problems. In *Complex Systems Design & Management*. Springer, 85–98.

[16] Min Gao, Zhongfu Wu, and Feng Jiang. 2011. Userrank for item-based collaborative filtering recommendation. *Inform. Process. Lett.* 111, 9 (2011), 440–446.

[17] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. 2010. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 257–260.

[18] Daniel G Goldstein and Dominique C Goldstein. 2006. Profiting from the long tail. *Harvard Business Review* 84, 6 (2006), 24–28.

[19] Marco Gori, Augusto Pucci, V Roma, and I Siena. 2007. ItemRank: A Random-Walk Based Scoring Algorithm for Recommender Engines.. In *IJCAI*, Vol. 7. 2766–2771.

[20] GroupLens. 2015. MovieLens-1m Data Set. http://grouplens.org/datasets/movielens/1m/. (2015). Accessed: 03/2015.

[21] Jonathan L Herlocker, Joseph A Konstan, Loren G Terveen, and John T Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)* 22, 1 (2004), 5–53.

[22] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*. IEEE, 263–272.

[23] Péter Kovács. 2015. Minimum-cost flow algorithms: an experimental evaluation. *Optimization Methods and Software* 30, 1 (2015), 94–127.

[24] Cecilia Mazanec. 2016. Will Algorithms Erode Our Decision-Making Skills? (2016). http://www.npr.org/sections/alltechconsidered/2017/02/08/514120713/will-algorithms-erode-our-decision-making-skills Accessed: 02/2016.

[25] Sean M McNee, John Riedl, and Joseph A Konstan. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 Human factors in computing systems*. ACM, 1097–1101.

[26] Cathy O'Neil. 2016. *Weapons of math destruction: How big data increases inequality and threatens democracy*. Crown Publishing Group (NY).

[27] James B Orlin. 1997. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming* 78, 2 (1997), 109–129.

[28] Eli Pariser. 2011. *The filter bubble: How the new personalized web is changing what we read and how we think*. Penguin.

[29] Xiaolong Ren, Linyuan Lü, Runran Liu, and Jianlin Zhang. 2014. Avoiding congestion in recommender systems. *New Journal of Physics* 16, 6 (2014), 063057.

[30] Alexander Schrijver. 2002. *Combinatorial optimization: polyhedra and efficiency*. Vol. 24. Springer Science & Business Media.

[31] Sylvain Senecal and Jacques Nantel. 2004. The influence of online product recommendations on consumers online choices. *Journal of Retailing* 80, 2 (2004), 159–169.

[32] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 257–297.

[33] Zoltán Szlávik, Wojtek Kowalczyk, and Martijn Schut. 2011. Diversity Measurement of Recommender Systems under Different User Choice Models. In *Fifth International AAAI Conference on Weblogs and Social Media*.

[34] Endel Tulving, Hans J Markowitsch, Shitij Kapur, Reza Habib, and Sylvain Houle. 1994. Novelty encoding networks in the human brain: positron emission tomography data. *NeuroReport* 5, 18 (1994), 2525–2528.

[35] Saúl Vargas. 2015. Novelty and diversity enhancement and evaluation in Recommender Systems. (2015).

[36] Saúl Vargas. 2015. Novelty and Diversity Evaluation and Enhancement in Recommender Systems. (2015).

[37] Saúl Vargas and Pablo Castells. 2011. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proc. of the 5th ACM Conf. on Rec. Systems*. ACM, 109–116.

[38] Saúl Vargas and Pablo Castells. 2014. Improving sales diversity by recommending users to items. In *Proceedings of the 8th ACM Conference on Recommender systems*. ACM, 145–152.

[39] Mi Zhang and Neil Hurley. 2008. Avoiding monotony: improving the diversity of recommendation lists. In *Proc. of the 2008 ACM Conf. on Rec. Systems*. ACM, 123–130.

[40] Renjie Zhou, Samamon Khemmarat, and Lixin Gao. 2010. The impact of YouTube recommendation system on video views. In *Proc. of the 10th ACM SIGCOMM Conf. on Internet measurement*. ACM, 404–410.