Distributed Online Learning in Social Recommender Systems

Cem Tekin*, *Member, IEEE*, Simpson Zhang, Mihaela van der Schaar, *Fellow, IEEE*
Electrical Engineering Department, University of California, Los Angeles
Email: cmtkn@ucla.edu, simpsonzhang@ucla.edu, mihaela@ee.ucla.edu

***Abstract*—In this paper, we consider decentralized sequential decision making in distributed online recommender systems, where items are recommended to users based on their search query as well as their specific background including history of bought items, gender and age, all of which comprise the context information of the user. In contrast to centralized recommender systems, in which there is a single centralized seller who has access to the complete inventory of items as well as the complete record of sales and user information, in decentralized recommender systems each seller/learner only has access to the inventory of items and user information for its own products and not the products and user information of other sellers, but can get commission if it sells an item of another seller. Therefore the sellers must distributedly find out for an incoming user which items to recommend (from the set of own items or items of another seller), in order to maximize the revenue from own sales and commissions. We formulate this problem as a cooperative contextual bandit problem, analytically bound the performance of the sellers compared to the best recommendation strategy given the complete realization of user arrivals and the inventory of items, as well as the context-dependent purchase probabilities of each item, and verify our results via numerical examples on a distributed data set adapted based on Amazon data. Our results indicate that decentralized online recommendation systems can achieve performance close to centralized recommendation systems, while distributing the computation and learning over multiple learners. We also show that cooperating with other sellers increases the revenue of a seller compared to the case where it only recommends its own items to its own users. We evaluate the dependence of the performance of a seller on the inventory of items the seller has, the number of connections it has with the other sellers, and the commissions which the seller gets by selling items of other sellers to its users.**

***Index Terms*—Online learning, multi-agent learning, collaborative learning, regret based learning, distributed recommender systems, social networks, contextual bandits, similarity measure.**

I. Introduction

One of the most powerful benefits of a social network is the ability for cooperation and coordination on a large scale over a wide range of different agents [1]. By forming a network, agents are able to share information and opportunities in a mutually beneficial fashion. For example, companies can collaborate to sell products, charities can work together to raise money, and a group of workers can help each other search for jobs. Through such cooperation, agents are able to attain much greater rewards than would be possible individually. But sustaining efficient cooperation can also prove extremely challenging. First, agents operate with only incomplete information, and must learn the environment parameters slowly over time. Second, agents are decentralized and thus uncertain about their neighbor's information and preferences. Finally, agents are selfish and may opt not to reciprocate aid to their neighbors, favoring personal gain over social gain. This paper produces a class of mechanisms that effectively addresses all of these issues: at once allowing decentralized agents to take near-optimal actions in the face of incomplete information, while still incentivizing them to fully cooperate within the network.

The framework we consider is very broad and applicable to a wide range of social networking situations. We analyze a group of agents that are connected together via a fixed network, each of whom experiences inflows of users to its page. Each time a user arrives, an agent chooses from among a set of items to offer to that user, and the user will either reject or accept each item. These items can represent a variety of things, from a good that the agent is trying to sell, to a cause that the agent is trying to promote, to a photo that the agent is trying to circulate. In each application, the action of accepting or rejecting by the user will likewise have a distinct meaning. When choosing among the items to offer, the agent is uncertain about the user's acceptance probability of each item, but the agent is able to observe specific background information about the user, such as the user's gender, location, age, etc. Users with different backgrounds will have different probabilities of accepting each item, and so the agent must learn this probability over time by making different offers.

We allow for cooperation in this network by letting each agent recommend items of neighboring agents to incoming users, in addition to its own items. Thus if the specific background of the incoming user makes it unlikely for him to accept any of the agent's items, the agent can instead recommend him some items from a neighboring agent with more attractive offerings. By trading referrals in this fashion, all of the agents that are connected together can benefit. To provide proper incentives, a commission will be paid to the recommending agent every time an item is accepted by a user from the recommended agent. When defined appropriately, this commission ensures that both sides will benefit each time a recommendation occurs and thus is able to sustain cooperation.

However, since agents are decentralized, they do not directly share the information that they learn over time about user preferences for their own items. So when the decision to recommend a neighboring agent occurs, it is done based solely on the previous successes the agent had when recommending that neighbor. Thus agents must learn about their neighbor's acceptance probabilities through their own trial and error, unlike in other social learning papers such as [2]–[5], where agents share information directly with their neighbors.

Another key feature of our algorithm is that it is non-Bayesian unlike [2], [3]. Instead we model the learning through contextual bandits, where the context is based on the user's background. By building upon the theory of contextual bandits, we produce a class of mechanisms that allows agents to take near-optimal actions even with decentralized learning. We prove specific bounds for the regret, which is the difference between the total expected reward of an agent using a learning algorithm and the total expected reward of the optimal policy for the agent, which is computed given perfect knowledge about acceptance probabilities for each context. We show that the regret is sublinear in time in all cases. We further show that our mechanism can operate regardless of the specific network

structure and the degree of connectivity inside the network, although the performance is better if the network is more connected since each agent will have access to more items of other agents.

This work represents a significant departure from the other works in contextual bandits, which consider only centralized agents, single arms played at once, and no incentive issues. Most of the prior work on contextual bandits is focused on a single agent choosing one arm at a time based on the context information provided to it at each time step [6]–[9]. In these works the system is centralized, so the agent can directly access all the arms. Our framework in this paper differs from the centralized contextual bandit framework in two important aspects. First, multiple agents who can only access a subset of arms, and who only get feedback about this subset, cooperate to maximize their total reward. Second, each agent can choose multiple arms at each time step, which makes the arm selection problem combinatorial in the number of arms. To the best of our knowledge, our work is the first to provide rigorous solutions for online learning by multiple cooperative agents selecting multiple arms at each time step when context information is present. We had previously proposed a multi-agent contextual bandit framework in [10], [11] where each agent only selects a single arm at each time step. Different from these works, in this paper we assume that an agent can select multiple arms, and the expected reward of the agent from an arm may depend on the other selected arms. This makes the problem size grow combinatorially in the arm space, which requires the design of novel learning algorithms to quicken the learning process. Combinatorial bandits [12] have been studied before in the multi-armed bandit setting, but to the best of our knowledge we are the first to propose the decentralized contextual combinatorial bandit model studied in this paper. This decentralization is important because it allows us to analyze a social network framework and the fundamental challenges associated with it including commissions, third-party sellers, etc. We are also able to consider the specific effects of the network structure on the regret in our model. In contrast, our approach in [10] is purely theoretical and in [11], we address challenges specific to Big Data mining, all without the network structure concerns. Several other examples of related work in contextual bandits are [13], in which a contextual bandit model is used for recommending personalized news articles based on a variant of the UCB algorithm [14] designed for linear rewards. In [15] the authors solve a classification problem using contextual bandits, where they propose a perceptron based algorithm that achieves sublinear regret when the feedback about the rewards are binary and the rewards are generated by an adversary.

Apart from contextual bandits, there is a large set of literature concerned in multi-user learning using a multi-armed bandit framework [16]–[20]. We provide a detailed comparison between our work and related work in multi-armed bandit learning in Table I. Our cooperative contextual learning framework can be seen as an important extension of the centralized contextual bandits framework [6]. The main differences are that: (i) a three phase learning algorithm with *training*, *exploration* and *exploitation* phases is needed instead of the standard two phase algorithms with *exploration* and *exploitation* phases that are commonly used in centralized contextual bandit problems; (ii) the adaptive partitions of the context space should be formed in a way that each learner can efficiently utilize what is learned by other learners about the same context; (iii) since each agent has multiple selections at each time step, the set of actions for each agent is very large, making the learning rate slow. Therefore, the correlation between the arms of the agents should be exploited to quicken the learning process. In our distributed multi-agent multiple-play contextual bandit formulation, the training phase, which balances the learning rates of the agents, is necessary since the context arrivals to agents are different which makes the learning rates of the agents for various context different.

	[6]–[9]	[16], [20], [21]	[10], [11]	This work
Multi-agent	no	yes	yes	yes
Cooperative	N/A	yes	yes	yes
Contextual	yes	no	yes	yes
Context arrival process	arbitrary	N/A	arbitrary	arbitraty
synchronous(syn), asynchronous(asn)	N/A	syn	both	both
Regret	sublinear	logarithmic	sublinear	sublinear
Multi-play for each agent	no	no	no	yes
Combinatorial	no	no	no	yes
Action sets of the agents	N/A	same	different	different

TABLE I: Comparison with multi-armed bandit works

There is also an extensive literature on recommender systems that incorporates a variety of different methods and frameworks. Table II provides a summary of how our work is related to other work. Of note, there are several papers that also use a similar multi-armed bandit framework for recommendations. For example, [22] considers a bandit framework where a recommender system learns about the preferences of users over time as each user submits ratings. It uses a linear bandit model for the ratings, which are assumed to be functions of the specific user as well as the features of the product. It then proposes an algorithm that combines exploration and exploitation steps together in a continuous fashion, which results in good performance in both the short and the long run. [23] is another work that utilizes multi-armed bandits in a recommendation system. It considers a model that must constantly update recommendations as both preferences and the item set changes over time. Through its "Independent Bandit" Algorithm, the paper is able to show that keeping independent track of the acceptance probabilities of different items can prove better than only keeping tracking of which specific item was chosen first among a specific set of items.

There are also numerous examples of works that do not use a bandit framework for recommendations. One of the most commonly used methods for recommendations are collaborative filtering algorithms, which can include non-bandit algorithms. Collaborative filtering algorithms such as [24]–[30] make recommendations by predicting the user's preferences based on a similarity measure with other users. Items with the highest similarity score are then recommended to each user; for instance items may be ranked based on the number of purchases by similar users. There are numerous ways to perform the similarity groupings, such as the cluster model

	Item-based (IB), user-based (UB)	Memory-based, model-based	Uses context info.	Performance measure	Similarity distance	Centralized(C), Decentralized(D)
[32]	UB	Memory-based	No	Ranking precision	-	C
[24]	UB	Bayesian-based latent semantic model	No	MAE, RMS, 0/1 loss	Pearson correlation	C
[25]	UB	Bayesian-based Markov model	No	Precision& Recall	Pearson correlation	C
[26]	IB	Cluster model	No	-	Cosine	C
[27]	UB	Memory-based	Yes	Precision& Recall	-	C
[28]	UB	Bayesian classifier model	No	Precision& Recall	Pearson correlation	C
[29]	UB	Cluster model	No	MAE& Coverage	Pearson correlation	C
[30]	UB	MDP model	No	Recall	Self-defined similarity	C
[22]	UB	MAB model	No	Reward	Lipschitz continuous	C
[23]	UB	MAB model	Yes	Regret	Lipschitz continuous	C
Our work	UB	MAB model	Yes	Regret	Lipschitz continuous	D

TABLE II: Comparison with works in recommender systems in [26], [29] that groups users together with a set of like-minded users and then makes recommendations based on what the users in this set choose. Another possibility is presented in [27], which pre-filters ratings based on context before the recommendation algorithm is started.

An important difference to keep in mind is that the recommendation systems in other works are a single centralized system, such as Amazon or Netflix. Thus the system has complete information at every moment in time, and does not need to worry about incentive issues or pay commissions. However, in this paper each agent is in effect its own separate recommendation system, since agents do not directly share information with each other. Therefore the mechanism we propose must be applied separately by every agent in the system based on that agent's history of user acceptances. So in effect our model is a giant collection of recommendation systems that are running in parallel. The only cross-over between these different systems is when one agent suggests which of its items should be recommended by another agent. This allows for an indirect transfer of information, and lets that other agent make better choices than it could without this suggested list of items.

Also, it is important to note that decentralization in the context of our paper does not mean the same thing as in other papers such as [31]. Those papers assume that the $users$ are decentralized, and develop mechanisms based on trust and the similarity of different users in order to allow $users$ to recommend items to each other. We are assuming that the $agents$ are decentralized, and so each user still only gets a recommendation from one source, it is just that this source may be different depending on which agent this user arrives at. Thus this paper is fundamentally different from the works in that literature.

The rest of the paper is organized as follows. The problem formulation is given in Section II. In Section III, we consider the online learning problem involving multiple decentralized agents and analyze its regret. In Section III-A we develop an algorithm to achieve sublinear regret when the purchase probability of the items depend on the other recommended items, and in Section III-C we develop a faster learning algorithm when item purchase probabilities are independent of each other. The effect of connectivity between the agents is given in Section IV. In Section V, numerical results demonstrating the effects of commissions, size of the set of items of agents and connectivity of agents are given. Finally, we conclude the paper in Section VI.

II. Problem Formulation

There are M decentralized agents/learners which are indexed by the set $\mathcal{M} := \{1, 2, \ldots, M\}$. Each agent i has an inventory of items denoted by $\mathcal{F}_i$, which it can offer to its users and the users of other agents by paying some commission. For now, we will assume that all the agents in the network are connected, so that any agent can sell items to the users of any other agent. Let $\mathcal{F} := \cup_{i\in\mathcal{M}}\mathcal{F}_i$ be the set of items of all agents. We assume that there is an unlimited supply of each type of item. This assumption holds for digital goods such as e-books, movies, videos, songs, photos, etc. An agent does not know the inventory of items of the other agents but knows an upper bound on $|\mathcal{F}_j|$[1], $j \in \mathcal{M}$ which is equal to $F_{\max}$. Let $\mathcal{K}_i = \mathcal{F}_i \cup \mathcal{M}_{-i}$ be the set of *options* of agent i.

We note that our model is suitable for a broad range of applications. The agent can represent a company, an entrepreneur, a content provider, a blogger, etc., and the items can be goods, services, jobs, videos, songs, etc. The notion of an item can be generalized even further to include such things as celebrities that are recommended to be followed in Twitter, Google+, etc. And the prices that we introduce later can also be generalized to mean any type of benefit the agent can receive from a user. For expositional convenience, we will adopt the formulation of firms selling goods to customers for most of the paper, but we emphasize that many other interpretations are equally valid and applicable.

At each time step $t = 1, 2, \ldots$, a user with a specific search query indicating the type of item the user wants, or other information including a list of previous purchases, price-range, age, gender etc., arrives to agent i. We define all the properties of the arriving user known to agent i at time t as the context of that user, and denote it by $x_i(t)$. We assume that the contexts of all users belong to a known space $\mathcal{X}$, which without loss of generality is taken to be $[0, 1]^d$ in this paper, where d is the dimension of the context space. Most of our results in this paper will hold without any assumptions on the contexts of the users. Although the model we propose in this paper has synchronous arrivals, it can be easily extended to the asynchronous case where agents have different user arrival rates, and even when no user arrives in some time slots. The only difference of this from our framework is that instead of keeping the same time index t for every agent, we will have different time indices for each agent depending on the number of user arrivals to that agent.

In order to incentivize the agents to recommend each other's items, they will provide commissions to each other. In this

[1]For a set A, $|A|$ denotes its cardinality.

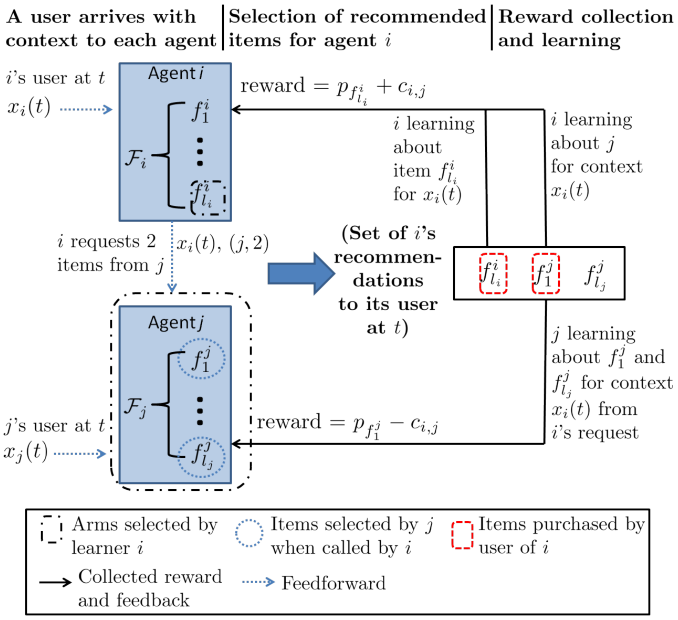

Fig. 1: Operation of the system for agent i for $N = 3$ recommendations. At each time a user arrives to agent i with context $x_i(t)$, agent i recommends a set of its own items and items from other agents.

paper we focus on sales commission, which is paid by the recommended agent to the recommending agent every time a user from the recommending agent purchases an item of the recommended agent. We assume that these commissions are fixed at the beginning and do not change over time. The system model is shown in Fig. 1. When there is sales commission, if agent i recommends an item f_j of agent j to its user, and if that user buys the item of agent j, then agent i obtains a fixed commission which is equal to $c_{i,j}$. All of our results in this paper will also hold for the case when the commission is a function of the price of the item f_j sold by agent j, i.e., $c_{i,j}(p_{f_j})$. However we use the fixed commission assumption, since privacy concerns may cause the agent j or the user to not want to reveal the exact sales price to agent i. Note that if these recommendations are made by the agents through an online marketplace like Amazon or eBay, an additional commission can be taken by these entities. For example, when a fixed price C is charged by the marketplace for every item sold, agent i's revenue from selling item $f_i \in \mathcal{F}_i$ will be $p_{f_i} - C$ and agent j's revenue from selling item $f_j \in \mathcal{F}_j$ to agent i's user will be $p_{f_j} - c_{i,j} - C$. The following analysis will still hold, as long as the recommendations continue to be decentralized and not performed by the marketplace entity itself.

Agent i recommends N items to its user at each time step. For example, N can be the number of recommendation slots the agent has on its website, or it can be the number of ads it can place in a magazine. We assume that N is fixed throughout the paper. These N items are chosen in the following way: An item can be chosen from the inventory of agent i, i.e., $\mathcal{F}_i$, or agent i can call another agent j and send the context information of the user $x_i(t)$, then agent j returns back an item f_j with price p_{f_j}[2] to be recommended to agent i based on the context information. Let $\mathcal{N}_i(t)$ be the set of items recommended by agent i to the user at time t. We will first consider the case when the user's purchase probability for an item $f \in \mathcal{N}_i(t)$ depends on $\mathcal{N}_i(t)$. Then, we will consider the case when the user's purchase probabilities of recommended items are independent of each other. In general, the purchase probability of the item will depend on the user's context as well as the set of items recommended along with that item. Let $\mathcal{A}_N$ be the set of subsets of $\mathcal{F}$ with N items. Let $\mathcal{N} \in \mathcal{A}_N$ be a set of N recommended items.

Assumption 1: **Dependent purchase probability:** For each item f recommended together with the set of items $\mathcal{N} \in \mathcal{A}_N$, a user with context $x \in \mathcal{X}$ will buy the item with an unknown probability $q_f(x, \mathcal{N})$ for which there exists $L > 0$, $\alpha > 0$ such that for all $x, x' \in \mathcal{X}$, we have $|q_f(x, \mathcal{N}) - q_f(x', \mathcal{N})| \leq L||x - x'||^\alpha$, where $||.||$ denotes the Euclidian norm in $\mathbb{R}^d$.

Assumption 1 indicates that the purchasing probability of an item for users with similar contexts will be similar to each other. However, we do not require the purchase probabilities to be similar for different sets of recommendations $\mathcal{N}$ and $\mathcal{N}' \in \mathcal{A}_N$. Even though the Lipschitz condition can hold with different constants L_f and α_f for each item $f \in \mathcal{F}$, taking L to be the largest among L_f and α to be the smallest among α_f we get the condition in Assumption 1. For example, the context can be the age of the user, and users with ages similar to each other can like similar items. In order to learn the best set of items to recommend for each user, the agents need to estimate the purchase probability of an item separately for each set of items that the item is recommended together with. We will also consider a simplified version, in which the purchase probability of an item is independent of the set of other items recommended along with that item.

Assumption 2: **Independent purchase probability:** For each item f offered along with the items in the set $\mathcal{N} \in \mathcal{A}_N$, a user with context x will buy the item with an unknown probability $q_f(x)$, independent of the other items in $\mathcal{N}_i(t)$, for which there exists $L > 0$, $\alpha > 0$ such that for all $x, x' \in \mathcal{X}$, we have $|q_f(x) - q_f(x')| \leq L||x - x'||^\alpha$, where $||.||$ denotes the Euclidian norm in $\mathbb{R}^d$.

When Assumption 2 holds, the agents can estimate the purchase probability of an item by using the empirical mean of the number of times the item is purchased by users with similar context information. The purchase probabilities of items can be learned much faster in this case compared with Assumption 1, since the same empirical mean of the number of purchases will be updated for an item every time the item is recommended.

The goal of agent i is to maximize its total expected revenue. The expected revenue of agent i at time t from recommending a set of items $\mathcal{N}_i$ is given by $\sum_{f \in \mathcal{N}_i - \mathcal{F}_i} c_{i,j(f)} q_f(x_i(t), \mathcal{N}_i) + \sum_{f \in \mathcal{N}_i - (\mathcal{N}_i - \mathcal{F}_i)} p_f q_f(x_i(t), \mathcal{N}_i)$, where $j(f)$ is the agent who owns item f. Given a user with context x, the set of items which maximizes the one-step expected reward of agent

[2]If agent j does not want to reveal the price to agent i, then the recommendation rule can be modified as follows: Agent j's item will be recommended by agent i without a price tag, and when the user clicks to agent j's item it will be directed to agent j's website where the price will be revealed to the user.

i is $\mathcal{N}_i^*(x) := \arg\max_{\mathcal{N}\in\mathcal{A}_N} \sum_{f\in\mathcal{N}-\mathcal{F}_i} c_{i,j(f)} q_f(x,\mathcal{N}) + \sum_{f\in\mathcal{N}-(\mathcal{N}-\mathcal{F}_i)} p_f q_f(x,\mathcal{N})$. Since the inventory of other agents and $q_f(x,\mathcal{N}), x\in\mathcal{X}, \mathcal{N}\in\mathcal{A}_N$ are unknown a priori to agent i, $\mathcal{N}_i^*(x)$ is unknown to agent i for all contexts $x\in\mathcal{X}$.

The set of actions available to agent i at any time step is the pair (u_i, n_{u_i}), where u_i denotes the item to be recommended for $u_i \in \mathcal{F}_i$ or another agent to be recommended for $u_i \in \mathcal{M}_{-i}$, and n_{u_i} denotes whether item u_i is recommended ($n_{u_i}=1$) or not ($n_{u_i}=0$) for $u_i \in \mathcal{F}_i$, or how many distinct items agent u_i should recommend to agent i for $u_i \in \mathcal{M}_{-i}$. Based on this, let $\mathcal{L}_i = \{(u_i, n_{u_i}) \in \mathcal{F}_i \times \{0,1\}$ or $(u_i, n_{u_i}) \in \mathcal{M}_{-i} \times \{0,1,\ldots,N\}$ such that $\sum_{u_i\in\mathcal{K}_i} n_{u_i} = N\}$, be the set of actions available to agent i. We assume that $|\mathcal{F}_j| \geq N$ for all $j\in\mathcal{M}$.

Let α_i be the recommendation strategy adopted by agent i for its own users, i.e., based on its past observations and decisions, agent i chooses a vector $\alpha_i(t) \in \mathcal{L}_i$ at each time step. Let β_i be the recommendation strategy adopted by agent i when it is called by another agent to recommend its own items. Let $\boldsymbol{\alpha} = (\alpha_1,\ldots,\alpha_M)$ and $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_M)$. Let $Q^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$ be the expected total reward of agent i by time T from item sales to its own users and users of other agents and commissions it gets from item sales of other agents to its own users. Let $S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T) := E_{\boldsymbol{\alpha},\boldsymbol{\beta}}\left[\sum_{t=1}^T \sum_{f_i\in\mathcal{F}_i} p_{f_i} q_{f_i}(x_i(t),\mathcal{N}_i(t)) I(n^{\alpha_i}_{f_i}(t)=1)\right] + E_{\boldsymbol{\alpha},\boldsymbol{\beta}}\left[\sum_{t=1}^T \sum_{j\in\mathcal{M}_{-i}} I(n^{\alpha_i}_j(t)>0)\left(\sum_{f_j\in\mathcal{F}_j} c_{i,j}\right.\right.$ $\left.\left. q_{f_j}(x_i(t),\mathcal{N}_i(t)) I(n^{\beta_j}_{f_j}(t)=1)\right)\right]$, where $n^{\beta_j}_{f_j}(t) = 1$, when j recommends its item f_j to agent i when called by agent i, and 0 otherwise. $S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$ is the total expected reward agent i can get based only on recommendations to its own users.

We can see that $Q^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T) \geq S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$. Agent i's goal is to maximize its total reward $S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$ from its own users for any T. Since agents are cooperative agent i also helps other agents $j\in\mathcal{M}_{-i}$ to maximize $S^j_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$ by recommending its items to them. Hence the total reward the agent gets, $Q^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$, is at least $S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$.

We assume that user arrivals to the agents are independent of each other. Therefore, agent j will also benefit from agent i if its item can be sold by agent i. In this paper, we develop distributed online learning algorithms for the agents in $\mathcal{M}$, i.e., $(\alpha_i,\beta_i)_{i\in\mathcal{M}}$ such that the expected total reward for any agent $S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T)$ is maximized for all $i\in\mathcal{M}$. In other words, we define the regret of agent i to be

$$R_i(T) := \sum_{t=1}^{T} \sum_{f\in\mathcal{N}_i^*(x_i(t))-\mathcal{F}_i} c_{i,j} q_f(x_i(t)) + \sum_{f\in\mathcal{F}_i-(\mathcal{N}_i^*(x_i(t)))-\mathcal{F}_i)} p_f q_f(x_i(t),\mathcal{N}_i^*(x_i(t))) - S^i_{\boldsymbol{\alpha},\boldsymbol{\beta}}(T), \quad (1)$$

and design online learning algorithms that will minimize the regret. Note that the regret is calculated with respect to the highest expected reward agent i can obtain from its own users, but not the users of other agents. Therefore, agent i does not act strategically to attract the users of other agents, such as by cutting its own prices or paying commissions even when an item is not sold to increase its chance of being recommended by another agent. We assume that agents are fully cooperative and follow the rules of the proposed algorithm.

We will show that the regret of the algorithms proposed in this paper will be sublinear in time, which means that the distributed learning scheme converges to the average reward of the best recommender strategy $\mathcal{N}_i^*(x)$ for each $i\in\mathcal{M}$, $x\in\mathcal{X}$. Moreover, the regret also provides us with a bound on how fast our algorithm converges to the best recommender strategy.

III. Contextual Partitioning Algorithms for Multiple Recommendations

In this section we propose a distributed online learning algorithm called *context based multiple recommendations* (CBMR) algorithm. Basically, an agent using CBMR forms a partition of the context space $[0,1]^d$, depending on the final time T, consisting of $(m_T)^d$ sets where each set is a d-dimensional hypercube with dimensions $1/m_T \times 1/m_T \times \ldots \times 1/m_T$. The sets in this partition are indexed by $\mathcal{I}_T = \{1,2,\ldots,(m_T)^d\}$. We denote the set with index l with I_l. Agent i learns the purchase probability of the items in each set in the partition independently from the other sets in the partition based on the context information of the users that arrived to agent i and the users for which agent i is recommended by another agent. Since users with similar contexts have similar purchase probabilities, it is expected that the optimal recommendations are similar for users located in the same set in $\mathcal{I}_T$. Since the best recommendations are learned independently for each set in $\mathcal{I}_T$, there is a tradeoff between the number of sets in $\mathcal{I}_T$ and the estimation of the best recommendations for contexts in each set in $\mathcal{I}_T$. We will show that in order to bound regret sublinearly over time, the parameter m_T should be non-decreasing in T. There are two versions of CBMR: one for general purchase probabilities given by Assumption 1, i.e., CBMR-d and the other for independent purchase probabilities given by Assumption 2, i.e., CBMR-ind. The difference between these two is that CBMR-d calculates the expected reward from each action in $\mathcal{L}_i$ for agent i separately, while CBMR-ind forms the expected reward of each action in $\mathcal{L}_i$ based on the expected rewards of the items recommended in the chosen action. Usually $|\mathcal{L}_i|$ is exponential in $|\mathcal{K}_i|$, and thus CBMR-ind provides much faster learning than CBMR-d when Assumption 2 holds. We explain these algorithms in the following subsections.

A. Context based multiple recommendations for dependent purchase probabilities (CBMR-d)

The pseudocode of CBMR-d is given in Fig. 2. The set of actions available to agent i is given by $\mathcal{L}_i$. The action $k_i\in\mathcal{L}_i$ chosen by agent i induces a set of recommendations $\mathcal{N}_i(t) \in \mathcal{A}_N$ along with the items recommended by the other agents called by agent i when agent i takes action k_i, i.e., $\mathcal{M}_i(k_i)$. The cardinality of $\mathcal{L}_i$ is combinatorial in $\mathcal{F}_i$ and $\mathcal{M}_{-i}$. CBMR-d can be in any of the following three phases at any time step: the *training* phase in which agent i trains another agent j by asking it for recommendations and providing i's user context information x and i's action k_i so that agent j will learn to recommend its items with the highest

probability of being purchased, the *exploration* phase in which agent i updates the estimated reward from the set of actions in $\mathcal{L}_i$ by selecting it, and the *exploitation* phase in which agent i selects the action in $\mathcal{L}_i$ with the highest estimated reward. The pseudocodes of these phases are given in Fig. 3. In all these phases, when agent i requests recommendations agent j, it sends its user's context information and the arm its selected $k_i \in \mathcal{L}_i$ to agent j.

At each time t, agent i first checks which set in the partition $\mathcal{I}_T$ $x_i(t)$ belong to. Let $N_l^i(t)$ be the number of users who arrived to agent i with contexts in the lth set of the partition of agent i by time t. We separate the set of actions in $\mathcal{L}_i$ into two. Let $\hat{\mathcal{L}}_i = \{(k, n_k) \in \mathcal{F}_i \times \{0,1\}$ or $(k, n_k) \in \mathcal{M}_{-i} \times \{0\}$ such that $\sum_{k \in \mathcal{K}_i} n_k = N\}$, be the set of actions in which all recommendations belong to agent i and $\tilde{\mathcal{L}}_i = \mathcal{L}_i - \hat{\mathcal{L}}_i$, be the set of actions in which at least one recommendation comes from an agent in $\mathcal{M}_{-i}$. We have $|\mathcal{L}_i| = \sum_{n=0}^{N} \binom{|\mathcal{F}_i|}{n}(M-1)^{N-n}$, which grows exponentially in N and polynomially in M and $|\mathcal{F}_i|$.

The training phase is required for actions in $\tilde{\mathcal{L}}_i$, while only exploration and exploitation phases are required for actions in $\hat{\mathcal{L}}_i$. When agent i chooses an action $k_i \in \tilde{\mathcal{L}}_i$, the agents $j \in \mathcal{M}_{-i}$ for which $n_j(k_i) > 0$ recommend $n_j(k_i)$ items from their own set of items to agent i. Recall that $\mathcal{N}_i(t)$ denotes the set of N items that is recommended to agent i's user at time t, based on the actions chosen by agent i and the recommendations chosen by other agents for agent i. Therefore, the reward agent i gets at time t is $Q_i(t) = S_i(t) + O_i(t)$, where $S_i(t) := \sum_{f \in \mathcal{N}_i(t) - \mathcal{F}_i} c_{i,j(f)} I(f \in F_i(t)) + \sum_{f \in \mathcal{N}_i(t) - (\mathcal{N}_i(t) - \mathcal{F}_i)} p_f I(f \in F_i(t))$, and $O_i(t) := \sum_{j \in \mathcal{M}_{-i}} O_{i,j}(t)$, where $O_{i,j}(t) := \sum_{f \in \mathcal{F}_i \cap \mathcal{N}_j(t)} (p_f - c_{j,i}) I(f \in F_j(t))$, $F_i(t)$ is the set of items purchased by agent i's user at time t.

For $k_i \in \hat{\mathcal{L}}_i$, let $N_{k_i,l}^i(t)$ be the number of times action k_i is selected in response to a context arriving to the set I_l in agent i's partition $\mathcal{I}_T$ of $\mathcal{X}$ by time t. Note that agent i does not know anything about agent j's set of available items $\mathcal{F}_j$. Therefore, before forming estimates about the rewards of recommendations of agent j, it needs to make sure that j will almost always recommend an item which has the highest probability of being purchased by the user of agent i. This is why the training phase is needed for actions in $\tilde{\mathcal{L}}_i$. It is very important to distinguish between cooperative agents which we assume in this paper and strategic agents, which we provide a comparison for in the following remark.

Remark 1: In our cooperative algorithms, an agent j when called by agent i recommends a set of items that has the highest probability of being purchased by agent i's user. This recommendation may only improve the performance of agent j, compared to the case when j does not cooperate with any other agent, since $p_{f_j} \geq c_{i,j}$ for all $f_j \in \mathcal{F}_j$ and $i \in \mathcal{M}_{-j}$. Moreover, the long term benefit to agent j of cooperating with agent i is that agent j also learns about the purchase probabilities of its own items from agent i's users, and thus it learns at a faster rate than without cooperation. However, when the commission is fixed, recommending the set of items with the highest probability of being purchased does not always maximize j's reward. For example, j may have another item which has a slightly smaller probability of being purchased by agent i's user, but has a much higher price than the item which maximizes the purchase probability. Then, it is of interest to agent j to recommend that item to agent i rather than the item that maximizes the purchase probability. This problem can be avoided by charging a commission which is equal to a percentage of the sales price of the item, i.e., $c_{i,j}(p_{f_j}) = c_j p_{f_j}$ for some $0 < c_j < 1$ for $f_j \in \mathcal{F}_j$. Our theoretical results will hold for this case as well.

In order to separate training, exploration and exploitation phases, agent i keeps two counters for actions $k_i \in \tilde{\mathcal{L}}_i$. The first one, i.e., $N_{1,k_i,l}^i(t)$, counts the number of context arrivals to agent i in set l by time t which are also sent to other agents $j \in \mathcal{M}_{-i}$ for which $n_j(k_i) > 0$ in the training phases of i. The second one, i.e., $N_{2,k_i,l}^i(t)$, counts the number of context arrivals to agent i in set l by time t which are used to estimate the expected reward of agent i from taking action k_i.

When a user with context $x_i(t) \in I_l$ arrives at time t, in order to make sure that the purchase probabilities of all the items of all agents are explored sufficiently for each action in $\mathcal{L}_i$ agent i checks if the following set is nonempty: $\mathcal{S}_{i,l}(t) := \Big\{k_i \in \hat{\mathcal{L}}_i$ such that $N_{k_i,l}^i(t) \leq D_1(t)$ or $k_i \in \tilde{\mathcal{L}}_i$ such that $N_{1,k_i,l}^i(t) \leq D_{2,k_i}(t)$ or $N_{2,k_i,l}^i(t) \leq D_{3,k_i}(t)\Big\}$. For $k_i \in \tilde{\mathcal{L}}_i$, let $\mathcal{E}_{k_i,l}^i(t)$ be the set of rewards collected from selections of action k_i when agent i's user's context is in set I_l, and all the other agents in $\mathcal{M}_{-i}$ are trained sufficiently, i.e., $N_{1,k_i,l}^i(t) > D_{2,k_i}(t)$, by time t. For $k_i \in \hat{\mathcal{L}}_i$, let $\mathcal{E}_{k_i,l}^i(t)$ be the set of rewards collected from action k_i by time t. If $\mathcal{S}_{i,l}(t) \neq \emptyset$, then agent i explores by randomly choosing an action $\alpha_i(t) \in \mathcal{S}_{i,l}(t)$. If $\mathcal{S}_{i,l}(t) = \emptyset$, this implies that all actions in $\mathcal{L}_i$ have been explored and trained sufficiently, so that agent i exploits by choosing the action with the highest sample mean estimate, i.e., $\alpha_i(t) \in \arg\max_{k_i \in \mathcal{L}_i} \bar{r}_{k_i,l}^i(t)$, where $\bar{r}_{k_i,l}^i(t)$ is the sample mean of the rewards in set $\mathcal{E}_{k_i,l}^i(t)$. We have $\bar{r}_{k,l}^i(t) = (\sum_{r \in \mathcal{E}_{k,l}^i(t)} r)/|\mathcal{E}_{k,l}^i(t)|$. When there is more than one action which has the highest sample mean estimate, one of them is randomly selected. The exploration control functions $D_1(t)$, $D_{2,k_i}(t)$ and $D_3(t)$, $k_i \in \tilde{\mathcal{L}}_i$ ensure that each action is selected sufficiently many times so that the sample mean estimates $\bar{r}_{k,l}^i(t)$ are accurate enough.

The other learning algorithm, β_i, gives agent i the set of items to recommend to agent j when agent j takes action $k_j \in \mathcal{L}_j$, which calls agent i to recommend at least one item. In order to recommend the set of items with the maximum probability of being purchased by the user of agent j, agent i should learn the purchase probability of its own items for k_j. Let $\mathcal{B}_i(n)$ be the set of n item subsets of $\mathcal{F}_i$. When $x_j(t) \in I_l$, agent i will explore a set of items in the set $O_{i,l,k_j}(t) := \{\boldsymbol{f}_i \in \mathcal{B}_i(n_i(k_j))$ such that $N_{k_j,l,\boldsymbol{f}_i}^i \leq D_3(t)\}$, if this set is nonempty. Otherwise, agent i will exploit the best set of items in $\mathcal{B}_i(n_i(k_j))$ that maximizes the purchase probability for agent j, i.e., $\boldsymbol{f}_{i,k_j,x}^* := \arg\max_{\boldsymbol{f}_i \in \mathcal{B}_i(n_i(k_j))} \sum_{f \in \boldsymbol{f}_i} q_f(x, \boldsymbol{f}_j(k_j))$, where $\boldsymbol{f}_j(k_j)$ is the set of items in $\mathcal{F}_j$ recommended by agent j to j's own

user when agent j takes action $k_j \in \mathcal{L}_j$. In the following subsection we prove an upper bound on the regret of CBMR.

Context Based Multiple Recommendations for dependent purchase probabilities (CBMR-d for agent i):

Input: $D_1(t)$, $D_{2,k_i}(t)$, $k_i \in \mathcal{L}_i$, $D_3(t)$, T, m_T
Initialize: Partition $[0,1]^d$ into $(m_T)^d$ sets, indexed by the set $\mathcal{I}_T = \{1, 2, \ldots, (m_T)^d\}$. $N^i_{k,l} = 0, \forall k \in \hat{\mathcal{L}}_i, l \in \mathcal{I}_T$, $N^i_{1,k,l} = 0$, $N^i_{2,k,l} = 0$, $\forall k \in \tilde{\mathcal{L}}_{-i}, l \in \mathcal{I}_T$.
while $t \geq 1$ **do**
  Algorithm α_i (Send recommendations to own users)
  for $l = 1, \ldots, (m_T)^d$ **do**
    if $x_i(t) \in I_l$ **then**
      if $\exists k \in \hat{\mathcal{L}}_i$ such that $N^i_{k,l} \leq D_1(t)$ **then**
        Run **Explore**(k, $N^i_{k,l}$, $\bar{r}^i_{k,l}$)
      else if $\exists k \in \tilde{\mathcal{L}}_{-i}$ such that $N^i_{1,k,l} \leq D_{2,k}(t)$ **then**
        Run **Train**(k, $N^i_{1,k,l}$)
      else if $\exists k \in \tilde{\mathcal{L}}_{-i}$ such that $N^i_{2,k,l} \leq D_3(t)$ **then**
        Run **Explore**(k, $N^i_{2,k,l}$, $\bar{r}^i_{k,l}$)
      else
        Run **Exploit**($((N^i_{k,l})_{k\in\hat{\mathcal{L}}_i}, (N^i_{2,k,l})_{k\in\tilde{\mathcal{L}}_i}$, $\bar{\boldsymbol{r}}^i_l$, $\mathcal{L}_i$)
      end if
    end if
  end for
  Algorithm β_i (Send recommendations to other agents' users)
  for $j \in \mathcal{M}_{-i}$ **do**
    receive $k_j \in \mathcal{L}_j$ and $x_j(t)$
    for $l = 1, \ldots, (m_T)^d$ **do**
      if $x_i(t) \in I_l$ **then**
        If k_j is observed for the first time for a user with context $x_j(t)$, set $N^i_{k_j,\boldsymbol{f}_i,l} = 0$, $\forall \boldsymbol{f}_i \in \mathcal{B}_i(n_i(k_j))$.
        if $\exists\ N^i_{k_j,\boldsymbol{f}_i,l} \leq D_1(t)$ such that $\boldsymbol{f}_i \in \mathcal{B}_i(n_i(k_j))$ **then**
          Run **Explore2**($\boldsymbol{f}_i$, $N^i_{k_j,\boldsymbol{f}_i,l}$, $\bar{r}^i_{k_j,\boldsymbol{f}_i,l}$)
        else
          Run **Exploit2**($\mathcal{B}_i(n_i(k_j))$, $(\bar{r}^i_{k_j,\boldsymbol{f}_i,l})_{\boldsymbol{f}_i\in\mathcal{B}_i(n_i(k_j))}$, $(N^i_{k_j,\boldsymbol{f}_i,l})_{\boldsymbol{f}_i\in\mathcal{B}_i(n_i(k_j))}$)
        end if
      end if
    end for
  end for
  $t = t + 1$
end while

Fig. 2: Pseudocode for the CBMR-d algorithm.

B. Analysis of the regret of CBMR-d

Let $\mu_{i,k_i}(x)$ be the expected reward agent i gets from the optimal set of recommendations $\mathcal{N}^*_i(x)$ for context x, $k_i \in \mathcal{L}_i$, and let $\log(.)$ denote logarithm in base e. For each $I_l \in \mathcal{I}_T$ let $\overline{\mu}_{i,k_i,l} := \sup_{x\in I_l} \mu_{i,k_i}(x)$ and $\underline{\mu}_{i,k_i,l} := \inf_{x\in I_l} \mu_{i,k_i}(x)$, for $k_i \in \mathcal{L}_i$. Let x^*_l be the context at the center of the hypercube I_l. We define the optimal action of agent i for set I_l as $k^*_i(l) := \arg\max_{k_i\in\mathcal{L}_i} \mu_{i,k_i}(x^*_l)$. Let $\mathcal{U}^i_{\theta,l}(t) := \left\{k_i \in \mathcal{L}_i \text{ such that } \underline{\mu}_{i,k^*_i(l),l} - \overline{\mu}_{i,k_i,l} > a_1 t^\theta\right\}$, be the set of suboptimal actions for agent i at time t, where $\theta < 0$, $a_1 > 0$. The agents are not required to know the values of the parameters θ and a_1. They are only used in our analysis of the regret. First, we will give regret bounds that depend on values of θ and a_1, and then we will optimize over these values to find the best bound. Let $Y_R := \sum_{f\in\mathcal{Y}_R} p_f$, where $\mathcal{Y}_R$ is the set of N items in $\mathcal{F}$ with the highest prices. Every time a suboptimal action is chosen by an agent, the one step regret is upper bounded by Y_R.

Train(k, n):
Select action k. Receive reward $\tilde{r}_k(t) = S_i(t)$. $n++$.
Explore(k, n, r):
Select action k. Receive reward $\tilde{r}_k(t) = S_i(t)$. $r = \frac{nr+\tilde{r}_k(t)}{n+1}$. $n++$.
Exploit($\boldsymbol{n}$, $\boldsymbol{r}$, $\mathcal{K}_i$):
Select action $k \in \arg\max_{k'\in\mathcal{L}_i} r_j$. If $k \in \tilde{\mathcal{L}}_i$, ask learners in $\mathcal{M}_i(k_i)$ to send their recommendations. Recommend items $\mathcal{N}_i(t)$. Receive reward $\tilde{r}_k(t) = S_i(t)$. $\bar{r}_k = \frac{n_k\bar{r}_k+\tilde{r}_k(t)}{n_k+1}$. n_k++.
Explore2($\boldsymbol{f}$, n, r):
Recommend set of items $\boldsymbol{f}$. Receive feedback $\tilde{r}^\beta(t) = \sum_{f\in\boldsymbol{f}_i} I(f \in F_j(t))$. $r = \frac{nr+\tilde{r}^\beta(t)}{n+1}$. $n++$.
Exploit2($\mathcal{B}$, $\boldsymbol{r}$, $\boldsymbol{N}$):
Recommend set of items $\boldsymbol{f}^* = \arg\max_{\boldsymbol{f}\in\mathcal{B}} r_{\boldsymbol{f}}$. Receive feedback $\tilde{r}^\beta(t) = \sum_{f\in\boldsymbol{f}^*} I(f \in F_j(t))$. $r_{\boldsymbol{f}} = \frac{n_{\boldsymbol{f}^*} r_{\boldsymbol{f}} + \tilde{r}^\beta_k(t)}{n_{\boldsymbol{f}^*}+1}$. $n_{\boldsymbol{f}^*}++$.

Fig. 3: Pseudocode of the training, exploration and exploitation modules.

The regret given in (1) can be written as a sum of three components: $R_i(T) = E[R^e_i(T)] + E[R^s_i(T)] + E[R^n_i(T)]$, where $R^e_i(T)$ is the regret due to training and explorations by time T, $R^s_i(T)$ is the regret due to suboptimal action selections in exploitations by time T, and $R^n_i(T)$ is the regret due to near optimal action selections in exploitations by time T of agent i, which are all random variables. In the following lemmas we will bound each of these terms separately. The following lemma bounds $E[R^e_i(T)]$. Due to space limitations we give the lemmas in this and the following subsections without proofs. The proofs can be found in our online appendix [33].

Lemma 1: When CBMR-d is run with parameters $D_1(t) = D_3(t) = t^z \log t$, $D_{2,k_i}(t) = \max_{j\in\mathcal{M}_{-i}} \left(\binom{F_{\max}}{n_j(k_i)}\right) t^z \log t$, and $m_T = \lceil T^\gamma \rceil$[3], where $0 < z < 1$ and $0 < \gamma < 1/d$, we have $E[R^e_i(T)] \leq Y_R 2^d \left(|\mathcal{L}_i| + |\tilde{\mathcal{L}}_i| \binom{F_{\max}}{\lceil F_{\max}/2 \rceil}\right) T^{z+\gamma d} \log T + Y_R 2^d |\mathcal{L}_i| T^{\gamma d}$.

Proof: We sum over all exploration and training steps by time T. The contribution to the regret is at most Y_R in each of these steps. ∎

From Lemma 1, we see that the regret due to explorations is linear in the number of hypercubes $(m_T)^d$, hence exponential in parameters γ and z. We conclude that z and γ should be small enough to achieve sublinear regret in exploration phases.

For any $k_i \in \mathcal{L}_i$ and $I_l \in \mathcal{I}_T$, the sample mean $\bar{r}^i_{k_i,l}(t)$ represents a random variable which is the average of the independent samples in set $\mathcal{E}^i_{k_i,l}(t)$. Different from classical finite-armed bandit theory [14], these samples are not identically distributed. In order to facilitate our analysis of the regret, we generate two different artificial i.i.d. processes to bound the probabilities related to $\bar{r}^i_{k,l}(t)$, $k \in \mathcal{L}_i$. The first one is the *best* process in which rewards are generated according to a bounded i.i.d. process with expected reward $\overline{\mu}_{i,k,l}$, the other one is the *worst* process in which rewards are generated according to a bounded i.i.d. process with expected reward $\underline{\mu}_{i,k,l}$. Let $r^{\text{best}}_{k,l}(\tau)$ denote the sample mean of the τ samples from the best process and $r^{\text{worst}}_{k,l}(\tau)$ denote the sample mean of

[3]For a number $r \in \mathbb{R}$, let $\lceil r \rceil$ be the smallest integer that is greater than or equal to r.

the τ samples from the worst process. We will bound the terms $E[R_i^n(T)]$ and $E[R_i^s(T)]$ by using these artificial processes along with the Lipschitz condition given in Assumption 1. The following lemma bounds $E[R_i^s(T)]$.

Lemma 2: When CBMR-d is run with parameters $D_1(t) = D_3(t) = t^z \log t$, $D_{2,k_i}(t) = \max_{j\in\mathcal{M}_{-i}} \left(\binom{F_{\max}}{n_j(k_i)}\right) t^z \log t$, and $m_T = \lceil T^\gamma \rceil$, where $0 < z < 1$ and $0 < \gamma < 1/d$, given that $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R+2)t^{-z/2} \leq a_1 t^\theta$ we have $E[R_i^s(T)] \leq 2^{d+1} Y_R |\hat{\mathcal{L}}_i| \beta_2 T^{\gamma d} + 2^{d+2} Y_R |\tilde{\mathcal{L}}_i| \beta_2 T^{\gamma d + z/2}/z$.

Proof: This proof is similar to the proof of Lemma 2 in [10]. The difference is that instead of bounding the probabilities that a suboptimal arm in $\mathcal{F}_i$ will be chosen or another agent $j \in \mathcal{M}_{-i}$ chooses a suboptimal arm in $\mathcal{F}_j$ when called by agent i, we bound the probabilities that a suboptimal action in $\hat{\mathcal{L}}_i$ is chosen and a suboptimal action in $\tilde{\mathcal{L}}_i$ is chosen. Another difference is that every time a suboptimal action is chosen, the one-step regret can be at most Y_R. ∎

From Lemma 2, we see that the regret increases exponentially with parameters γ and z, similar to the result of Lemma 1. These two lemmas suggest that γ and z should be as small as possible, given that the condition $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R + 2)t^{-z/2} \leq a_1 t^\theta$, is satisfied.

Each time agent i selects action k_i for which $n_j(k_i) > 0$, agent j recommends $n_j(k_i)$ items from $\mathcal{F}_j$ to agent i There is a positive probability that agent j will recommend suboptimal items, which implies that even if agent j's best items are near optimal for agent i, selecting agent j may not yield a near optimal outcome. We need to take this into account, in order to bound $E[R_i^n(T)]$. The following lemma gives the bound on $E[R_i^n(T)]$.

Lemma 3: When CBMR-d is run with parameters $D_1(t) = D_3(t) = t^z \log t$, $D_{2,k_i}(t) = \max_{j\in\mathcal{M}_{-i}} \left(\binom{F_{\max}}{n_j(k_i)}\right) t^z \log t$, and $m_T = \lceil T^\gamma \rceil$, where $0 < z < 1$ and $0 < \gamma < 1/d$, given that $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R + 2)t^{-z/2} \leq a_1 t^\theta$, we have $E[R_i^n(T)] \leq (2a_1 T^{1+\theta})/(1+\theta) + 2Y_R \binom{F_{\max}}{\lceil F_{\max}/2 \rceil} \beta_2$.

Proof: If a near optimal action in $\hat{\mathcal{L}}_i$ is chosen at time t, the contribution to the regret is at most $a_1 t^\theta$. If a near optimal action $k_i \in \tilde{\mathcal{L}}_i$ is chosen at time t, and if some of the agents in $\mathcal{M}_i(k_i)$ choose their near optimal items to recommend to agent i, then the contribution to the regret is at most $2a_1 t^\theta$. Therefore, the total regret due to near optimal action selections in $\mathcal{L}_i$ by time T is upper bounded by $2a_1 \sum_{t=1}^T t^\theta \leq (2a_1 T^{1+\theta})/(1+\theta)$, by using the result in Appendix A in [10]. Each time a near optimal action in $k_i \in \tilde{\mathcal{L}}_i$ is chosen in an exploitation step, there is a small probability that an item chosen by some agent $j \in \mathcal{M}_i(k_i)$ is a suboptimal one. Using a result similar to Lemma 3 in [10], the expected number of times a suboptimal arm is chosen is bounded by $2\binom{F_{\max}}{\lceil F_{\max}/2 \rceil}\beta_2$. Each time a suboptimal arm is chosen, the regret can be at most Y_R. ∎

From Lemma 3, we see that the regret due to near optimal actions depends exponentially on θ which is related to the negative of γ and z. Therefore γ and z should be chosen as large as possible to minimize the regret due to near optimal actions. Combining the above lemmas, we obtain the finite time, uniform regret bound given in the following theorem.

Theorem 1: Let CBMR-d run with exploration control functions $D_1(t) = D_3(t) = t^{2\alpha/(3\alpha+d)} \log t$, $D_{2,k_i}(t) = \max_{j\in\mathcal{M}_{-i}} \left(\binom{F_{\max}}{n_j(k_i)}\right) t^{2\alpha/(3\alpha+d)} \log t$, and slicing parameter $m_T = T^{1/(3\alpha+d)}$. Then,

$$R_i(T) \leq T^{\frac{2\alpha+d}{3\alpha+d}} \left(\frac{4(Y_R(Ld^{\alpha/2}+1)+1)}{(2\alpha+d)/(3\alpha+d)} + Y_R 2^d Z_i \log T \right) + T^{\frac{\alpha+d}{3\alpha+d}} \frac{2^{d+2} Y_R |\tilde{\mathcal{L}}_i| \beta_2}{2\alpha/(3\alpha+d)} + T^{\frac{d}{3\alpha+d}} 2^d Y_R (2|\tilde{\mathcal{L}}_i|\beta_2 + |\mathcal{L}_i|) + 2Y_R \binom{F_{\max}}{\lceil F_{\max}/2 \rceil} \beta_2,$$

i.e., $R_i(T) = O\left(Z_i T^{\frac{2\alpha+d}{3\alpha+d}}\right)$, where $Z_i = |\mathcal{L}_i| + |\tilde{\mathcal{L}}_i| \binom{F_{\max}}{\lceil F_{\max}/2 \rceil}$.

Proof: The highest orders of regret come from explorations and near optimal actions, which are $O(T^{\gamma d + z})$ and $O(T^{1+\theta})$ respectively. We need to optimize them with respect to the constraint $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R+2)t^{-z/2} \leq a_1 t^\theta$, which is assumed in Lemmas 2 and 3. The values that minimize the regret for which this constraint hold are $\theta = -z/2$, $\gamma = z/(2\alpha)$ $a_1 = 2Y_R(Ld^{\alpha/2}+1)+4$ and $z = 2\alpha/(3\alpha+d)$. The result follows from summing the bounds in Lemmas 1, 2 and 3. ∎

Remark 2: Uniform partitioning of the context may not be very efficient when the context arrivals are concentrated into some (unknown) regions of the context space. In such a case, it is better to explore and train these regions more frequently than regions with sparse context arrivals. Algorithms that start with the entire context space as a single set and which adaptively partition the context space into smaller regions as more and more users arrive can be developed to recommend multiple items to the users. Such an algorithm that will work for $N = 1$ is given in [10].

Theorem 1 indicates that agent i can achieve sublinear regret with respect to the best recommendation strategy given that the purchase probabilities of all the items in $\mathcal{F}$ are known. However, the learning rate of CBMR-d can be slow when M and $\mathcal{F}_i$ and $F_{\max}$ are large since the set of arms is combinatorial in these parameters. We would also like to note that the number of trainings and explorations, and hence the regret, can be significantly reduced when agent i knows $|\mathcal{F}_j|$ for all $j \in \mathcal{M}_{-i}$. In this case agent i can use the control function $D_{2,k_i}(t) := \max_{j\in\mathcal{M}_{-i}} \left(\binom{|\mathcal{F}_j|}{n_j(k_i)}\right) t^{2\alpha/(3\alpha+d)} \log t$ to decide when to train. As a final remark, since regret is sublinear the average reward of CBMR-d converges to the average reward of the best recommendation strategy, i.e., $\lim_{T\to\infty} R_i(T)/T = 0$.

C. Context based multiple recommendations for independent purchase probabilities (CBMR-ind)

In this subsection, we propose another learning algorithm which chooses the actions based on the sum of the expected rewards of different agents or items selected by the agent. We call the algorithm in this section *context based multiple recommendations for independent purchase probabilities* (CBMR-ind). The pseudocode of CBMR-ind is given in Fig. 4 and Fig. 5. Due to the limited space, we do not include the pseudocode of the algorithm which agent i uses to recommend

items to other agents when called by them. Basically agent i will either explore one of its own items or exploit its item with the highest estimated purchase probability in that case.

In order to exploit the independence of the purchase probabilities of the items, we decouple the action space of agent i, i.e., $\mathcal{L}_i$. For this, let $\mathcal{J}_{i,j} := \{1_j, 2_j, \ldots, N_j\}$ denote the set of the number of recommendations agent i can request from agent j, where we use the subscript j to denote that the recommendations are requested from agent j. Let $\tilde{\mathcal{J}}_i := \cup_{j \in \mathcal{M}_{-i}} \mathcal{J}_{i,j}$, $\mathcal{J}_i := \mathcal{F}_i \cup \tilde{\mathcal{J}}_i$ be the set of *arms* of agent i. We have $|\mathcal{J}_i| = |\mathcal{F}_i| + (M-1)N$, which is linear in $|\mathcal{F}_i|$, M, N.

Different from CBMR-d, which estimates the reward of each action in $\mathcal{L}_i$ separately, CBMR-ind estimates the reward of the arms in $\mathcal{J}_i$. Then, in an exploitation step, it chooses the arms in $\mathcal{J}_i$ such that a total of N items are recommended and each arm in the chosen set offers the highest estimated reward to agent i among all the arms that are not included in that set. When the item acceptance probabilities are independent of the set of items recommended together, as given in Assumption 2, this choice corresponds to choosing the action in $\mathcal{L}_i$ which offers the highest estimated reward to agent i. The advantage of CBMR-ind is that the estimated reward of an arm $u_i \in \mathcal{J}_i$ can be updated based on the received reward whenever any action $\mathcal{L}_i$ that contains arm u_i is selected by agent i. Because of this, the number of explorations is linear in $|\mathcal{K}_i|$ (or $|\mathcal{J}_i|$), instead of being linear in $|\mathcal{L}_i|$ which is combinatorial in $|\mathcal{K}_i|$.

Unlike CBMR-d, CBMR-ind does not have exploration, exploitation and training phases for actions $k_i \in \mathcal{L}_i$. Rather than that, it has exploration and exploitation phases for each arm $u \in \mathcal{F}_i$, and exploration, exploitation and training phases for each arm $u \in \tilde{\mathcal{J}}_i$. Since a combination of arms is selected at each time step, arms can be in different phases. CBMR-ind gives priority to arms that are under-explored or under-trained. It only exploits arms when there is no other under-explored or under-trained arm such that when this arm is added to the set of arms to be selected, the total number of recommendations agent i makes to its own user does not exceed N. An arm $u \in \mathcal{F}_i$ will be given priority to be explored if $N^i_{u,l} \leq D_1(t)$. An arm $u \in \tilde{\mathcal{J}}_i$ will be given priority to be trained if $N^i_{1,u,l} \leq D_{2,u}(t)$, and it will be given priority to be explored if $N^i_{1,u,l} > D_{2,u}(t)$ and $N^i_{2,u,l} \leq D_3(t)$. In order to analyze the regret of CBMR-ind, we will bound the regret in exploration and training phases by showing that the number of explorations and trainings is linear in $|\mathcal{J}_i|$. Then, we will bound the regret in the exploitation phases by bounding the regret of sub-optimal and near-optimal arms selections as in the previous subsection.

Context Based Multiple Recommendations for independent purchase probabilities (CBMR-ind for agent i):

Input: $D_1(t)$, $D_{2,u}(t)$, $u \in \tilde{\mathcal{J}}_i$, $D_3(t)$, T, m_T
Initialize: Partition $[0,1]^d$ into $(m_T)^d$ sets, indexed by the set $\mathcal{I}_T = \{1, 2, \ldots, (m_T)^d\}$. $N^i_{u,l} = 0, \forall u \in \mathcal{F}_i, l \in \mathcal{I}_T$, $N^i_{1,u,l} = 0, N^i_{2,u,l} = 0, \forall u \in \tilde{\mathcal{J}}_i, l \in \mathcal{I}_T$.
while $t \geq 1$ **do**
  $\mathcal{N}_i = \emptyset$, $\mathcal{N}^{et}_i = \emptyset$, $\mathcal{N}^e_i = \emptyset$, $\mathcal{N}^t_i = \emptyset$, $cnt = 0$
  while $|\mathcal{N}_i| < N$ **do**
    for $l = 1, \ldots, (m_T)^d$ **do**
      if $x_i(t) \in I_l$ **then**
        $l^* = l$
        for $u \in \mathcal{J}_i$ **do**
          if $u \in \mathcal{F}_i$ such that $N^i_{k,l} \leq D_1(t)$ **then**
            $\mathcal{N}_i = \mathcal{N}_i \cup \{u\}$, $\mathcal{N}^e_i = \mathcal{N}^e_i \cup \{u\}$, $cnt = cnt + 1$
          else if $u \in \tilde{\mathcal{J}}_i$ such that $N^i_{1,u,l} \leq D_{2,u}(t)$ and $cnt + u \leq N$ **then**
            $\mathcal{N}_i = \mathcal{N}_i \cup \{u\}$, $\mathcal{N}^t_i = \mathcal{N}^t_i \cup \{u\}$, $cnt = cnt + u$
          else if $u \in \tilde{\mathcal{J}}_i$ such that $N^i_{k,l} \leq D_3(t)$ and $cnt + u \leq N$ **then**
            $\mathcal{N}_i = \mathcal{N}_i \cup \{u\}$, $\mathcal{N}^e_i = \mathcal{N}^e_i \cup \{u\}$, $cnt = cnt + u$
          end if
        end for
      end if
    end for
  end while
  $N' = N - |\mathcal{N}_i|$
  $\mathcal{N}^{et}_i =$ **Choose**$(N', \mathcal{N}_i, (\bar{r}^i_{u,l^*})_{u \in \mathcal{J}_i})$
  $\mathcal{N}_i = \mathcal{N}_i \cup \mathcal{N}^{et}_i$
  Play$(\mathcal{N}_i, \mathcal{N}^e_i, \mathcal{N}^t_i, \mathcal{N}^{et}_i, (N^i_{u,l^*})_{u \in \mathcal{F}_i}, (N^i_{1,u,l^*})_{u \in \tilde{\mathcal{J}}_i}, (N^i_{2,u,l^*})_{u \in \tilde{\mathcal{J}}_i}, (\bar{r}^i_{u,l^*})_{u \in \mathcal{J}_i})$
  $t = t + 1$
end while

Fig. 4: Pseudocode for the CBMR algorithm for independent purchase probabilities.

Choose$(N, \mathcal{N}, \boldsymbol{r})$:
Select arms $u \in \mathcal{J}_i - \mathcal{N}$ such that $u \notin \mathcal{J}_{i,j}$ if $\exists\, u' \in \mathcal{N} \cap \mathcal{J}_{i,j}$ for $j \in \mathcal{M}_{-i}$, $\sum_u n_u \leq N$ and $\sum_u r_u$ is maximized.
Play$(\mathcal{N}, \mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3\, \boldsymbol{N}, \boldsymbol{r})$:
Take action $\mathcal{N}$, get the recommendations of other agents, recommend $\mathcal{N}_i(t)$ to the user.
for $u \in \mathcal{N}$ **do**
  if $u \in \mathcal{N}_i(t) \cap \mathcal{F}_i$ **then**
    Receive reward $r_u(t) = I(u \in F_i(t))$. $r_u = \frac{N_{u,l} r_u + r_u(t)}{N_{u,l}+1}$, $N_{u,l}++$.
  else if $u \in (\mathcal{N} - \mathcal{F}_i) \cap \mathcal{N}^t_i$ **then**
    Receive reward $r_u(t) = \sum_{f \in \mathcal{F}_j} I(f \in F_i(t))$, $N_{1,u,l}++$
  else
    Receive reward $r_u(t) = \sum_{f \in \mathcal{F}_j} I(f \in F_i(t))$, $r_u = \frac{N_{2,u,l} r_u + r_u(t)}{N_{2,u,l}+1}$, $N_{2,u,l}++$ $N_{2,u,l}++$
  end if
end for

Fig. 5: Pseudocode of choose and play modules.

D. Analysis of the regret of CBMR-ind

For an arm $u \in \tilde{\mathcal{J}}_i$, let $\boldsymbol{f}^*_u$ be the set of $n(u)$ items with the highest purchase probabilities in $\mathcal{F}_{j(u)}$, where $j(u)$ denotes the agent selected by agent i when arm u is selected, and $n(u)$ denotes the number of items agent $j(u)$ recommends to agent i. For an item $f_i \in \mathcal{F}_i$, let $\mu_{i,f_i}(x) := p_{f_i} q_{f_i}(x)$ be the expected reward of that item for agent i, and for an item $f \in \mathcal{F} - \mathcal{F}_i$, let $\mu_{i,f}(x) := c_{i,j} q_{f_i}(x)$ be the expected reward of that item for agent i. Recall that $\mathcal{N}^*_i(x)$ is the set of N items in $\mathcal{F}$ which maximizes agent i's expected reward for context x. Let $f_n(\mathcal{N})$ denote the item in $\mathcal{N}$ with the nth highest expected reward for agent i. For an item $f \in \mathcal{F}$, let $\underline{\mu}_{i,f,l} := \inf_{x \in I_l} \mu_{i,f}(x)$, and $\bar{\mu}_{i,f,l} := \sup_{x \in I_l} \mu_{i,f}(x)$. For the set I_l of the partition $\mathcal{I}_T$, the set of suboptimal arms for agent i at time t is given by $\mathcal{U}^i_{\theta,l}(t) := \left\{ u \in \mathcal{J}_i \text{ such that } \underline{\mu}_{i,f_N(\mathcal{F}),l} - \bar{\mu}_{i,f_{n(u)}(\mathcal{F}_{j(u)}),l} \geq a_1 t^{\theta} \right\}$, where we will optimize over a_1 and θ as we did in Section III-B. Similar to the approach we took in Section III-B, we

divide the regret into three parts: $R_i^e(T)$, $R_i^s(T)$ and $R_i^n(T)$, and bound them individually. Different from the analysis of CBMR-d, here $R_i^s(T)$ denotes the regret in exploitation steps in which agent i chooses at least one suboptimal arm while $R_i^n(T)$ denotes the regret in exploitation steps in which all the arms chosen by agent i are near-optimal. In the following lemma, we bound the regret of CBMR-ind due to explorations and trainings.

Lemma 4: When CBMR-ind is run by agent i with parameters $D_1(t) = t^z \log t$, $D_{2,u}(t) = \binom{F_{\max}}{n(u)} t^z \log t$, $u \in \tilde{\mathcal{J}}_i$, $D_3(t) = t^z \log t$ and $m_T = \lceil T^\gamma \rceil$, where $0 < z < 1$ and $0 < \gamma < 1/d$, we have

$$E[R_i^e(T)] \leq Y_R 2^d (|\mathcal{J}_i| + (M-1)N) T^{\gamma d} + Y_R 2^d \left(|\mathcal{J}_i| + (M-1) \sum_{a=1}^{N} \binom{F_{\max}}{a} \right) T^{z+\gamma d} \log T.$$

Proof: For a set $I_l \in \mathcal{I}_T$, the regret due to explorations is bounded by $|\mathcal{J}_i| \lceil T^z \log T \rceil$. Agent i spends at most $\sum_{z=1}^{N} \binom{F_{\max}}{z} \lceil T^z \log T \rceil$ time steps to train agent j. Note that this is the worst-case number of trainings for which agent j does not learn about the purchase probabilities of its items in set I_l from its own users, and from the users of agents other than agent i. The result follows from summing over all sets in $\mathcal{I}_T$. ∎

In the next lemma, we bound $E[R_i^s(T)]$.

Lemma 5: When CBMR-ind is run with parameters $D_1(t) = t^z \log t$, $D_{2,u}(t) = \binom{F_{\max}}{n(u)} t^z \log t$, $u \in \tilde{\mathcal{J}}_i$, $D_3(t) = t^z \log t$ and $m_T = \lceil T^\gamma \rceil$, where $0 < z < 1$ and $0 < \gamma < 1/d$, given that $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R+2)t^{-z/2} \leq a_1 t^\theta$ we have $E[R_i^s(T)] \leq 2^{d+1} Y_R |\mathcal{J}_i| \beta_2 T^{\gamma d} + 2^{d+2} N Y_R |\tilde{\mathcal{J}}_i| \beta_2 T^{\gamma d + z/2}/z$.

Proof: Let Ω denote the space of all possible outcomes, and let w be a sample path. Let $\mathcal{W}_l^i(t) := \{w : S_{i,l}(t) = \emptyset\}$ denote the event that CBMR-ind is in the exploitation phase at time t. The idea is to bound the probability that agent i selects at least one suboptimal arm in an exploitation step, and then using this to bound the expected number of times a suboptimal arm is selected by agent i. Let $\mathcal{V}_{u,l}^i(t)$ be the event that a suboptimal action $u \in \mathcal{J}_i$ is chosen at time t by agent i. We have $E[R_s(T)] \leq Y_R \sum_{l \in \mathcal{I}_T} \sum_{t=1}^{T} \sum_{u \in \mathcal{U}_{\theta,l}^i(t)} P(\mathcal{V}_{u,l}^i(t), \mathcal{W}_l^i(t))$.

Similar to the proof of Lemma 2 in [10], by using a Chernoff bound it can be shown that for any $u \in \mathcal{U}_{\theta,l}^i(t) \cap \mathcal{F}_i$, we have $P(\mathcal{V}_{u,l}^i(t), \mathcal{W}_l^i(t)) \leq 2/t^2$, and for any $u \in \mathcal{U}_{\theta,l}^i(t) \cap \tilde{\mathcal{J}}_i$, we have $P(\mathcal{V}_{u,l}^i(t), \mathcal{W}_l^i(t)) \leq 2/t^2 + 2N|\mathcal{F}_{u(j)}|\beta_2/t^{1-z/2}$. Here different from Lemma 2 in [10], N comes from a union bound which is required to bound the probability that agent j will recommend an item which is not in the set of the best $n(u)$ items of agent j when agent i chooses arm u. We get the final regret result by summing the $P(\mathcal{V}_{u,l}^i(t), \mathcal{W}_l^i(t))$s over the set of suboptimal arms in each set in the partition $\mathcal{I}_T$, over the sets in the partition $\mathcal{I}_T$, and over time, and then by using the result in Appendix A in [10]. ∎

Different from Lemma 2, $E[R_i^s(T)]$ is linear in $\mathcal{J}_i$ instead of $\mathcal{L}_i$. In the next lemma, we bound the regret due to near-optimal arm selections by agent i by time T, i.e., $E[R_i^n(T)]$.

Lemma 6: When CBMR-ind is run with parameters $D_1(t) = t^z \log t$, $D_{2,u}(t) = \binom{F_{\max}}{n(u)} t^z \log t$, $u \in \tilde{\mathcal{J}}_i$, $D_3(t) = t^z \log t$ and $m_T = \lceil T^\gamma \rceil$, where $0 < z < 1$ and $0 < \gamma < 1/d$, given that $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R+2)t^{-z/2} \leq a_1 t^\theta$ we have $E[R_i^n(T)] \leq (2Na_1 T^{1+\theta})/(1+\theta) + 2Y_R N F_{\max} \beta_2$.

Proof: Since agent i can choose at most N arms at each time step, if all arms chosen at time t by agent i and by the other agents called by agent i are near-optimal, then the contribution to the regret is bounded by $2Na_1 t^\theta$. This is because for a near optimal arm $u \in \mathcal{F}_i$, the contribution to the regret is at most $a_1 t^\theta$, while for a near optimal arm in $u \in \tilde{\mathcal{J}}_i$, if agent $j(u)$ chooses its near optimal items to recommend to agent i, the contribution to the regret is at most $2a_1 t^\theta$. Therefore, the total regret due to near optimal arm selections by agent i and the agents i calls by time T is upper bounded by $2Na_1 \sum_{t=1}^{T} t^\theta \leq (2Na_1 T^{1+\theta})/(1+\theta)$ from the result in Appendix A in [10].

However, each time a near optimal arm in $u \in \tilde{\mathcal{J}}_i$ is chosen in an exploitation step, there is a small probability that an item chosen recommended by agent $j(u)$ is a suboptimal one. Using a result similar to Lemma 3 in [10], the expected number of times a suboptimal arm is chosen by the called agent is bounded by $n(u)F_{\max}\beta_2$. Each time a suboptimal item is chosen by the called agent $j(u)$, the regret can be at most Y_R. The result follows from summing these terms. ∎

Combining the above lemmas, we obtain the finite time, uniform regret bound for agents using CBMR-ind given in the following theorem.

Theorem 2: Let CBMR-ind run with exploration control functions $D_1(t) = D_3(t) = t^{2\alpha/(3\alpha+d)} \log t$, $D_{2,u}(t) = \binom{F_{\max}}{n(u)} t^{2\alpha/(3\alpha+d)} \log t$, $u \in \tilde{\mathcal{J}}_i$, $D_3(t) = t^{2\alpha/(3\alpha+d)} \log t$ and slicing parameter $m_T = T^{1/(3\alpha+d)}$. Then,

$$R_i(T) \leq T^{\frac{2\alpha+d}{3\alpha+d}} \left(\frac{4N(Y_R(Ld^{\alpha/2}+1)+1)}{(2\alpha+d)/(3\alpha+d)} + Y_R 2^d Z_i' \log T \right) + T^{\frac{\alpha+d}{3\alpha+d}} \frac{2^{d+2} N Y_R |\tilde{\mathcal{J}}_i| \beta_2}{2\alpha/(3\alpha+d)} + T^{\frac{d}{3\alpha+d}} 2^d Y_R (2|\mathcal{J}_i|\beta_2 + |\mathcal{J}_i| + (M-1)N) + 2Y_R N F_{\max} \beta_2,$$

i.e., $R_i(T) = O\left(Z_i' T^{\frac{2\alpha+d}{3\alpha+d}}\right)$, where $Z_i' = |\mathcal{J}_i| + (M-1)\sum_{a=1}^{N} \binom{F_{\max}}{a}$.

Proof: The highest orders of regret come from explorations, trainings and near optimal arms, which are $O(T^{\gamma d + z})$ and $O(T^{1+\theta})$ respectively. We need to optimize them with respect to the constraint $2LY_R(\sqrt{d})^\alpha t^{-\gamma\alpha} + 2(Y_R+2)t^{-z/2} \leq a_1 t^\theta$, which is assumed in Lemmas 5 and 6. This gives us $\theta = -z/2$, $\gamma = z/(2\alpha)$ $a_1 = 2Y_R(Ld^{\alpha/2}+1) + 4$ and $z = 2\alpha/(3\alpha+d)$. The result follows from summing the bounds in Lemmas 4, 5 and 6. ∎

The result of Theorem 2 indicates that the regret of CMBR-ind is sublinear in time and has the same time order as the regret of CBMR-d. However, the regret of CMBR-ind depends linearly on $|\mathcal{J}_i|$, which is much better than the linear dependence of the regret of CBMR-d on $|\mathcal{L}_i|$.

E. Comparison of CBMR-d and CBMR-ind

In this section we compare CBMR-d and CBMR-ind in terms of their regret bounds, exploration rates and memory requirements. Note that the regret bound of CBMR-d depends on the size of the action space $\mathcal{L}_i$, which grows exponentially with N, and is an N degree polynomial of M and $|\mathcal{F}_i|$. In

contrast the size of the arm space $\mathcal{J}_i$ grows linearly in M, N, and $\mathcal{F}_i$. This is due to the fact that CBMR-d explores and exploits each action without exploiting the correlations between different actions. When the purchase probabilities depend on the set of items offered together, in the worst case there may be no correlation between rewards of different sets of recommendations (actions), and therefore the best one can do is to form estimates of each action independently. However, when the purchase probabilities are independent of the items offered together, since the expected reward of agent i from an action is the sum of the expected rewards of the individual arms chosen by agent i in that action, substantial improvement over the regret bound is possible due to smaller number of explorations and trainings.

Another advantage of CBMR-ind is that it requires a significantly smaller amount of memory than CBMR-d. CBMR-d needs to keep sample mean reward estimates for all actions and for all partitions of the context space, while CBMR-ind only needs to keep sample mean reward estimates for all arms and for all partitions of the context space. Therefore the memory requirement of CBMR-ind is $O(|\mathcal{J}_i|/|\mathcal{L}_i|)$ of the memory requirement of CBMR-d. The memory savings become much more important as T increases since memory scales linearly in $|\mathcal{I}_T|$. As a final remark, we would like to note that the advantage of CBMR-ind over CBMR-d only holds when the purchase probabilities of items are independent of each other. In general when the purchase probabilities depend on each other in an arbitrary way, the regret bound of any learning algorithm will be linear in $|\mathcal{L}_i|$ (see e.g., [34]). However, in Section V, we observe that CBMR-ind have better performance than CBMR-d in numerical results for dependent purchase probabilities.

IV. Performance as a Function of the Network

In our analysis in the previous sections we assumed that all agents are directly connected to each other. In reality some agents may not be connected to each other. For example, agents i and j can both be connected to agent j', but there may not exist a link between agent i and agent j. This can happen when, for example, a pair of companies has a trade agreement between each other, but there is no trade agreement between i and j. We assume that whenever a trade link exists between agents i and j it is bidirectional. In this case, even though i cannot ask j for items to recommend, i can ask j' and j' can ask j for an item. Then, if i sells j's item, agent i will get commission $c_{i,j'}$ from j', while agent j' will get commission $c_{j',j} + c_{i,j'}$ from agent j so that it recovers the payment it made to agent i from agent j. We call agent j' the *relay agent*. Let $\mathcal{M}_i$ denote the set of agents that are directly connected to agent i.

Using CBMR-d and CBMR-ind with only $\mathcal{M}_i$, agent i can achieve the sublinear regret bounds given in Theorems 1 and 2, with respect to the optimal distributed recommendation policy involving the agents i and $\mathcal{M}_i$. However, since agent i cannot exploit the advantage of the items of other agents which are in $\mathcal{M}_{-i} - \mathcal{M}_i$, its regret can be linear with respect to the optimal distributed recommendation policy involving all the agents. Let d_k be the maximum degree of the network, which is the maximum number of connections an agent can have. Assume that the network is connected and the longest path in the network is d_h hops for agent i. CBMR-d and CBMR-ind can be modified in the following way to account for agents distributed over the network. Let $\mathcal{M}_{i,j}(p)$ be the set of relay agents between agents i and j when they are connected through path p. Let $j_1(p)$ be the agent that is connected directly to agent i in path p, let $j_2(p)$ be the agent that is connected directly to agent $j_1(p)$, and so on. Then, the commission framework is modified such that when agent i sells agent j's item it gets a commission $c_{i,j_1(p)}$ from agent $j_1(p)$, agent $j_1(p)$ gets a commission $c_{j_1(p),j_2(p)}+c_{i,j_1(p)}$ from agent $j_2(p)$, and so on, such that $c_{j_{|p|-2}(p),j} \leq p_{k_j}$, where k_j is the item recommended by agent j to agent i's user, so that all agents benefit from this transaction and it is better for them to cooperate based on the rules of the commission than to act on their own. We assume that agent j will not recommend an item to agent $j_{|p|-2}$ if $c_{j_{|p|-2}(p),j} > p_{k_j}$[4]. Assume a connected network of agents $G(\mathcal{M}, E)$ in which the maximum degree is d_k and the longest path has d_h hops, where E denotes the set of links between the agents. Assume that the commissions $c_{i,j} > 0$ are given between any agent i and j with direct links. We define agent i's regret $R_i^{G(\mathcal{M},E)}(T)$ to be the difference between the expected total reward of the optimal policy for which agent i has access to the set of all the items of all agents in the network $G(\mathcal{M}, E)$ (but if the item is in $\mathcal{F}_j$, it gets the commission from agent j', which is the agent that is directly connected to agent i in the smallest commission path from agent i to agent j) and the expected total reward of the learning algorithm used by agent i. The exploration and training control functions are run using $(d_k)^{d_h} F_{\max}$ instead of $F_{\max}$. This way agent i will help the relay agents learn the other agent's recommendations accurately such that sublinear regret bounds can be achieved. The following theorem gives a bound on the regret of agents when they use the modified version of CBMR-ind discussed above (CBMR-ind-N).[5]

Theorem 3: When Assumption 2 holds, if CBMR-ind-N is run by all agents in $G(\mathcal{M}, E)$, with exploration control functions $D_1(t) = D_3(t) = t^{2\alpha/(3\alpha+d)} \log t$, $D_{2,u}(t) = \binom{(d_k)^{d_h} F_{\max}}{n(u)} t^{2\alpha/(3\alpha+d)} \log t$, $u \in \tilde{\mathcal{J}}_i$, and slicing parameter $m_T = T^{1/(3\alpha+d)}$ for agent i, and if commissions are such that all agents $j \in \mathcal{M}_{-i}$ will get a positive reward when their items are sold by agent i[6], we have,

$$R_i^{G(\mathcal{M},E)}(T) \leq T^{\frac{\alpha+d}{3\alpha+d}} \frac{2^{d+2} N Y_R |\tilde{\mathcal{J}}_i| \beta_2}{2\alpha/(3\alpha+d)} + 2Y_R N (d_k)^{d_h} F_{\max} \beta_2$$

[4]However, there is an indirect benefit to agent j by recommending an item for another agent, which is the benefit of obtaining information about the purchase probability of its own item for the user with a specific context. Therefore, in some scenarios it may be better for agent j to recommend an item to another agent even when the cost of commission is greater than the price of the item. Since the exploration and training phases of our algorithms are designed to ensure that an agent's estimates about purchase probabilities are accurate enough, the agent's benefit from taking such an action to improve its estimates is negligible. However, such a possibility should be investigated when deriving lower bounds on the regret, which is out of the scope of this paper.

[5]A similar bound can also be proven for the modified version of CMBR-d.

[6]If the commission is greater than the price of the item, then that agent can be removed from the set of agents which agent i can cooperate with. Then our results will hold for the remaining set of agents.

$$+ T^{\frac{2\alpha+d}{3\alpha+d}} \left(\frac{4N(Y_R(Ld^{\alpha/2}+1)+1)}{(2\alpha+d)/(3\alpha+d)} + Y_R 2^d Z_i' \log T \right)$$
$$+ T^{\frac{d}{3\alpha+d}} 2^d Y_R (2|\mathcal{J}_i|\beta_2 + |\mathcal{J}_i| + (M-1)N),$$

i.e., $R_i^{G(\mathcal{M},E)}(T) = O\left(Z_i' T^{\frac{2\alpha+d}{3\alpha+d}}\right)$, where $Z_i' := |\mathcal{J}_i| + (M-1)\sum_{a=1}^{N} \binom{(d_k)^{d_h} F_{\max}}{a}$.

Proof: The proof is similar to the proof of Theorem 2, but more explorations and trainings are required to ensure that all agents in all paths learn the purchase probabilities of their items accurately for all user contexts before exploiting. ∎

Theorem 3 indicates that the regret increases exponentially in d_h and is an Nth degree polynomial in d_k. While this makes $CBMR-ind-N$ impractical in non-small world networks, $CBMR-ind-N$ can achieve small regret in small world networks in which most agents are not directly connected to each other but all agents can be reached a with small number of hops. Because of the additional commission the relay agent gets, an item which j will recommend when it is directly connected to agent i may not be recommended by j when it is connected via agent j'. This results in sub-optimality compared to the case when all agents are connected. In Section V we numerically compare the effect of the agents' commissions on the performance.

More refined versions of Theorem 3 can be derived if we focus on specific networks. One interesting network structure is when there is a monopolist agent which is directly connected to all of the other agents, while other agents are only directly connected to the monopolist agent. Corollary 1 gives the regret bound for agent i when it is the monopolist agent (network $G_1(\mathcal{M},E)$), and Corollary 2 gives the regret bound for agent i when it is not the monopolist agent (network $G_2(\mathcal{M},E)$).

Corollary 1: When agent i is the monopolist agent, if it runs CBMR-ind-N with $D_{2,u}(t) = \binom{F_{\max}}{n(u)} t^{2\alpha/(3\alpha+d)} \log t$, $u \in \tilde{\mathcal{J}}_i$ and everything else remaining the same as in Theorem 3, it will achieve $R_i^{G_1(\mathcal{M},E)}(T) = O\left((\mathcal{F}_i + (M-1)N) T^{\frac{2\alpha+d}{3\alpha+d}}\right)$.

Corollary 2: When agent i is a non-monopolist agent, if it runs CBMR-ind-N with $D_{2,u}(t) = \binom{M^2 F_{\max}}{n(u)} t^{2\alpha/(3\alpha+d)} \log t$, $u \in \tilde{\mathcal{J}}_i$ and everything else remaining the same as in Theorem 3, it will achieve $R_i^{G_2(\mathcal{M},E)}(T) = O\left((\mathcal{F}_i + N) M^{2N} T^{\frac{2\alpha+d}{3\alpha+d}}\right)$.

Corollaries 1 and 2 imply that the learning rate is much faster and hence the regret is much smaller when an agent has direct connections with more agents. This is because agent i learns directly about the purchase probabilities of items in agent j's inventory when it is directly connected to it, while the learning is indirect when there is no direct connection between agents i and j.

V. Numerical Results

A. Description of the Data Set

The Amazon product co-purchasing network data set includes product IDs, sales ranks of the products, and for each product the IDs of products which are frequently purchased with that product. This data is collected by crawling the Amazon website [35] and contains $410,236$ products and $3,356,824$ edges between products that are frequently co-purchased together. We simulate CBMR-ind and CBMR-d using the following distributed data sets adapted based on Amazon data. For a set of N_1 chosen products, we take that product and the F_1 products that are frequently co-purchased with that product.

The set of products that are taken in the first step of the above procedure is denoted by $\mathcal{C}_h$. The set of all products $\mathcal{F}$ contains these N_1 products and the products co-purchased frequently with them, which we denote by set $\mathcal{C}_f$. We assume that each item has a unit price of 1, but have different purchase probabilities for different types of users. Since user information is not present in the data set, we generate it by assuming that a user searches for a specific item. This search query will then be the context information of the user. The context space is discrete, thus we set $\mathcal{I}_T = \mathcal{C}_h$. Based on this, the agent that the user arrives to recommends N items to the user. The agent's goal is to maximize the total number of items sold to the users.

We generate the purchase probabilities in the following way: For dependent purchase probabilities, when n of the products recommended for context x are in the set of frequently co-purchased products, then the purchase probability of each of these products will be $g_c(n) = 1 - an$, where $0 < a < 1/N$. For the other $N - n$ products which are not frequently co-purchased in context x, their purchase probability is $g_{nc} = b$, where $0 < b < 1 - a$. For independent purchase probabilities, when a product recommended for context x is in the set of frequently co-purchased products, the purchase probability of that product will be g_c. When it is not, the purchase probability of that product will be g_{nc}, for which we have $g_c > g_{nc}$.

We assume that there are 3 agents and evaluate the performance of agent 1 based on the number of users arriving to agent 1 with a specific context x^*, which we take as the first item in set $\mathcal{C}_h$. We assume that $T = 100,000$, which means that $100,000$ users with context x^* arrive to agent 1. Since the arrival rate of context x^* can be different for the agents, we assume arrivals with context x^* to other agents are drawn from a random process. We take $N_1 = 20$, $F_1 = 2$ and $N = 2$. As a result, we get 30 distinct items in $\mathcal{F}$ which are distributed among the agents such that $|\mathcal{F}_i| = 10$. Since the context space is discrete we have $d = 1$, and there is no Lipschitz assumption on the purchase probabilities as given in Assumptions 1 and 2, hence we take $\alpha = 1/13$ such that $2\alpha/(3\alpha + d) = 1/8$. Unless otherwise stated, we assume that $g_c = 0.1$, $g_{nc} = 0.01$, $a = 0.5$ and $c_{i,j} = 0.5$.

B. Comparison of the reward and regret of CBMR-d and CBMR-ind for dependent purchase probabilities

We run both CBMR-d and CBMR-ind for dependent purchase probabilities when both items that are frequently co-purchased with context (product) x^* are in $\mathcal{F}_1$. For this case, the optimal policy for agent 1 will recommend one of the frequently co-purchased items and another item in agent 1's inventory to the user with context x^*. Expected total reward of the optimal policy, total rewards of CBMR-d and CBMR-ind, and the number of training steps of CBMR-d and CBMR-ind are shown in Table III for agent 1. We have $|\mathcal{L}_1| = 68$ and $|\mathcal{J}_1| = 14$. We see that the total reward of CBMR-ind is higher than the total reward of CBMR-d for this case. This is due to

	Optimal	CBMR-d	CBMR-ind
Total Reward	11000	8724	9485
Trainings	-	5342	12391

TABLE III: Total rewards of the optimal policy, CBMR-d and CBMR-ind, and number of trainings of CBMR-d and CBMR-ind for dependent purchase probabilities.

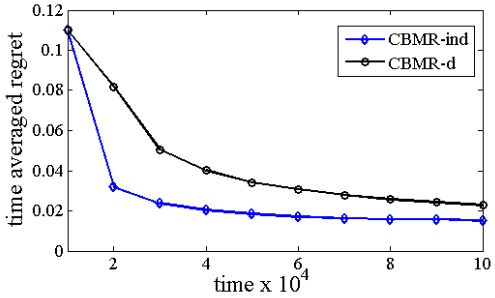

Fig. 6: Time averaged regrets of CBMR-d and CBMR-ind for dependent purchase probabilities.

the fact that CBMR-d spends more than twice the time CBMR-ind spends in training steps since it trains actions separately. The time averaged regrets of CBMR-d and CBMR-ind are given in Fig. 6. From this, we can observe that CBMR-ind outperforms CBMR-d in all time steps, and the time averaged regret goes to 0. From these results it seems that CBMR-ind is a good alternative to CBMR-d even for dependent purchase probabilities.

C. Effect of commission on the performance

In this subsection we numerically simulate the effect of commissions $c_{1,j}$ that agent 1 charges to other agents on the total reward of agent 1. We consider CBMR-ind for independent purchase probabilities. We assume that agent 1 has one of the frequently co-purchased items for context x^*, while agent 3 has the other frequently co-purchased item. The total reward of agent 1 as a function of the commissions $c_{1,2} = c_{1,3} = c$ is given in Table IV. We note that there is no increase in the total reward when the commission is increased to 0.1, because this amount is not enough to incentivize agent 1 to recommend other agent's items. However, for commissions greater than 0.1, the optimal policy recommends the two frequently co-purchased items together, hence agent 1 learns that it should get recommendations from other agents. Therefore, when commission is greater than 0.1, the total reward of the agent is increasing in the commission. Another remark is that $c = 0$ corresponds to the case when agent 1 is not connected to the other agents. Therefore, this example also illustrates how the rewards change as network connectivity changes. Since prices of items are set to 1 in this section, the commission agent 1 charges can increase up to 1. But if the prices are different, then the commission cannot exceed the recommended item's price. In theory, in order to maximize its total reward from its own users, agent i can adaptively select its commissions $c_{i,j}$, $j \in \mathcal{M}_{-i}$ based on what it learned about the purchase probabilities. CBMR-d and CBMR-ind can be modified to adaptively select the commissions. Due to the limited space, we leave this as a future research topic.

D. Effect of the set of items of each agent on the performance

In this subsection we compare three cases for independent purchase probabilities when agents use CBMR-ind. In C-1 agent 1 has both items that are frequently co-purchased in

Commission c	0	0.1	0.2	0.3	0.4	0.5
Reward (CBMR-ind)	10471	10422	11476	12393	13340	14249

TABLE IV: The total reward of agent 1 as a function of the commission it charges to other agents.

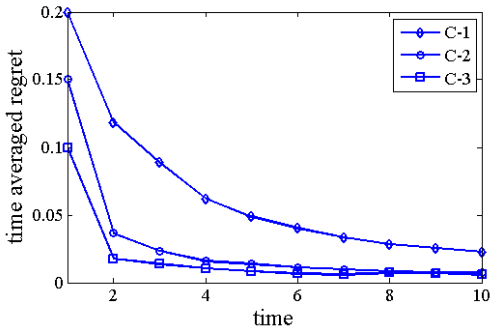

Fig. 7: Time averaged regret of CBMR-ind for independent purchase probabilities when agent 1 has both frequently co-purchased items (C-1), only one of the frequently co-purchased items (C-2) and none of the frequently co-purchased items (C-3).

context x^*, in C-2 it has one of the items that is frequently co-purchased in context x^*, and in C-3 it has none of the items that are frequently co-purchased in context x^*. The total reward of agent 1 for these cases is 17744, 14249 and 9402 respectively, while the total expected reward of the optimal policy is 20000, 15000 and 10000 respectively. Note that the total reward for C-3 is almost half of the total reward for C-1 since the commission agent 1 gets for a frequently co-purchased item is 0.5. The time averaged regret of CBMR-ind for all these cases is given in Figure 7. We see that the convergence rate for C-1 is slower than C-2 and C-3. This is due to the fact that in all of the trainings step in C-1 a suboptimal set of items is recommended, while for C-2 and C-3 in some of the training steps the optimal set of items is recommended.

VI. Conclusion

In this paper we have presented two novel algorithms for multi-agent learning within a decentralized social network, and characterized the effect of different network structures on performance. Our algorithms are able to achieve sublinear regret in all cases, with the regret being much smaller if the user's acceptance probabilities for different items are independent. This paper can be used as a groundwork for agents in many different types of networks to cooperate in a mutually beneficial manner, from companies, charities, and celebrities who wish to generate revenue to artists, musicians, and photographers who simply want to have their work publicized. By cooperating in a decentralized manner and using our algorithms, agents have the benefit of retaining their autonomy and privacy while still achieving near optimal performance.

References

[1] K.-C. Chen, M. Chiang, and H. Poor, "From technological networks to social networks," *Selected Areas in Communications, IEEE Journal on*, vol. 31, no. 9, pp. 548–572, 2013.

[2] V. Krishnamurthy, "Quickest detection pomdps with social learning: Interaction of local and global decision makers," *Information Theory, IEEE Transactions on*, vol. 58, no. 8, pp. 5563–5587, 2012.

[3] V. Krishnamurthy and H. V. Poor, "Social learning and bayesian games in multiagent signal processing: How do local and global decision makers interact?" *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 43–57, 2013.

[4] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, "Non-bayesian social learning," *Games and Economic Behavior*, vol. 76, no. 1, pp. 210–225, 2012.

[5] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *Automatic Control, IEEE Transactions on*, vol. 54, no. 1, pp. 48–61, 2009.

[6] A. Slivkins, "Contextual bandits with similarity information," *arXiv preprint arXiv:0907.3986*, 2009.

[7] M. Dudik, D. Hsu, S. Kale, N. Karampatziakis, J. Langford, L. Reyzin, and T. Zhang, "Efficient optimal learning for contextual bandits," *arXiv preprint arXiv:1106.2369*, 2011.

[8] J. Langford and T. Zhang, "The epoch-greedy algorithm for contextual multi-armed bandits," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1096–1103, 2007.

[9] W. Chu, L. Li, L. Reyzin, and R. E. Schapire, "Contextual bandits with linear payoff functions," in *Proc. of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[10] C. Tekin and M. van der Schaar, "Distributed online learning via cooperative contextual bandits, arxiv:1308.4568," *submitted to Signal Processing, IEEE Transactions on,*, 2013.

[11] ——, "Decentralized online big data classification - a bandit framework," *arXiv preprint arXiv:1308.4565*, 2013.

[12] Y. Gai, B. Krishnamachari, and R. Jain, "Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations," *IEEE/ACM Transactions on Networking (TON)*, vol. 20, no. 5, pp. 1466–1478, 2012.

[13] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. of the 19th international conference on World wide web*. ACM, 2010, pp. 661–670.

[14] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, p. 235256, 2002.

[15] K. Crammer and C. Gentile, "Multiclass classification with bandit feedback using adaptive regularization," 2011.

[16] A. Anandkumar, N. Michael, and A. Tang, "Opportunistic spectrum access with multiple players: Learning under competition," in *Proc. of IEEE INFOCOM*, March 2010.

[17] K. Liu and Q. Zhao, "Distributed learning in multi-armed bandit with multiple players," *Signal Processing, IEEE Transactions on*, vol. 58, no. 11, pp. 5667–5681, 2010.

[18] C. Tekin and M. Liu, "Online learning of rested and restless bandits," *Information Theory, IEEE Transactions on*, vol. 58, no. 8, pp. 5588–5611, 2012.

[19] H. Liu, K. Liu, and Q. Zhao, "Learning in a changing world: Restless multiarmed bandit with unknown dynamics," *Information Theory, IEEE Transactions on*, vol. 59, no. 3, pp. 1902–1916, 2013.

[20] C. Tekin and M. Liu, "Online learning in decentralized multi-user spectrum access with synchronized explorations," in *Proc. of IEEE MILCOM 2012*, 2012.

[21] H. Liu, K. Liu, and Q. Zhao, "Learning in a changing world: Non-bayesian restless multi-armed bandit," *Techinal Report, UC Davis*, October 2010.

[22] Y. Deshpande and A. Montanari, "Linear bandits in high dimension and recommendation systems," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE, 2012, pp. 1750–1754.

[23] P. Kohli, M. Salek, and G. Stoddard, "A fast bandit algorithm for recommendations to users with heterogeneous tastes," 2013.

[24] T. Hofmann, "Latent semantic models for collaborative filtering," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 89–115, 2004.

[25] N. Sahoo, P. V. Singh, and T. Mukhopadhyay, "A hidden markov model for collaborative filtering," *MIS Quarterly*, vol. 36, no. 4, pp. 1329–1356, 2012.

[26] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *Internet Computing, IEEE*, vol. 7, no. 1, pp. 76–80, 2003.

[27] U. Panniello, A. Tuzhilin, and M. Gorgoglione, "Comparing context-aware recommender systems in terms of accuracy and diversity," *User Modeling and User-Adapted Interaction*, pp. 1–31, 2012.

[28] K. Miyahara and M. J. Pazzani, "Collaborative filtering with the simple bayesian classifier," in *PRICAI 2000 Topics in Artificial Intelligence*. Springer, 2000, pp. 679–689.

[29] M. OConnor and J. Herlocker, "Clustering items for collaborative filtering," in *Proceedings of the ACM SIGIR workshop on recommender systems*, vol. 128. UC Berkeley, 1999.

[30] G. Shani, R. I. Brafman, and D. Heckerman, "An mdp-based recommender system," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2002, pp. 453–460.

[31] C.-N. Ziegler, "Towards decentralized recommender systems." Ph.D. dissertation, Citeseer, 2005.

[32] M. Balabanović and Y. Shoham, "Fab: content-based, collaborative recommendation," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[33] C. Tekin, S. Zhang, and M. van der Schaar, "Distributed online learning in social recommender systems," *http://medianetlab.ee.ucla.edu/papers/JSTSPonline.pdf*, 2013.

[34] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, pp. 4–22, 1985.

[35] J. Leskovec, L. A. Adamic, and B. A. Huberman, "The dynamics of viral marketing," *ACM Transactions on the Web (TWEB)*, vol. 1, no. 1, p. 5, 2007.